Červenka Consulting s.r.o.
Na Hrebenkach 55
150 00 Prague
Czech Republic
Phone: +420 220 610 018
E-mail: cervenka@cervenka.cz
Web: http://www.cervenka.cz

ATENA Program Documentation
Part 1

Theory

Written by
Vladimír Červenka, Libor Jendele,
and **Jan Červenka**

Prague, March 31st, 2021

Ackowledgements:

The software was developed with partial support of Eurostars funding program.

The following models were developed with the financial support of TA ČR: CC3DNonLinCementitious2HPRFC, CC3DNonLinCementitious2FRC, CC3DNonLinCementitious2SHCC, CCModelGeneral, CCTransportMaterial

Durability Anslysis module has been developed during the project TA04031458, „Software for prediction and modelling of durability and safety of transportation structures" funded by TA ČR

ATENA3DPrint module has been developed during the project TF04000051, "digiCON2 – Simulation Service for Digital Concrete Production" funded by TA ČR.

ATENA modul CeSTaR – for modelling of prefabricated reinforced concrete segments with multi-spiral reinforcement was developed under the project TF05000040 „CeSTaR - Computer simulation and experimental validation - complex service for flexible and efficient design of pre-cast concrete" funded by TA ČR.

Contents

1 CONTINUUM GOVERNING EQUATIONS

1.1 Introduction

This chapter presents the general governing continuum equations for nonlinear analysis. In general, there exist many variants of nonlinear analysis depending on how many nonlinear effects are accounted for. Hence, this chapter first introduces some basic terms and entities commonly used for nonlinear structural analyses, and then it concentrates on formulations that are implemented in ATENA.

It is important to realize that the whole structure does not have to be analyzed using a full nonlinear formulation. However, a simplified (or even linear) formulation can be used in many cases. It is a matter of engineering knowledge and practice to assess, whether the inaccuracies due to a simplified formulation are acceptable or not.

The simplest formulation, i.e., linear formulation, is characterized by the following assumptions:

The constitutive equation is linear, i.e., the generalized form of Hook's law is used.

The geometric equation is linear; that is, the quadratic terms are neglected. It means that during analysis, we neglect the change of shape and position of the structure.

Both loading and boundary conditions are conservative, i.e., they are constant throughout the whole analysis irrespective of the structural deformation, time etc.

Generally, linear constitutive equations can be employed for a material, which is far from its failure point, usually up to 50% of its maximum strength. Of course, this depends on the type of material, e.g., rubber needs to be considered as a nonlinear material earlier. But for usual civil engineering materials, the previous assumption is satisfactory.

Geometric equations can be considered linear if the deflections of a structure are much smaller than its dimensions. This must be satisfied not only for the whole structure but also for its parts. Then the geometric equations for the loaded structure can then be written using the original (unloaded) geometry.

It is also important to realize that a linear solution is permissible only in the case of small strains. This is closely related to the material property because if strains are high, the stresses are usually, although not necessarily, high as well.

Despite the fact that for the vast majority of structures linear simplifications are quite acceptable, there are structures when it is necessary to take into account some nonlinear behavior. The resulting governing equations are then much more complicated, and normally they do not have a closed-form solution. Consequently, some nonlinear iterative solution schemes must be used (see Chapter Solution of Nonlinear Equations further in this document).

Nonlinear analysis can be classified according to a type of nonlinear behavior:

Nonlinear material behavior only needs to be accounted for. This is the most common case for ordinary reinforced concrete structures. Because of serviceability limitations, deformations are relatively small. However, the very low tensile strength of concrete needs to be accounted for.

Deformations (either displacements only or both displacements and rotations) are large enough such that the equilibrium equations must use the deformed shape of the structure. However, the relative deformations (strains) are still small. The complete form of the geometric

equations, including quadratic terms, has to be employed, but constitutive equations are linear. This group of nonlinear analyses includes most stability problems.

The last group uses nonlinear both material and geometric equations. In addition, it is usually not possible to suddenly apply the total value of load, but it is necessary to integrate in time increments (or loading increments). This is the most accurate and general approach but unfortunately, is also the most complicated.

There are two basic possibilities for formulating the general structural behavior based on its deformed shape:

Lagrange formulation:

In this case, we are interested in the behavior of infinitesimal particles of volume dV. Their volume will vary dependent on the loading level applied and, consequently, on the extent of current deformations. This method is usually used to calculate civil engineering structures.

Euler formulation:

The essential idea of Euler's formulation is to study the "flow" of the structural material through infinitesimal and fixed volumes of the structure. This is the favorite formulation for fluid analysis, analysis of gas flow, tribulation etc., where large material flows exist.

For structural analysis, however, the Lagrangian formulation is better, and therefore the attention will be restricted to this. Two forms of the Lagrangian formulation are possible. The governing equations can either be written with respect to the original undeformed configuration at time t = 0 or with respect to the most recent deformed configuration at time t. The former case is called Total Lagrangian formulation (TL), while the latter one is called the Updated Lagrangian formulation (UL).

It is difficult to say which formulation is better because both have their advantages and drawbacks. Usually, it depends on a particular structure being analyzed and which one to use is a matter of engineering judgment. Generally, provided the constitutive equations are adequate, the results for both methods are identical.

ATENA currently uses the Updated Lagrangian formulation (which is described later in this chapter) and supports the highest, i.e., 3rd level of nonlinear behavior. Soon, it should also support Total Lagrangian formulation.

1.2 General Problem Formulation

A general analysis of a structure usually consists of the application of many small load increments. At each of those increments, an iterative solution procedure has to be executed to obtain a structural response at the end of the increment. Hence, denoting the start and end of the load increment by t and $t+\Delta t$, at each step, we know the structural state at the time t (from the previous steps) and solve for the state at the time $t+\Delta t$. This procedure is repeated as many times as necessary to reach the final (total) level of loading.

This process is depicted in Fig. 1-1. At the time $t=0$ the volume of the structure is ${}^{0}V$, the surface area is ${}^{0}S$, and any arbitrary point M has coordinates ${}^{0}X_1, {}^{0}X_2, {}^{0}X_3$. Similarly, at the time t the same structure has a volume ${}^{t}V$, surface area ${}^{t}S$, and coordinates of the point M are ${}^{t}X_1, {}^{t}X_2, {}^{t}X_3$. A similar definition applies for the time $t+\Delta t$ by replacing index t by $t+\Delta t$.

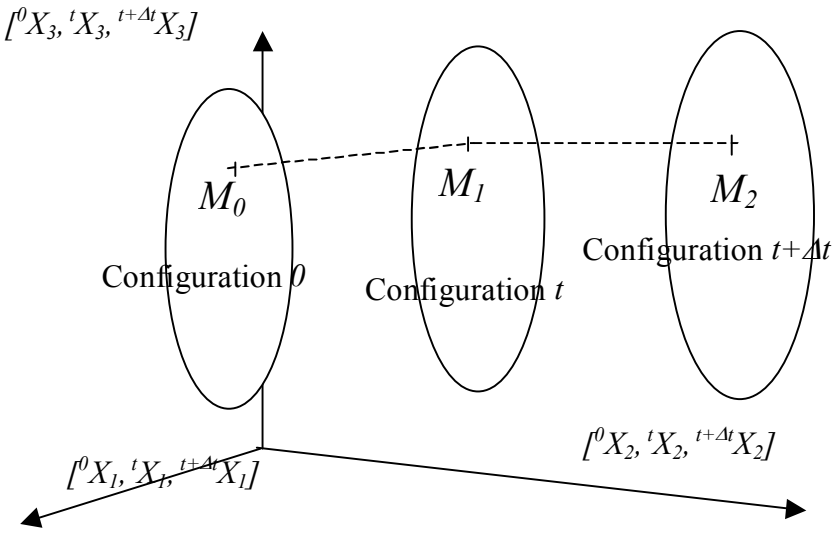


Fig. 1-1 The movement of body of structure in Cartesian coordinate system.

For the derivations of nonlinear equations, it is important to use clear and simple notations. The same system of notation will be used throughout this document:

Displacements u are defined in a similar manner to that adopted for coordinates, hence ${}^{t}u_i$ is the i-th element of the displacement vector at the time t,

$u_i = {}^{t+\Delta t}X_i - {}^{t}X_i$ is i-th element of the vector of displacement increments at the time t,

The left superscript denotes the time corresponding to the value of the entity, the left subscript denotes the configuration with respect to which the value is measured, and subscripts on the right identify the relationships to the coordinate axis. Thus, for example ${}^{t+\Delta t}_{0}\tau_{ij}$ denotes element i, j of stress tensor τ at the time $t+\Delta t$ with respect to the original (undeformed) configuration.

For derivatives, the abbreviated notation will be used, i.e., all right subscripts that appear after a comma declare derivatives. For example:

$$ {}^{t+\Delta t}_{0}u_{i,j} = \frac{\partial}{\partial X_j}\,{}^{t+\Delta t}u_i \tag{1.2}$$

The general governing equations can be derived in the form of a set of partial differential equations (for example, using the displacement method), or an energy approach can be used. The final results are the same.

One of the most general methods of establishing the governing equations is to apply the principle of virtual work. There are three basic variants of this:

The principle of virtual displacements,

The principle of virtual forces,

The Clapeyron divergent theorem.

Using the virtual work theorems, it is possible to derive several different variation principles (Lagrange principle, Clapeyron principle, Hellinger-Reissner principle, Hu-Washizu principle etc.). There are popular especially in linear analysis. They can be used to establish equilibrium equations, to study possible deformation modes in finite element discretization etc. Unfortunately, in the nonlinear analysis, they do not always work.

In this document, all the following derivations will be presented in their displacement forms, and consequently, the principle of virtual displacements will be used throughout.

The following section deals with the definition of the stress and strain tensors, which are usually used in nonlinear analysis. All of them are symmetric.

1.3 Stress Tensors

1.3.1 Cauchy Stress Tensor

This tensor is well known from linear mechanics. It expresses the forces that act on infinitesimally small areas of the deformed body at time t. Sometimes, these are also called an "engineering" stress. The Cauchy stress tensor is the main entity for checking ultimate stress values in materials. In the following text, it will be denoted by τ. It is energetically conjugated with an Engineering strain tensor described later.

1.3.2 2nd Piola-Kirchhoff Stress Tensor

The 2nd Piola-Kirchhoff tensor is a fictitious entity, having no physical representation of it as in the case of the Cauchy tensor. It expresses the forces, which act on infinitesimal areas of the body in the undeformed configuration. Hence it relates forces to the shape of the structure, which no longer exists.

The mathematical definition is given by:

$$ {}^{t}_{0}S_{ij} = \frac{{}^{0}\rho}{{}^{t}\rho} \; {}^{0}_{t}X_{i,m} \; {}^{t}\tau_{mn} \; {}^{0}_{t}X_{j,n} \tag{1.3} $$

where

$\frac{{}^{0}\rho}{{}^{t}\rho}$ is the ratio of density of the material at time 0 and t,

${}^{t}\tau_{mn}$ is the Cauchy stress tensor at time t,

${}^{0}_{t}X_{i,m}$ is the derivative of coordinates, ref. (1.5).

Using inverse transformation, we can express Cauchy stress tensor in terms of the 2nd Piola-Kirchhoff stress tensor, i.e.:

$$ {}^{t}\tau_{mn} = \frac{{}^{t}\rho}{{}^{0}\rho} \; {}^{t}_{0}X_{m,i} \; {}^{t}_{0}S_{mn} \; {}^{t}_{0}X_{n,j} \tag{1.4} $$

The elements ${}^{0}_{t}X_{i,m}$ are usually collected in the so-called Deformation gradient matrix:

$$ {}^{t}_{0}X = \left({}_{0}\underline{\nabla} \; {}^{t}\underline{X}^{T} \right)^{T} \tag{1.5} $$

where:

$$ {}_{0}\underline{\nabla}^{T} = \left[\frac{\partial}{\partial {}^{0}X_{1}}, \frac{\partial}{\partial {}^{0}X_{2}}, \frac{\partial}{\partial {}^{0}X_{3}} \right]^{T} $$

$$ {}^{t}\underline{X}^{T} = \left[{}^{t}X_{1}, {}^{t}X_{2}, {}^{t}X_{3} \right] $$

The ratio $\frac{{}^{0}\rho}{{}^{t}\rho}$ can be computed using:

$$ {}^{0}\rho = {}^{t}\rho \det({}^{t}_{0}X) \tag{1.6}$$

Expression (1.6) is based on the assumption that the weight of an infinitesimal particle is constant during the loading process.

Some important properties can be deduced from the definition of 2nd Piola-Kirchhoff tensor (1.3) :

at time 0, i.e., the undeformed configuration, there is no distinction between 2nd Piola-Kirchhoff and Cauchy stress tensors because ${}^{0}_{0}X = E$, i.e., unity matrix and the density ratio $\frac{{}^{0}\rho}{{}^{t}\rho} = 1.$,

2nd Piola-Kirchhoff tensor is an objective entity in the sense that it is independent of any movement of the body provided the loading conditions are frozen. This is a very important property. The Cauchy stress tensor does not satisfy this because it is sensitive to the rotation of the body. It is energetically conjugated with the Green-Lagrange tensor described later.

They're some other stress tensor commonly used for nonlinear structural analysis, e.g., Jaumann stress rate tensor (describes stress rate rather than its final values) etc.; however, they are not used in ATENA and therefore not described in this document.

1.4 Strain Tensors

1.4.1 Engineering Strain

It is the most commonly used strain tensor, comprising strains that are called Engineering strains. Its main importance is that it is used in linear mechanics as a counterpart to the Cauchy stress tensor.

$$ {}_{t}e_{mn} = \frac{1}{2}\left(\frac{\partial u_m}{\partial {}^{t}X_n} + \frac{\partial u_n}{\partial {}^{t}X_m} \right) \tag{1.7}$$

1.4.2 Green-Lagrange Strain

This is the energy conjugate of the 2nd Piola-Kirchhoff tensor and its properties are similar (i.e., objective etc.). It is defined as:

$$ {}^{t}_{0}\varepsilon_{ij} = \frac{1}{2}\left({}^{t}_{0}u_{i,j} + {}^{t}_{0}u_{j,i} + {}^{t}_{0}u_{k,i}\,{}^{t}_{0}u_{k,j} \right) \tag{1.8}$$

If we calculate the length of an infinitesimal fibber prior to and after deformation in the original coordinates, we get exactly the terms of the Green-Lagrange tensor.

The following equation gives a relation between variation of Green-Lagrange and Engineering strain tensors:

$$ \partial\left({}^{t}_{0}\varepsilon_{ij} \right) = \frac{\partial {}^{t}X_m}{\partial {}^{0}X_i}\,\frac{\partial {}^{t}X_n}{\partial {}^{0}X_j}\,\partial\left({}_{t}e_{mn} \right) \tag{1.9}$$

These are the strain tensors used in ATENA. From the other strain tensors commonly used in the nonlinear analysis we can mention Almansi strain tensor, co-rotated logarithmic strain, strain rate tensor etc.

1.5 Constitutive Tensor

Although the whole chapter later in this document is dedicated to the problem of constitutive equations and to material failure criteria, assume for the moment that stress-strain relation can be written in the following form:

$$ {}^{t}_{0}S_{ij} = {}^{t}_{0}C_{ijrs} \; {}^{t}_{0}\varepsilon_{rs} \tag{1.10} $$

where ${}^{t}_{0}C_{ijrs}$ is the constitutive tensor.

This form is applicable for linear materials, or in its incremental form, it can also be used for nonlinear materials. The following important relations apply for transformation from coordinates to time 0 to coordinates at the time t:

$$ {}^{t}_{t}C_{mnpq} = \frac{{}^{t}\rho}{{}^{0}\rho} \, {}^{t}_{0}x_{m,i} \; {}^{t}_{0}x_{n,j} \; {}^{t}_{0}C_{ijrs} \; {}^{t}_{0}x_{p,r} \; {}^{t}_{0}x_{q,s} \tag{1.11} $$

or in the other direction

$$ {}^{t}_{0}C_{ijrs} = \frac{{}^{0}\rho}{{}^{t}\rho} \, {}^{0}_{t}x_{i,m} \; {}^{0}_{t}x_{j,n} \; {}^{t}_{t}C_{mnpq} \; {}^{0}_{t}x_{r,p} \; {}^{0}_{t}x_{s,q} \tag{1.12} $$

Using constitutive tensor (1.11) and Almansi strains ${}^{t}_{t}\varepsilon$, we can write for Cauchy stresses (with respect to coordinates at time t):

$$ {}^{t}\tau_{ij} = {}^{t}_{t}C_{ijrs} \; {}^{t}_{t}\varepsilon_{rs} \tag{1.13} $$

Almansi strains are defined (related to Green-Lagrange strains ${}^{t}_{0}\varepsilon_{ij}$ by

$$ {}^{t}_{t}\varepsilon_{mn} = {}^{0}_{t}x_{i,m} \; {}^{0}_{t}x_{j,n} \; {}^{t}_{0}\varepsilon_{ij} \tag{1.14} $$

or can be calculated directly:

$$ {}^{t}_{t}\varepsilon_{ij} = \frac{1}{2}\left({}^{t}_{t}u_{i,j} + {}^{t}_{t}u_{j,i} - {}^{t}_{t}u_{k,i} \; {}^{t}_{t}u_{k,j} \right) \tag{1.15} $$

The equation (1.13) is equivalent to the equation (1.10) that was written for the original configuration of the structure. It is very important to know, with respect to which coordinate system the stress, strain, and constitutive tensors are defined, as the actual value can significantly differ. ATENA currently assumes that all these tensors are defined at coordinates at time t.

1.6 The Principle of Virtual Displacements

This section presents how the principle of virtual displacement can be applied to the analysis of a structure. For completeness, both the Lagrangian Total and Updated formulations will be discussed. In all derivations, it is assumed that the response of the structure up to time t is known. Now, at the time $t + \Delta t$ we apply load increment and using the principle of virtual displacement will solve for the state of the structure at $t + \Delta t$.

Virtual work of the structure yields the following. For Total formulation:

$$\int_{{}^0V} \left({}_0^{t+dt}S_{ij}\, \delta \left({}_0^{t+dt}\varepsilon_{ij} \right) dV \right) = {}_0^{t+dt}R \tag{1.16}$$

for Updated formulation:

$$\int_{{}^tV} \left({}_t^{t+dt}S_{ij}\, \delta \left({}_t^{t+dt}\varepsilon_{ij} \right) dV \right) = {}_t^{t+dt}R \tag{1.17}$$

where ${}^0V$, ${}^tV$ denotes the structure volume corresponding to time 0 and t ${}^{t+dt}R$ is the total virtual work of the external forces. The symbol δ denotes variation of the entity. Since energy must be invariant with respect to the reference coordinate system (1.16) and (1.17) must lead to identical results.

Substituting expressions for strain and stress tensors, the final governing equation for structure can be derived. They are summarized in (1.18) through (1.29). Note that the relationships are expressed with respect to configurations at an arbitrary time t and an iteration $^{(i)}$. Typically, the time t may by 0, in which case we have Total Lagrangian formulation or $t+\Delta t(i-1)$, in which case, we have Updated Lagrangian formulation, where some terms can be omitted. ATENA also supports "semi" Updated Lagrangian formulation when t conforms to time at the beginning of time increment, i.e., the beginning of load step. The following table compares the above-mentioned formulations:

Table 1.6-1 Comparison of different Lagrangian formulation.

Lagrangian formulation	Transform each iteration		Transform each load increment		Transform stress and strain for output	Calculate ${}^{t+\Delta t(i-1)}_{t}u_{i,j}$ for ${}_te_{ij}$
	IP state variables	Material properties	IP state variables	Material properties		
Total	No	No	No	No	Yes	Yes
Updated	Yes	Yes	Yes	Yes	No	No
"Semi"-Updated	No	No	Yes	Yes	No	Yes

Governing equations:

$$\int_{{}^tV} {}_t^{t+\Delta t}S_{ij}^{(i)}\, \partial \left({}_t^{t+\Delta t}\varepsilon_{ij}^{(i)} \right) {}^tdV = {}^{t+\Delta t}R \tag{1.18}$$

where 2nd Piola-Kirchhoff stress and Green Lagrange strain tensor are:

$${}_t^{t+\Delta t}S_{ij}^{(i)} = \frac{{}^t\rho}{{}^{t+\Delta t}\rho}\, {}_{t+\Delta t}^{t}x_{i,m}^{(i)}\; {}_t^{t+\Delta t}\tau_{mn}^{(i)}\; {}_{t+\Delta t}^{t}x_{j,n}^{(i)} \tag{1.19}$$

$$\delta\, {}_t^{t+\Delta t}\varepsilon_{ij}^{(i)} = \delta \frac{1}{2}\left({}_t^{t+\Delta t}u_{i,j}^{(i)} + {}_t^{t+\Delta t}u_{i,j}^{(i)} + {}_t^{t+\Delta t}u_{k,i}^{(i)}\; {}_t^{t+\Delta t}u_{k,j}^{(i)} \right) \tag{1.20}$$

The stress and strain increments:

$${}_t^{t+\Delta t}S_{ij}^{(i)} = {}_t^{t+\Delta t}S_{ij}^{(i-1)} + {}_t\, S_{ij}^{(i)} \tag{1.21}$$

$$
\begin{aligned}
{}^{t+\Delta t}_{t}\varepsilon_{ij}^{(i)} &= {}^{t+\Delta t}_{t}\varepsilon_{ij}^{(i-1)} + {}_{t}\varepsilon_{ij}^{(i)} \\
{}_{t}\varepsilon_{ij}^{(i)} &= {}_{t}e_{ij}^{(i)} + {}_{t}\eta_{ij}^{(i)}
\end{aligned}
\tag{1.22}
$$

where linear part of the strain increment is calculated by:

$$
{}_{t}e_{ij}^{(i)} = \frac{1}{2}\left({}_{t}u_{i,j}^{(i)} + {}_{t}u_{i,j}^{(i)} + {}^{t+\Delta t}_{t}u_{k,i}^{(i-1)}\ {}_{t}u_{k,j}^{(i)} + {}^{t+\Delta t}_{t}u_{k,j}^{(i-1)}\ {}_{t}u_{k,i}^{(i)} \right) \tag{1.23}
$$

and nonlinear part by:

$$
{}_{t}\eta_{ij}^{(i)} = \frac{1}{2}\left({}_{t}u_{k,i}^{(i)}\ {}_{t}u_{k,j}^{(i)} \right) \tag{1.24}
$$

Using constitutive equations in form:

$$
{}^{t+\Delta t}_{t}S_{ij}^{(i)} = {}_{t}C_{ijrs}\ {}_{t}\varepsilon_{rs}^{(i)} \tag{1.25}
$$

where ${}_{t}C_{ijrs}^{(i)}$ is tangent material tensors and noting that $\delta\left({}^{t+\Delta t}_{t}\varepsilon_{ij}^{(i)} \right) = \delta\left({}_{t}\varepsilon_{ij}^{(i)} \right)$, an incremental form of (1.18) can be derived:

$$
\int_{{}^{t}V} {}_{t}C_{ijrs}\ {}_{t}\left({}_{t}e_{ij}^{(i)} + {}_{t}\eta_{ij}^{(i)} \right)\ \delta\left({}_{t}e_{ij}^{(i)} + {}_{t}\eta_{ij}^{(i)} \right)^{t} dV + \int_{{}^{t}V} {}^{t+\Delta t}_{t}S_{ij}^{(i)}\ \delta\left({}_{t}e_{ij}^{(i)} + {}_{t}\eta_{ij}^{(i)} \right)^{t} dV = {}^{t+\Delta t}\ R \tag{1.26}
$$

After linearisation, i.e., neglecting 2nd order terms in (1.26):

$$
\int_{{}^{t}V} {}_{t}C_{ijrs}\ {}_{t}\left({}_{t}e_{ij}^{(i)} + {}_{t}\eta_{ij}^{(i)} \right)\ \delta\left({}_{t}e_{ij}^{(i)} + {}_{t}\eta_{ij}^{(i)} \right)^{t} dV \approx \int_{{}^{t}V} {}_{t}C_{ijrs\,t}\ {}_{t}e_{ij}^{(i)}\ \ \delta\left({}_{t}e_{ij}^{(i)} \right)^{t} dV \tag{1.27}
$$

we arrive to the final form of the governing equations:

$$
\begin{aligned}
&\int_{{}^{t}V} {}_{t}C_{ijrs}\ {}_{t}e_{rs}^{(i)}\ \delta\left({}_{t}e_{ij}^{(i)} \right)^{t} dV + \int_{{}^{t}V} {}^{t+\Delta t}_{t}S_{ij}^{(i-1)}\ \delta\left({}_{t}\eta_{ij}^{(i)} \right)^{t} dV = \\
&{}^{t+\Delta t}R - \int_{{}^{t}V} {}^{t+\Delta t}_{t}S_{ij}^{(i-1)}\ \delta\left({}_{t}e_{ij}^{(i)} \right)^{t} dV
\end{aligned}
\tag{1.28}
$$

Note that the term $\delta\left({}_{t}e_{ij}^{(i)} \right) = \delta\left({}_{t}e_{ij} \right)$ is constant, i.e., independent of ${}_{t}u_{i}^{(i)}$, hence it is on RHS of (1.28).

1.7 The Work Done by the External Forces

So far only the incremental virtual internal work has been considered. This work has to be balanced by the work done by the external forces. It is calculated as follows:

$$
{}^{t+\Delta t}R = \int_{{}^{t}V} {}^{t+\Delta t}fb_{i}\ \delta\left({}_{t}u^{i} \right) dV + \int_{{}^{t}S} {}^{t+\Delta t}fs_{i}\ \delta\left({}_{t}u^{i} \right) dS + \int_{{}^{t}V} {}^{t}\rho \frac{\partial^{2}\ {}^{t+\Delta t}_{t}u_{i}^{(i-1)}}{\partial t^{2}} dV \tag{1.29}
$$

where fb_i and fs_i are body and surface forces, ${}^{t}S$ and ${}^{t}V$ denotes integration with respect to the surface with the prescribed boundary forces and volume of the structure (at the time and t).

The 1st integral in (1.29) accounts for external work on a surface (e.g., external forces), the second one for work done by body forces (e.g., weight), and the last one accounts for work done by inertia forces, which are applicable only for dynamic analysis problems).

At this point, all the relationships for incremental analysis have been presented. In order to proceed further, the problem must be discretized and solved by iterations (described in Chapter Solution of Nonlinear Equations).

1.8 Problem Discretisation Using Finite Element Method

Spatial discretization consists of discretizing the primary variable, (i.e., deformation in case of ATENA) over the domain of the structure. It is done in ATENA by the Finite Element Method. The domain is decomposed into many finite elements, and at each of these elements, the deformation field is approximated by

$$ {}^{t}u_i = h_j \ {}^{t}u_i^j \tag{1.30}$$

where
j is the index for finite element node, $j = 1...n$,
n is the number of element nodes,
h_j are interpolation function usually grouped in matrix $H_j = \left[h_1(r,s,t), h_2(r,s,t).....h_n(r,s,t)\right]$,
r,s,t are the local element coordinates.

The interpolation functions h_j are usually created in the way that $h_j = 1$ at the node j and $h_j = 0$ at any other element nodes.

Combining (1.30) and equation for strain definition (1.8) it can be derived:

$$ {}^{t+\Delta t}_{t}\underline{\varepsilon}^{(i)} = \left({}_{t}\boldsymbol{B}_{L0} + {}^{t+\Delta t}_{t}\boldsymbol{B}_{L1}^{(i-1)} + {}^{t+\Delta t}_{t}\boldsymbol{B}_{NL}^{(i-1)} \right) {}^{t+\Delta t}\underline{U}^{(i)} \tag{1.31}$$

where
${}^{t+\Delta t}_{t}\underline{\varepsilon}^{(i)}$ is the vector of Green-Lagrange strains,
${}^{t+\Delta t}\underline{U}^{(i)}$ is the vector of displacements,
${}_{t}\boldsymbol{B}_{L0}$, ${}^{t+\Delta t}_{t}\boldsymbol{B}_{L1}^{(i-1)}$, ${}^{t+\Delta t}_{t}\boldsymbol{B}_{NL}^{(i-1)}$ are linear strain-displacements transformation matrices (the 1st two of them) and nonlinear strain-displacements transformation matrix (the last one).

A similar equation can also be written for stress tensor.

$$ {}^{t+\Delta t}_{t}\underline{S}^{(i)} = {}^{t+\Delta t}_{t}\boldsymbol{C}^{(i)} \ {}^{t+\Delta t}_{t}\underline{\varepsilon}^{(i)} \tag{1.32}$$

where:
${}^{t+\Delta t}_{t}\underline{S}^{(i)}$ is vector of 2nd Piola-Kirchhoff stress tensor and
${}^{t+\Delta t}_{t}\boldsymbol{C}^{(i)}$ is incremental stress-strain material properties matrix.

Applying the above discretization for each finite element of the structure and assembling the results, the continuum based governing equations in (1.28) can be re-written in the following form:

$$ {}^{t}\boldsymbol{M} \frac{\partial}{\partial t^2} {}^{t+\Delta t}\underline{U}^{(i)} + ({}_{t}\boldsymbol{K}_L + {}^{t+\Delta t}_{t}\boldsymbol{K}_{NL}^{(i-1)}) \ \Delta {}^{t+\Delta t}\underline{U}^{(i)} = {}^{t+\Delta t}\underline{R} - {}^{t+\Delta t}\underline{F}^{(i-1)} \tag{1.33}$$

where

${}_{t}\boldsymbol{K}_L$ is the linear strain incremental stiffness matrix,

${}^{t+\Delta t}_{t}\boldsymbol{K}_{NL}^{(i-1)}$ is the nonlinear strain incremental stiffness matrix,

${}^{t}\boldsymbol{M}$ is the structural mass matrix,

$\Delta^{t+\Delta t}\underline{U}^{(i)}$ is the vector of nodal point displacements increments at the time $t+\Delta t$, iteration i;

${}^{t+\Delta t}\dfrac{\partial}{\partial t^2}\left({}^{t+\Delta t}\underline{U}^{(i)}\right)$ is the vector of nodal accelerations,

${}^{t+\Delta t}\underline{R}$, ${}^{t+\Delta t}\underline{F}^{(i-1)}$ is the vector of applied external forces and internal forces,

$^{(i)}$, $^{(i-1)}$ superscripts indicate iteration numbers.

Note that (1.33) also contains inertial term needed only for dynamic analysis. Finite element matrices in (1.33) and corresponding analytical expressions are summarized:

$$ {}_t\boldsymbol{K}_L\,\Delta\underline{U}^{(i)} = \left(\int_{{}^tV} {}_t\boldsymbol{B}_L^T\ {}_tC\ {}_t\boldsymbol{B}_L\,dV\right)\Delta\underline{U}^{(i)} \approx \int_{{}^tV} {}_tC_{ijrs}\,{}_te_{rs}^{(i)}\,\delta\left({}_te_{ij}^{(i)}\right)^t dV $$

$$ {}^{t+\Delta t}_{\ t}\boldsymbol{K}_{NL}^{(i-1)}\,\Delta\underline{U}^{(i)} = \left(\int_{{}^tV} {}^{t+\Delta t}_{\ t}\boldsymbol{B}_{NL}^{(i-1)T}\ {}^{t+\Delta t}_{\ t}\boldsymbol{S}_{ij}^{(i-1)}\ {}^{t+\Delta t}_{\ t}\boldsymbol{B}_{NL}^{(i-1)}\,dV\right)\Delta\underline{U}^{(i)} \approx \int_{{}^tV} {}^{t+\Delta t}_{\ t}S_{ij}^{(i-1)}\,\delta\left({}_t\eta_{ij}^{(i)}\right)^t dV $$

$$ {}^{t+\Delta t}\underline{F}^{(i-1)} = \int_{{}^tV} {}^{t+\Delta t}_{\ t}\boldsymbol{S}_{ij}^{(i-1)}\,dV \approx \int_{{}^tV} {}^{t+\Delta t}_{\ t}S_{ij}^{(i-1)}\,\delta\left({}_te_{ij}^{(i)}\right)^t dV \qquad (1.34) $$

$$ {}^{t+\Delta t}\underline{R} = \int_{{}^tA} \boldsymbol{H}^{\boldsymbol{T}}\ {}^{t+\Delta t}_{\ t}\underline{f}^{A}\,dV\,dA + \int_{{}^tV} \boldsymbol{H}^{\boldsymbol{T}}\ {}^{t+\Delta t}_{\ t}\underline{f}^{B}\,dV \approx {}^{t+\Delta t}R $$

$$ {}^t\boldsymbol{M}\,\frac{\partial}{\partial t^2}\,{}^{t+\Delta t}\underline{U}^{(i)} = \left(\int_{{}^tV} \boldsymbol{H}^{\boldsymbol{T}}\ {}^t\rho\ \boldsymbol{H}\,dV\right)\frac{\partial}{\partial t^2}\,{}^{t+\Delta t}\underline{U}^{(i)} \approx \int_{{}^tV} \frac{\partial^{t+\Delta t}\boldsymbol{u}_i^{(i)}}{\partial t^2}\ {}^t\rho\,\delta\left(\frac{\partial^{t+\Delta t}\boldsymbol{u}_i^{(i)}}{\partial t^2}\right)dV $$

1.9 Stress and Strain Smoothing

All derivations and solution procedures in ATENA software are based on the deformational form of the finite element method. Any structure is solved using the weak (or integral) form of equilibrium equations. The whole structure is divided into many finite elements, and displacement $\underline{u}$ at each particular element (at any location) is approximated by approximation functions h_i and element displacements $\underline{u}^i$ as follows: $\underline{u} = \sum_i h_i \underline{u}^i$, ($i$ is index of an element node). It is important to note that in order not to lose any internal energy of the structure, the displacements over the whole structure must be continuous. The continuity within finite elements is trivial. The use of continuous approximation functions h_j ensures this requirement. A bit more complicated situation is on boundaries between adjacent elements; however, if the adjacent elements are of the same type, their displacements are also continuous. Note that there exist are some techniques that alleviate the continuity requirement, but in ATENA they are not used.

Unlike displacements, stress and strain fields are typically discontinuous. Moreover, a structure is investigated within so-called material (or integral) points, which are points located somewhere within each element. Their position is derived from the requirement to minimize the approximation error. In other words, the standard finite element method provides stress and strain values only at those material points, and these values must be later somehow extrapolated

into element nodal points. Often, some sort of smoothing is required in order to remove the mentioned stress and strain discontinuity. This section describes how this goal is done in ATENA.

There are two steps in the process of stress and strain smoothing: 1/ extrapolation of stress and strain from material points to element nodes and 2/ averaging of stress in global node. The whole technique is described briefly. All details and derivations can be found e.g. (ZIENKIEWICZ, TAYLOR 1989) and ČERVENKA et. al. 1993.

1.9.1 Extrapolation of Stress and Strain to Element Nodes

The extrapolation is done as follows (for each component of structural stress $\underline{\sigma}$ and strain $\underline{\varepsilon}$).

Let us define a vector of stresses $\underline{\tilde{\sigma}}_{xx}$ at element nodes i such as $\underline{\tilde{\sigma}}_{xx} = \left\{\tilde{\sigma}_{xx,1}, \tilde{\sigma}_{xx,2}, ...\tilde{\sigma}_{xx,n}\right\}^T$, where the 2nd index indicates element node number. Let us also define a vector $\underline{P}_{xx} = \left\{P_{xx,1}, P_{xx,2}, ...P_{xx,n}\right\}^T$, whose component are calculated

$$P_{xx,i} = \int_{\Omega_e} h_i \, \sigma_{xx} \, d\Omega_e \tag{1.35}$$

The nodal value $\underline{\tilde{\sigma}}_{xx}$ (with values of σ_{xx} at nodes i=1..n) is then calculated as follows:

$$\underline{\tilde{\sigma}}_{xx} = \left[M\right]^{inv} \underline{P}_{xx} \tag{1.36}$$

where:

$$M_{ij} = \int_{\Omega_e} h_i \, h_j \; d\Omega_e \tag{1.37}$$

In the above σ_{xx} is an extrapolated field of the stress σ_{xx} calculated by FEM. It is typically discontinuous. n is the number of element nods, Ω_e is the volume of the investigated finite element. The same strategy is also used for the remaining stress and strain components.

This smoothing technique is called variational as it is base on averaging energy over the element.

In addition, ATENA also supports another way of extrapolating vales from integration points to element nodes. In this case, (1.37) is assumed to be a "lumped" diagonal matrix in order to eliminate the need for solving a system of linear equations. The process of lumping is characterized as follows:

$$M_{ij} = \int_{\Omega_e} h_i \sum_{k=1,n} h_k \, \delta_{ij} \; d\Omega_e \tag{1.38}$$

As most element space approximations satisfy $\sum_{k=1,n} h_k = 1$, the above equation is simplified to:

$$M_{ij} = \int_{\Omega_e} h_i \, \delta_{ij} \, d\Omega \tag{1.39}$$

where δ_{ij} is Kronecker delta. This "lumped" formulation ATENA uses by default.

The above values are output as nodal element stress/strain values. It follows to calculate averaged stress/stain value $\underline{\hat{\sigma}}_i = \left\{\hat{\sigma}_{xx}, \hat{\sigma}_{yy}, ...\hat{\sigma}_{xz}\right\}_i$ in a global node i that is participated by all elements k with an incidence at the global node i.

$$\underline{\hat{\sigma}}_i = \frac{\sum_k \underline{\tilde{\sigma}}_i \, \Omega_{e_k}}{\sum_k \Omega_{e_k}} \tag{1.40}$$

where is the vector of stresses $\underline{\tilde{\sigma}}_i = \left\{ \tilde{\sigma}_{xx}, \tilde{\sigma}_{yy}, ... \tilde{\sigma}_{xz} \right\}_i$ at a node i , Ω_{e_k} is the volume of the element k that has the incidence of global node i. It should be noted that in ATENA, the same extrapolation techniques are used for other integration point quantities as well such as: fracturing strains, plastic strains and others.

1.10 Simple, Complex Supports and Master-Slave Boundary Conditions.

Simple support and complex support boundary conditions represent boundary conditions of Dirichlet types, i.e., boundary conditions that prescribe displacements. On the other hand, Simple load boundary conditions are an example of von Neumann type boundary conditions when forces are prescribed.

Let $\mathbf{K}$ is structural stiffness matrix, $\underline{u}$ is the vector of nodal displacements, and $\underline{R}$ is a vector of nodal forces. Further, let $\underline{u}$ is subdivided into the vector of free degrees of freedom $\underline{u}_N$ (with von Neumann boundary conditions) and constrained degrees of freedom $\underline{u}_D$ (with Dirichlet boundary conditions):

$$\underline{u} = \begin{bmatrix} \underline{u}_N \\ \underline{u}_D \end{bmatrix} \tag{1.41}$$

The problem governing equations can then be written:

$$\begin{bmatrix} \mathbf{K}_{NN} & \mathbf{K}_{ND} \\ \mathbf{K}_{DN} & \mathbf{K}_{DD} \end{bmatrix} \begin{bmatrix} \underline{u}_N \\ \underline{u}_D \end{bmatrix} = \begin{bmatrix} \underline{R}_N \\ \underline{R}_D \end{bmatrix} \tag{1.42}$$

ATENA software supports that any constrained degree of freedom can be a linear combination of other degrees of freedom plus some constant term:

$$\underline{u}_D^i = \underline{u}_D^{i,0} + \sum_k \alpha_k \, \underline{u}_N^k \tag{1.43}$$

where $\underline{u}_D^{i,0}$ is the constant term and α_k are coefficients of the linear combination. Of course, the equation (1.43) can also include the term $\sum_l \alpha_l \underline{u}_D^l$; however, it is transformed into the constant term.

The free degrees of freedom are then solved by

$$\underline{u}_N = \left(\mathbf{K}_{NN} \right)^{-1} \left(\underline{R}_N - \mathbf{K}_{ND} \underline{R}_D \right) \tag{1.44}$$

and the dependent $\underline{R}_D$ is solved by

$$\underline{R}_D = \mathbf{K}_{DN} \underline{u}_N + \mathbf{K}_{DD} \underline{u}_D \tag{1.45}$$

The ATENA **simple support** boundary conditions mean that the boundary conditions use only constant terms are $\underline{u}_D^{i,0}$, (i.e. $\alpha_k = 0$). The **complex support** boundary conditions use the full form of (1.43).

The boundary conditions as described above allow to specify for one degree of freedom either Dirichlet, or von Neumann boundary condition, but not both of them at the same time. It comes from the nature of the finite element method. However, ATENA can also deal with this case of more complex boundary conditions by introducing Lagrange multipliers. The derivation of the theory behind this kind of boundary conditions is beyond the scope of this manual. Details can be found elsewhere, e.g., in (Bathe 1982). To apply this type of boundary conditions in ATENA, specify for those degrees of freedom both simple load and complex support boundary condition, the latter one with the keyword "RELAX" keyword in its definition.

A useful feature of ATENA is that at any time, it stores in RAM only $\mathbf{K}_{NN}$ and all the elimination with the remaining blocks of **K** is done at element level at the process of assembling the structural stiffness matrix.

A special type of complex boundary conditions of the Dirichlet type are so-called master-slave boundary conditions. Such a boundary condition specifies that all (available) degrees of one finite node (i.e., slave node) are equal to degrees of freedom of another node (i.e., master node). If more master nodes are specified, then these master nodes are assumed to form a finite element and degrees of freedom of the slave node are assumed to be a node within that element. Its (slave) degrees of freedom are approximated by element nodal (i.e., master) degrees of freedom in the same way as displacements approximation within a finite element. The coefficients α_k in (1.43) are thus calculated automatically. This type of boundary condition is used for example, for fixing discrete reinforcement bars to the surrounding solid element.

1.11 References

BATHE, K.J. (1982), Finite Element Procedures In Engineering Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632, ISBN 0-13-317305-4.

ČERVENKA, J., KEATING, S.C., AND FELIPPA, C.A. (1993), "Comparison of strain recovery techniques for the mixed iterative method", Communications in Numerical Methods in Engineering, Vol. 9, 925-932.

ZIENKIEWICZ, O.C., TAYLOR, R.L., (1989), The Finite Element Method, Volume 1, McGraw-Hill Book Company, ISBN 0-07-084174-8.

2 CONSTITUTIVE MODELS

2.1 Constitutive Model SBETA (CCSbetaMaterial)

2.1.1 Basic Assumptions

2.1.1.1 Stress, Strain, Material Stiffness

The formulation of constitutive relations is considered in the plane stress state. A smeared approach is used to model the material properties, such as cracks or distributed reinforcement. This means that material properties defined for a material point are valid within a certain material volume, which is in this case associated with the entire finite element. The constitutive model is based on the stiffness and is described by the equation of equilibrium in a material point:

$$\mathbf{s} = \mathbf{D}\,\mathbf{e},\ \mathbf{s} = \left\{\sigma_x, \sigma_y, \tau_{xy}\right\}^T,\ \mathbf{e} = \left\{\varepsilon_x, \varepsilon_y, \gamma_{xy}\right\}^T \tag{2.1}$$

where **s, D** and **e** are a stress vector, a material stiffness matrix and a strain vector, respectively. The stress and strain vectors are composed of the stress components of the plane stress state $\sigma_x, \sigma_y, \tau_{xy}$, Fig. 2-1, and the strain components $\varepsilon_x, \varepsilon_y, \gamma_{xy}$, Fig. 2-2, where γ_{xy} is the engineering shear strain. The strains are common for all materials. The stress vector **s** and the material matrix **D** can be decomposed into the material components due to concrete and reinforcement as:

$$\mathbf{s} = \mathbf{s}_c + \mathbf{s}_s,\ \mathbf{D} = \mathbf{D}_c + \mathbf{D}_s \tag{2.2}$$

The stress vector **s** and both component stress vectors $\mathbf{s}_c, \mathbf{s}_s$ are related to the total cross section area. The concrete stress $\mathbf{s}_c$ is acting on the material area of concrete $\mathbf{A}_c$, which is approximately set equal to the cross section of the composite material $\mathbf{A}_c \approx \mathbf{A}$ (the area of concrete occupied by reinforcement is not subtracted).

The matrix **D** has a form of the Hooke's law for either isotropic or orthotropic material, as will be shown in Section 2.1.11.

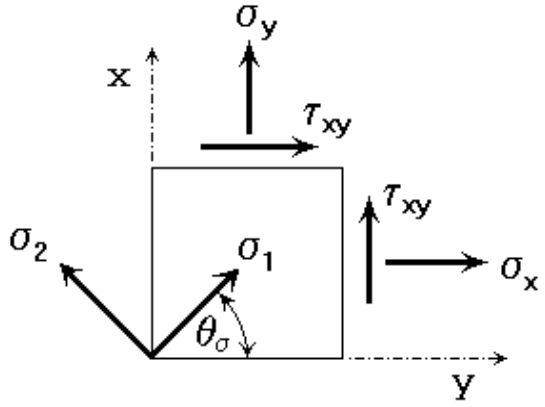


Fig. 2-1 Components of plane stress state.

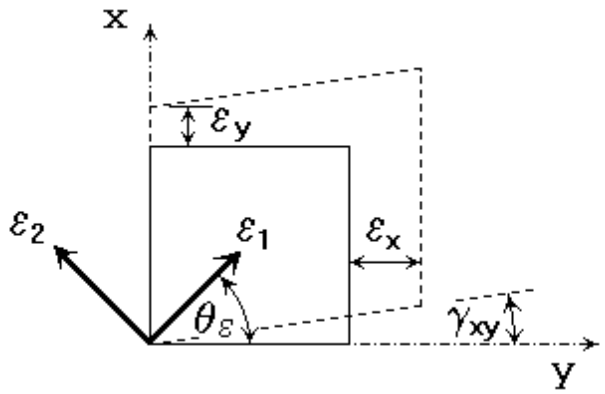


Fig. 2-2 Components of strain state.

The reinforcement stress vector $\mathbf{s}_s$ is the sum of stresses of all the smeared reinforcement components:

$$\mathbf{s}_s = \sum_{i=1}^{n} \mathbf{s}_{si} \tag{2.3}$$

where n is the number of the smeared reinforcement components. For the i^{th} reinforcement, the global component reinforcement stress $\mathbf{s}_{si}$ is related to the local reinforcement stress σ'_{si} by the transformation:

$$\mathbf{s}_{si} = \mathbf{T}_\sigma p_i \sigma'_{si} \tag{2.4}$$

where p_i is the reinforcing ratio $p_i = \frac{A_{si}}{A_c}$, A_{si} is the reinforcement cross section area. The local reinforcement stress σ'_{si} is acting on the reinforcement area A_{si}

The stress and strain vectors are transformed according to the equations bellow in the plane stress state. New axes *u, v* are rotated from the global *x, y* axes by the angle α. The angle α is positive in the counterclockwise direction, as shown in Fig. 2-3.

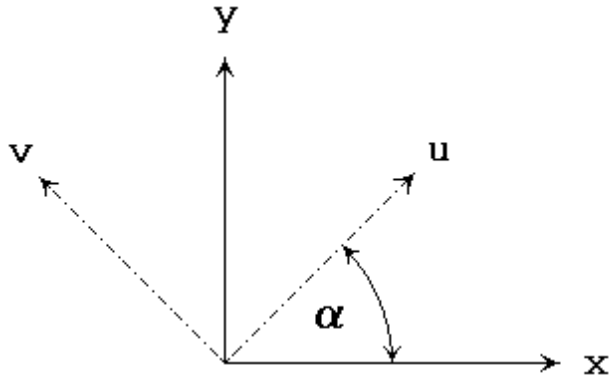


Fig. 2-3 Rotation of reference coordinate axes.

The transformation of the stresses:

$$\mathbf{s}_{(u)} = \mathbf{T}_\sigma \mathbf{s}_{(x)} \tag{2.5}$$

$$\mathbf{T}_{\sigma} = \begin{bmatrix} \cos(\alpha)^2 & \sin(\alpha)^2 & 2\cos(\alpha)\sin(\alpha) \\ \sin(\alpha)^2 & \cos(\alpha)^2 & -2\cos(\alpha)\sin(\alpha) \\ -\cos(\alpha)\sin(\alpha) & \cos(\alpha)\sin(\alpha) & \cos(\alpha)^2 - \sin(\alpha)^2 \end{bmatrix} \tag{2.6}$$

$\mathbf{s}_{(u)} = \{\sigma_u, \sigma_v, \tau_{uv}\}^T$, $\mathbf{s}_{(x)} = \{\sigma_x, \sigma_y, \tau_{xy}\}^T$

The transformation of the strains:

$$\mathbf{e}_{(u)} = \mathbf{T}_{\varepsilon}\,\mathbf{e}_{(x)} \tag{2.7}$$

$$\mathbf{T}_{\varepsilon} = \begin{bmatrix} \cos(\alpha)^2 & \sin(\alpha)^2 & \cos(\alpha)\sin(\alpha) \\ \sin(\alpha)^2 & \cos(\alpha)^2 & -\cos(\alpha)\sin(\alpha) \\ -2\cos(\alpha)\sin(\alpha) & 2\cos(\alpha)\sin(\alpha) & \cos(\alpha)^2 - \sin(\alpha)^2 \end{bmatrix} \tag{2.8}$$

$\mathbf{e}_{(u)} = \{\varepsilon_u, \varepsilon_v, \gamma_{uv}\}^T$, $\mathbf{e}_{(x)} = \{\varepsilon_x, \varepsilon_y, \gamma_{xy}\}^T$.

The angles of principal axes of the stresses and strains, Fig. 2-1, Fig. 2-2, are found from the equations:

$$\tan(2\vartheta_{\sigma}) = \frac{2\tau_{xy}}{\sigma_x - \sigma_y},\ \tan(2\vartheta_{\varepsilon}) = \frac{\gamma_{xy}}{\varepsilon_x - \varepsilon_y} \tag{2.9}$$

where ϑ_{σ} is the angle of the first principal stress axis and ϑ_{ε} is the angle of the first principal strain axis.

In case of isotropic material (un-cracked concrete) the principal directions of the stress and strains are identical; in case of anisotropic material (cracked concrete) they can be different. The sign convention for the normal stresses, employed within this program, uses the positive values for the tensile stress (strain) and negative values for the compressive stress (strain). The shear stress (strain) is positive if acting upwards on the right face of a unit element.

2.1.1.2 Concept of Material Model SBETA

The material model SBETA includes the following effects of concrete behavior:

- non-linear behavior in compression including hardening and softening,
- fracture of concrete in tension based on the nonlinear fracture mechanics,
- biaxial strength failure criterion,
- reduction of compressive strength after cracking,
- tension stiffening effect,
- reduction of the shear stiffness after cracking (variable shear retention),
- two crack models: fixed crack direction and rotated crack direction.

Perfect bond between concrete and reinforcement is assumed within the smeared concept. No bond slip can be directly modeled except for the one included inherently in the tension stiffening. However, on a macro-level a relative slip displacement of reinforcement with respect to concrete over a certain distance can arise if concrete is cracked or crushed. This corresponds to a real mechanism of bond failure in case of the bars with ribs.

The reinforcement in both forms, smeared and discrete, is in the uniaxial stress state and its constitutive law is a multi-linear stress-strain diagram.

The material matrix is derived using the nonlinear elastic approach. In this approach the elastic constants are derived from a stress-strain function called here the equivalent uniaxial law. This approach is like the nonlinear hypo-elastic constitutive model, except that different laws are used here for loading and unloading, causing the dissipation of energy exhausted for the damage of material. The detailed treatment of the theoretical background of this subject can be found, for example, in the book CHEN (1982). This approach can be also regarded as an isotropic damage model, with the unloading modulus (see next section) representing the damage modulus.

The name SBETA comes from the former program, in which this material model was first used. It means the abbreviation for the analysis of reinforced concrete in German language - StahlBETonAnalyse.

2.1.2 Stress-Strain Relations for Concrete

2.1.2.1 Equivalent Uniaxial Law

The nonlinear behavior of concrete in the biaxial stress state is described by means of the so-called effective stress σ_c^{ef}, and the equivalent uniaxial strain ε^{eq}. The effective stress is in most cases a principal stress.

The equivalent uniaxial strain is introduced to eliminate the Poisson's effect in the plane stress state.

$$\varepsilon^{eq} = \frac{\sigma_{ci}}{E_{ci}} \tag{2.10}$$

The equivalent uniaxial strain can be considered as the strain, that would be produced by the stress σ_{ci} in a uniaxial test with modulus E_{ci} associated with the direction *i*. Within this assumption, the nonlinearity representing a damage is caused only by the governing stress σ_{ci}. The details can be found in CHEN (1982).

The complete equivalent uniaxial stress-strain diagram for concrete is shown in Fig. 2-4.

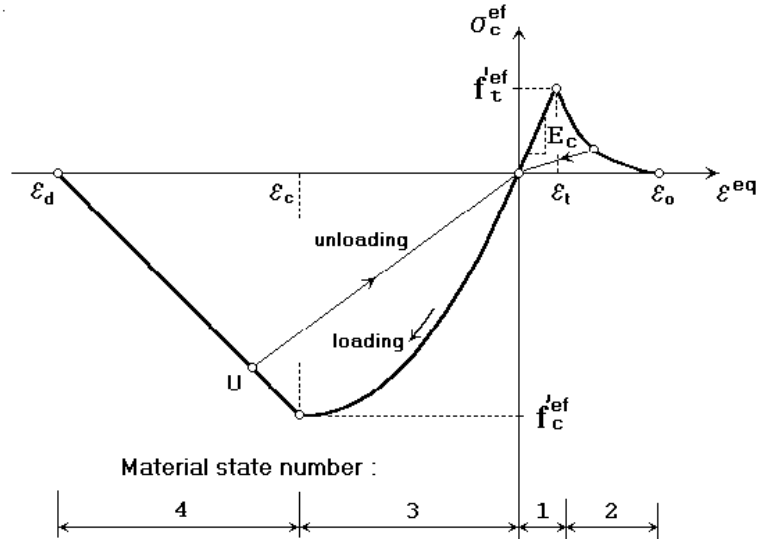


Fig. 2-4 Uniaxial stress-strain law for concrete.

The numbers of the diagram parts in Fig. 2-4 (material state numbers) are used in the results of the analysis to indicate the state of damage of concrete.

Unloading is a linear function to the origin. An example of the unloading point U is shown in Fig. 2-4. Thus, the relation between stress σ_c^{ef} and strain ε^{eq} is not unique and depends on a load history. A change from loading to unloading occurs when the increment of the effective strain changes the sign. If subsequent reloading occurs the linear unloading path is followed until the last loading point U is reached again. Then, the loading function is resumed.

The peak values of stress in compression f'^{ef}_c and in tension f'^{ef}_t are calculated according to the biaxial stress state as will be shown in Sec.2.1.5. Thus, the equivalent uniaxial stress-strain law reflects the biaxial stress state.

The above defined stress-strain relation is used to calculate the elastic modulus for the material stiffness matrices, Sect. 2.1.11. The secant modulus is calculated as

$$E_c^{\ s} = \frac{\sigma_c}{\varepsilon^{eq}} \tag{2.11}$$

It is used in the constitutive equation to calculate stresses for the given strain state, Sect. 2.1.12.

The tangent modulus $E_c^{\ t}$ is used in the material matrix $\mathbf{D}_c$ for construction of an element stiffness matrix for the iterative solution. The tangent modulus is the slope of the stress-strain curve at a given strain. It is always positive. In cases when the slope of the curve is less then the minimum value $E_{min}^{\ t}$ the value of the tangent modulus is set $E_c^{\ t} = E_{min}^{\ t}$. This occurs in the softening ranges and near the compressive peak.

Detail description of the stress-strain law is given in the following subsections.

2.1.2.2 Tension before Cracking

The behavior of concrete in tension without cracks is assumed linear elastic. E_c is the initial elastic modulus of concrete, $f_t^{'ef}$ is the effective tensile strength derived from the biaxial failure function, Section 2.1.5.2.

$$\sigma_c^{\ ef} = E_c \varepsilon^{eq}, 0 \le \sigma_c \le f_t^{'ef} \tag{2.12}$$

2.1.2.3 Tension after Cracking

Two types of formulations are used for the crack opening:

- A fictitious crack model based on a crack-opening law and fracture energy. This formulation is suitable for modeling of crack propagation in concrete. It is used in combination with the crack band, see Sect.2.1.3.
- A stress-strain relation in a material point. This formulation is not suitable for normal cases of crack propagation in concrete and should be used only in some special cases.

In following subsections are described five softening models included in SBETA material model.

(1) Exponential Crack Opening Law

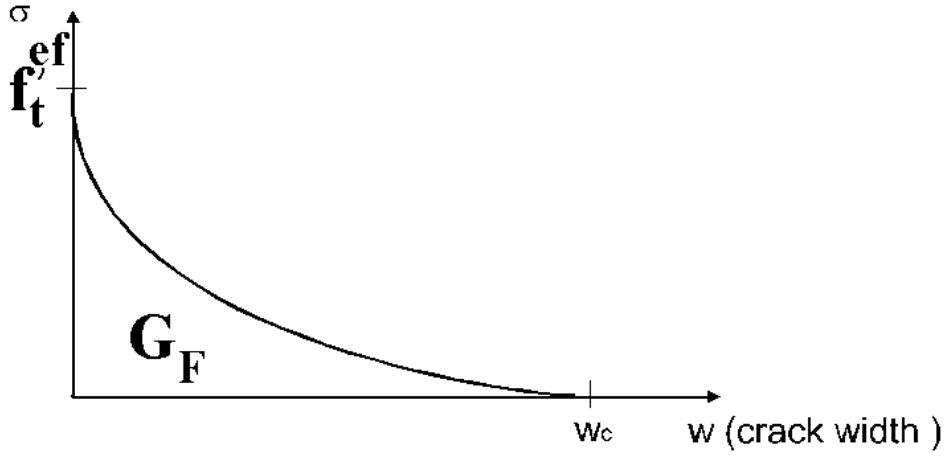


Fig. 2-5 Exponential crack opening law.

This function of crack opening was derived experimentally by HORDIJK (1991).

$$\frac{\sigma}{f_t^{'ef}} = \left\{1+\left(c_1\frac{w}{w_c}\right)^3\right\}\exp\left(-c_2\frac{w}{w_c}\right)-\frac{w}{w_c}\left(1+c_1^3\right)\exp\left(-c_2\right), \tag{2.13}$$

$$w_c = 5.14\frac{G_f}{f_t^{'ef}}$$

where w is the crack opening, w_c is the crack opening at the complete release of stress, σ is the normal stress in the crack (crack cohesion). Values of the constants are, c_1=3, c_2=6.93. G_f is the fracture energy needed to create a unit area of stress-free crack, $f_t^{'ef}$ is the effective tensile strength derived from a failure function, Eq.(2.22). The crack opening displacement w is derived from strains according to the crack band theory in Eq.(2.18).

(2) Linear Crack Opening Law

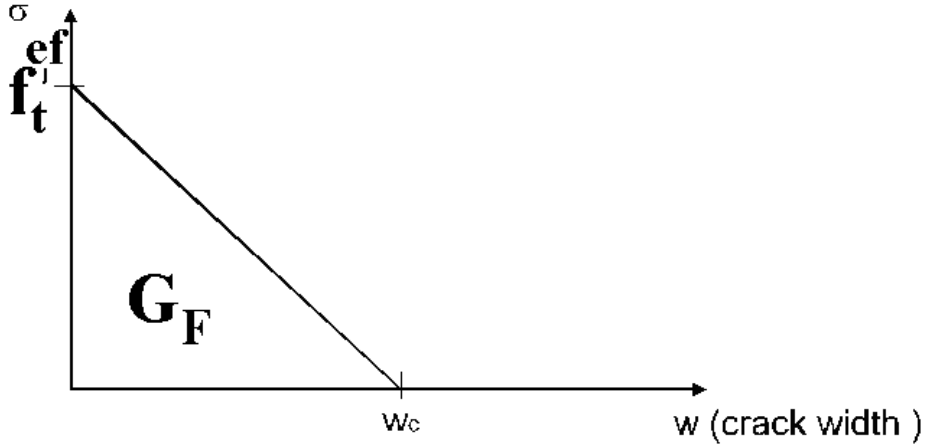


Fig. 2-6 Linear crack opening law.

$$\frac{\sigma_c^{ef}}{f_t^{'ef}} = \frac{f_t^{'}}{w_c}\left(w_c - w\right), w_c = \frac{2G_f}{f_t^{'}} \tag{2.14}$$

(3) Linear Softening Based on Local Strain

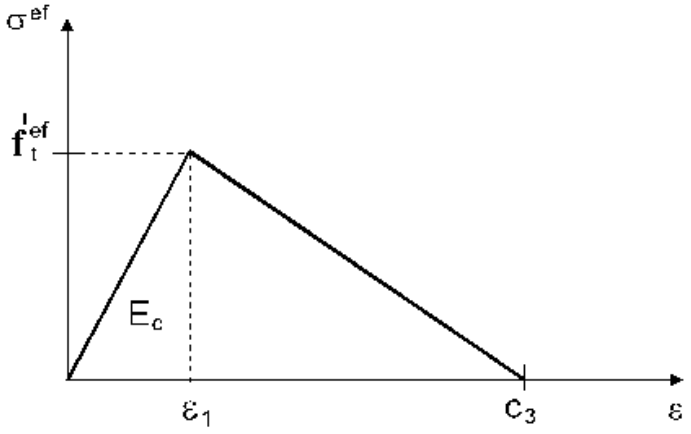


Fig. 2-7 Linear softening based on strain.

The descending branch of the stress-strain diagram is defined by the strain c_3 corresponding to zero stress (complete release of stress).

(4) SFRC Based on Fracture Energy

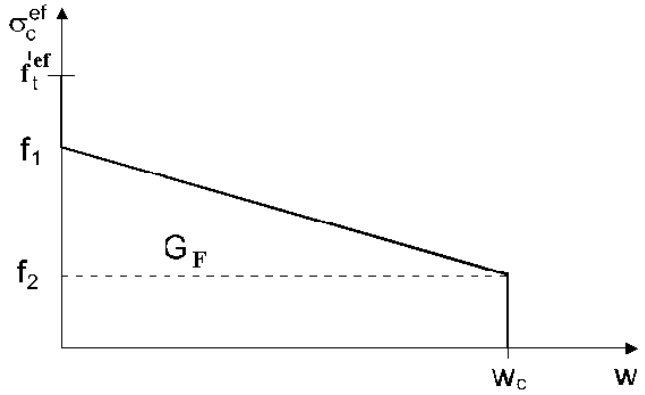


Fig. 2-8 Steel fiber reinforced concrete based on fracture energy.

Parameters: $$c_1 = \frac{f_1}{f_t'^{ef}}, c_2 = \frac{f_2}{f_t'^{ef}}, w_c = \frac{2G_f}{f_1 + f_2}$$

(5) SFRC Based on Strain

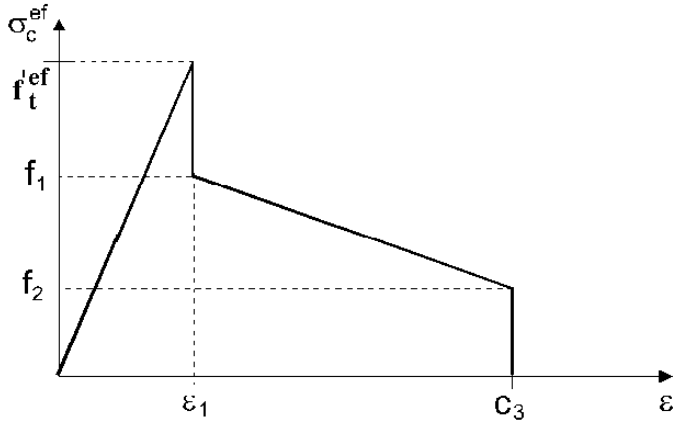


Fig. 2-9 Steel fiber reinforced concrete based on strain.

Parameters: $c_1 = \frac{f_1}{f_t^{'ef}}, c_2 = \frac{f_2}{f_t^{'ef}}$

Parameters c_1 and c_2 are relative positions of stress levels, and c_3 is the end strain.

2.1.2.4 Compression before Peak Stress

The formula recommended by CEB-FIP Model Code 90 has been adopted for the ascending branch of the concrete stress-strain law in compression, Fig. 2-10. This formula enables wide range of curve forms, from linear to curved, and is appropriate for normal as well as high strength concrete.

$$\sigma_c^{ef} = f_c^{'ef} \frac{kx - x^2}{1 + (k-2)x}, x = \frac{\varepsilon}{\varepsilon_c}, k = \frac{E_o}{E_c} \tag{2.15}$$

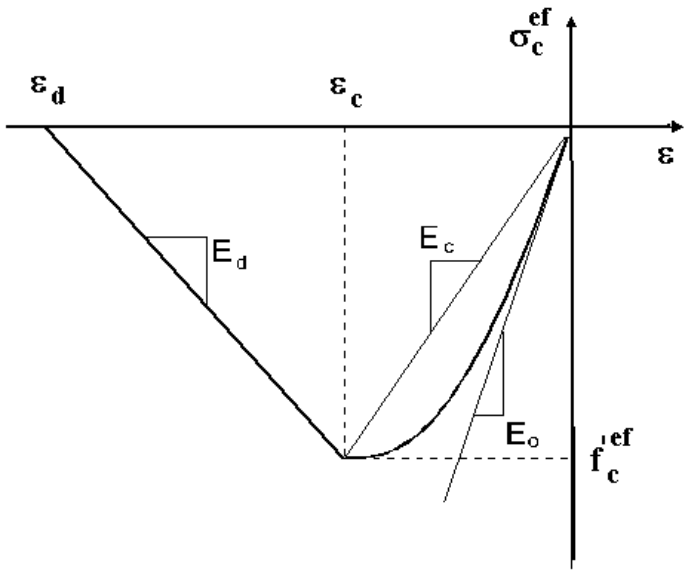


Fig. 2-10 Compressive stress-strain diagram.

Meaning of the symbols in the above formula in:

σ_c^{ef} - concrete compressive stress,

$f_c^{'ef}$ - concrete effective compressive strength (See Section 2.1.5.1)

x - normalized strain,

ε - strain,

ε_c - strain at the peak stress f'^{ef}_c,

k - shape parameter,

E_o - initial elastic modulus,

E_c - secant elastic modulus at the peak stress, $E_c = \frac{f_c^{'ef}}{\varepsilon_c}$.

Parameter k may have any positive value greater than or equal 1. Examples: k=1. linear, k=2. - parabola.

As a consequence of the above assumption, distributed damage is considered before the peak stress is reached. Contrary to the localized damage, which is considered after the peak.

2.1.2.5 Compression after Peak Stress

The softening law in compression is linearly descending. There are two models of strain softening in compression, one based on dissipated energy, and other based on local strain softening.

2.1.2.5.1 Fictitious Compression Plane Model

The fictitious compression plane model assumes, that compression failure is localized in a plane normal to the direction of compressive principal stress. All post-peak compressive displacements and energy dissipation are localized in this plane. It is assumed that this displacement is independent on the size of the structure. This hypothesis is supported by experiments conducted by Van MIER (1986).

This assumption is analogous to the Fictitious Crack Theory for tension, where the shape of the crack-opening law and the fracture energy are defined and are considered as material properties.

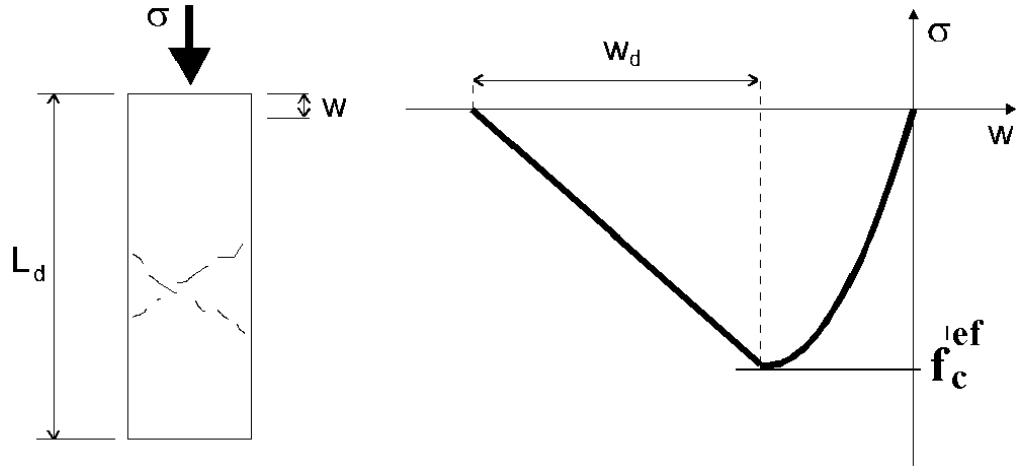


Fig. 2-11 Softening displacement law in compression.

In case of compression, the end point of the softening curve is defined by means of the plastic displacement w_d. In this way, the energy needed for generation of a unit area of the failure plane is indirectly defined. From the experiments of Van MIER (1986), the value of w_d =0.5mm for normal concrete. This value is used as default for the definition of the softening in compression.

The softening law is transformed from a fictitious failure plane, Fig. 2-11, to the stress-strain relation valid for the corresponding volume of continuous material, Fig. 2-10. The slope of the softening part of the stress-strain diagram is defined by two points: a peak of the diagram at the maximal stress and a limit compressive strain ε_d at the zero stress. This strain is calculated from a plastic displacement w_d and a band size $L_d^{'}$ (see Section 2.1.3) according to the following expression:

$$\varepsilon_d = \varepsilon_c + \frac{w_d}{L_d^{'}} \tag{2.16}$$

The advantage of this formulation is reduced dependency on finite element mesh.

2.1.2.5.2 Compression Strain Softening Law Based on Strain.

A slope of the softening law is defined by means of the softening modulus E_d . This formulation is dependent on the size of the finite element mesh.

2.1.3 Localization Limiters

So-called localization limiter controls localization of deformations in the failure state. It is a region (band) of material, which represents a discrete failure plane in the finite element analysis. In tension it is a crack, in compression it is a plane of crushing. These failure regions have some dimension. However, since according to the experiments, the dimensions of the failure regions are independent on the structural size, they are assumed as fictitious planes. In case of tensile cracks, this approach is known as rack the "crack band theory", BAZANT, OH (1983). Here is the same concept used also for the compression failure. The purpose of the failure band is to eliminate two deficiencies, which occur in connection with the application of the finite element model: element size effect and element orientation effect.

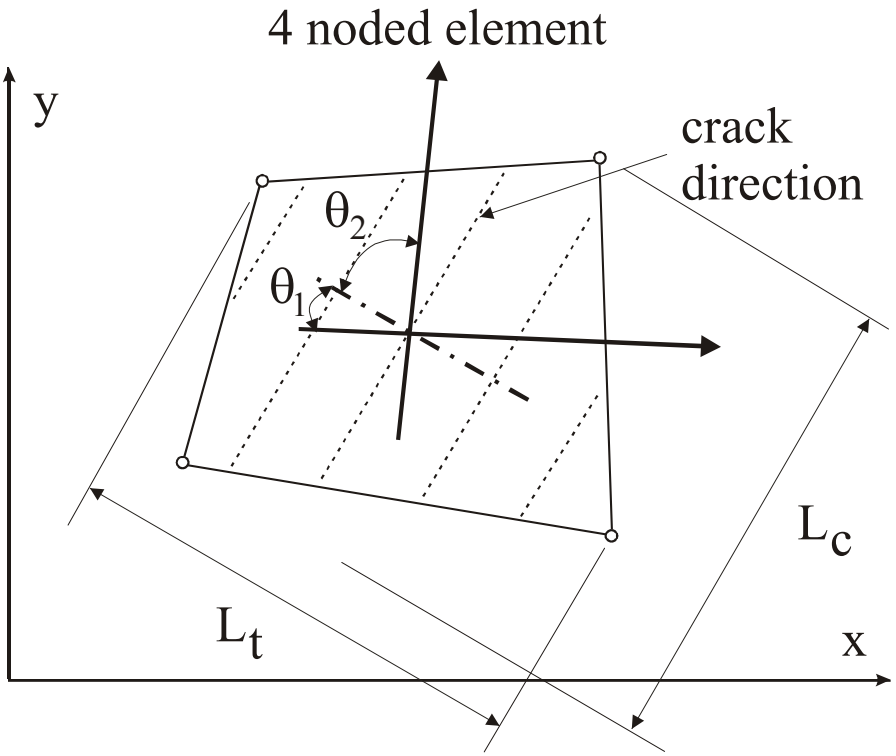


Fig. 2-12 Definition of localization bands.

2.1.3.1 Element Size Effect.

The direction of the failure planes is assumed to be normal to the principal stresses in tension and compression, respectively. The failure bands (for tension L_t and for compression L_d) are defined as projections of the finite element dimensions on the failure planes as shown in Fig. 2-12.

2.1.3.2 Element Orientation Effect.

The element orientation effect is reduced, by further increasing of the failure band for skew meshes, by the following formula (proposed by CERVENKA et al. 1995).

$$L_t^{'} = \gamma L_t, L_d^{'} = \gamma L_d$$

$$\gamma = 1 + (\gamma^{\max} - 1)\frac{\theta}{45}, \qquad \theta \in \langle 0; 45 \rangle \tag{2.17}$$

An angle θ is the minimal angle ($\min(\theta_1, \theta_2)$) between the direction of the normal to the failure plane and element sides. In case of a general quadrilateral element the element sides directions are calculated as average side directions for the two opposite edges. The above formula is a linear interpolation between the factor γ=1.0 for the direction parallel with element sides, and $\gamma = \gamma^{max}$, for the direction inclined at 45°. The recommended (and default) value of γ^{max} =1.5.

2.1.4 Fracture Process, Crack Width

The process of crack formation can be divided into three stages, Fig. 2-13. The uncracked stage is before a tensile strength is reached. The crack formation takes place in the process zone of a potential crack with decreasing tensile stress on a crack face due to a bridging effect. Finally, after a complete release of the stress, the crack opening continues without the stress.

The crack width *w* is calculated as a total crack opening displacement within the crack band.

$$w = \varepsilon_{cr} L_t' \tag{2.18}$$

where ε_{cr} is the crack opening strain, which is equal to the strain normal to the crack direction in the cracked state after the complete stress release.

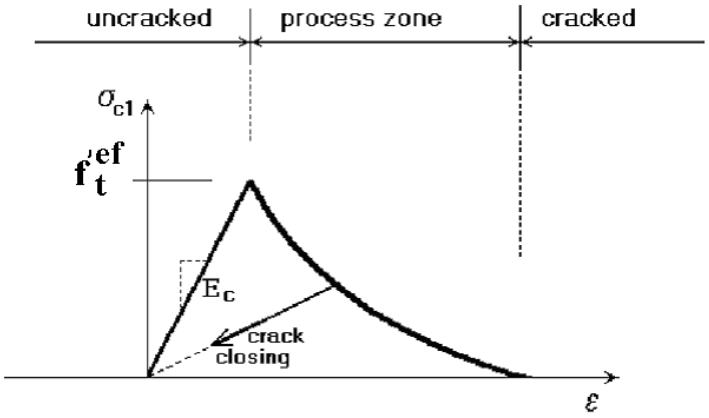


Fig. 2-13 Stages of crack opening.

It has been shown that the smeared model based on the refined crack band theory can successfully describe the discrete crack propagation in plain, as well as reinforced concrete (CERVENKA et al. 1991, 1992, and 1995).

It is also possible, that the second stress, parallel to the crack direction, exceeds the tensile strength. Then the second crack, in the direction orthogonal to the first one, is formed using the same softening model as the first crack. (Note: The second crack may not be shown in a graphical post-processing. It can be identified by the concrete state number in the second direction at the numerical output.)

2.1.5 Biaxial Stress Failure Criterion of Concrete

2.1.5.1 Compressive Failure

A biaxial stress failure criterion according to KUPFER et al. (1969) is used as shown in Fig. 2-14. In the compression-compression stress state the failure function is

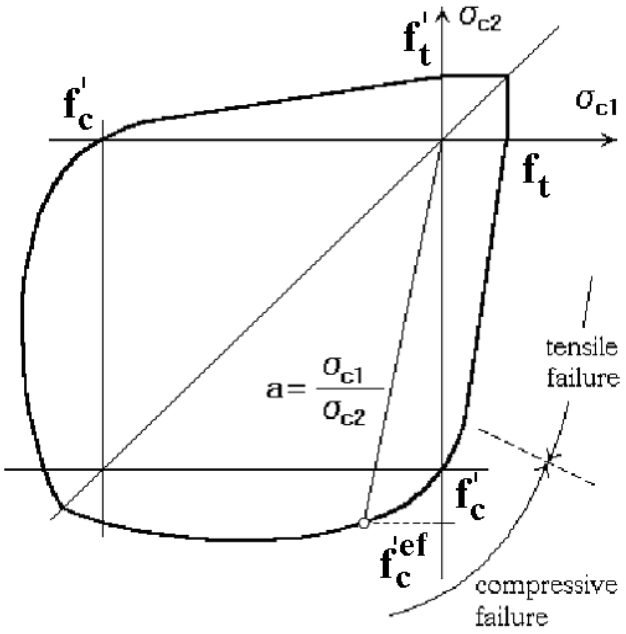


Fig. 2-14 Biaxial failure function for concrete.

$$f_c^{'ef} = \frac{1+3.65a}{(1+a)^2} f_c', a = \frac{\sigma_{c1}}{\sigma_{c2}} \tag{2.19}$$

where σ_{c1}, σ_{c2} are the principal stresses in concrete and f'_c is the uniaxial cylinder strength. In the biaxial stress state, the strength of concrete is predicted under the assumption of a proportional stress path.

In the tension-compression state, the failure function continues linearly from the point $\sigma_{c1} = 0, \sigma_{c2} = f_c'$ into the tension-compression region with the linearly decreasing strength:

$$f_c^{'ef} = f_c' r_{ec}, \quad r_{ec} = (1 + 5.3278\frac{\sigma_{c1}}{f_c'}), \quad 1.0 \geq r_{ec} \geq 0.9 \tag{2.20}$$

where r_{ec} is the reduction factor of the compressive strength in the principal direction 2 due to the tensile stress in the principal direction 1.

2.1.5.2 Tensile Failure

In the tension-tension state, the tensile strength is constant and equal to the uniaxial tensile strength f'_t. In the tension-compression state, the tensile strength is reduced by the relation:

$$f_t^{'ef} = f_t' r_{et} \tag{2.21}$$

where r_{et} is the reduction factor of the tensile strength in the direction 1 due to the compressive stress in the direction 2. The reduction function has one of the following forms, Fig. 2-15.

$$r_{et} = 1 - 0.95\frac{\sigma_{c2}}{f_c'} \tag{2.22}$$

$$r_{et} = \frac{A + (A-1)B}{AB}, B = Kx + A, x = \frac{\sigma_{c2}}{f_c'} \tag{2.23}$$

The relation in Eq.(2.22) is the linear decrease of the tensile strength and (2.23) is the hyperbolic decrease.

Two predefined shapes of the hyperbola are given by the position of an intermediate point *r, x*. Constants *K* and *A* define the shape of the hyperbola. The values of the constants for the two positions of the intermediate point are given in the following table.

type	*point*		*parameters*	
	r	*x*	*A*	*K*
a	0.5	0.4	0.75	1.125
b	0.5	0.2	1.0625	6.0208

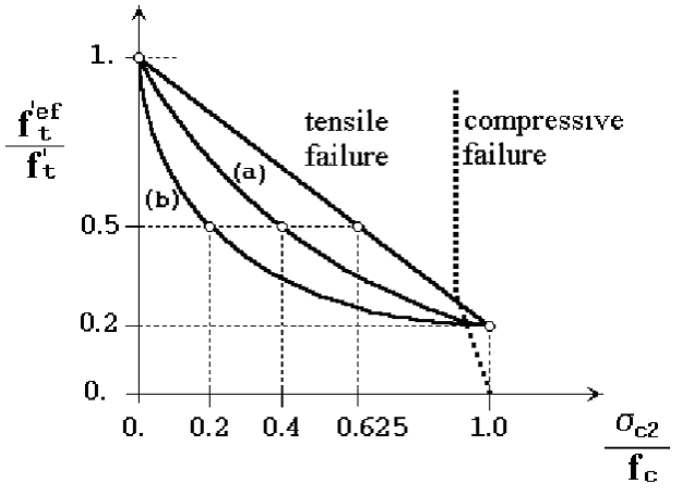


Fig. 2-15 Tension-compression failure function for concrete.

2.1.6 Two Models of Smeared Cracks

The smeared crack approach for modeling of the cracks is adopted in the model SBETA. Within the smeared concept two options are available for crack models: the fixed crack model and the rotated crack model. In both models the crack is formed when the principal stress exceeds the tensile strength. It is assumed that the cracks are uniformly distributed within the material volume. This is reflected in the constitutive model by an introduction of orthotropy.

2.1.6.1 Fixed Crack Model

In the fixed crack model (CERVENKA 1985, DARWIN 1974) the crack direction is given by the principal stress direction at the moment of the crack initiation. During further loading this direction is fixed and represents the material axis of the orthotropy.

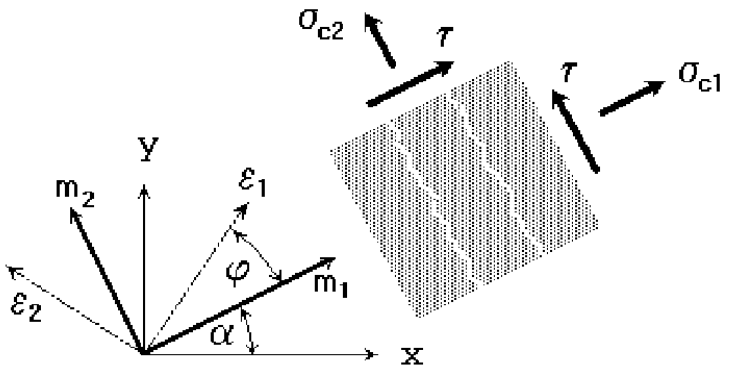


Fig. 2-16 Fixed crack model. Stress and strain state.

The principal stress and strain directions coincide in the uncracked concrete, because of the assumption of isotropy in the concrete component. After cracking the orthotropy is introduced. The weak material axis m_1 is normal to the crack direction, the strong axis m_2 is parallel with the cracks.

In a general case the principal strain axes ε_1 and ε_2 rotate and need not to coincide with the axes of the orthotropy m_1 and m_2. This produces a shear stress on the crack face as shown in Fig. 2-16. The stress components σ_{c1} and σ_{c2} denote, respectively, the stresses normal and parallel to the crack plane and, due to shear stress, they are not the principal stresses. The shear stress and stiffness in the cracked concrete is described in Section 2.1.7.

2.1.6.2 Rotated Crack Model

In the rotated crack model (VECCHIO 1986, CRISFIELD 1989), the direction of the principal stress coincides with the direction of the principal strain. Thus, no shear strain occurs on the crack plane and only two normal stress components must be defined, as shown in Fig. 2-17.

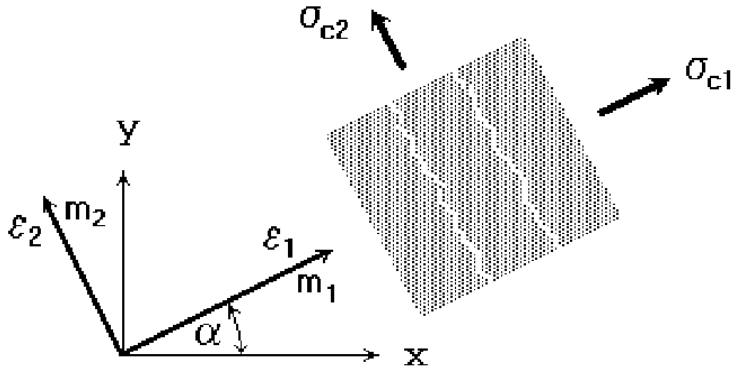


Fig. 2-17 Rotated crack model. Stress and strain state.

If the principal strain axes rotate during the loading the direction of the cracks rotate, too. In order to ensure the co-axiality of the principal strain axes with the material axes the tangent shear modulus G_t is calculated according to CRISFIELD 1989 as

$$G_t = \frac{\sigma_{c1} - \sigma_{c2}}{2(\varepsilon_1 - \varepsilon_2)} \tag{2.24}$$

2.1.7 Shear Stress and Stiffness in Cracked Concrete

In case of the fixed crack model, the shear modulus is reduced according to the law derived by KOLMAR (1986) after cracking. The shear modulus is reduced with growing strain normal to the crack, Fig. 2-18 and this represents a reduction of the shear stiffness due to the crack opening.

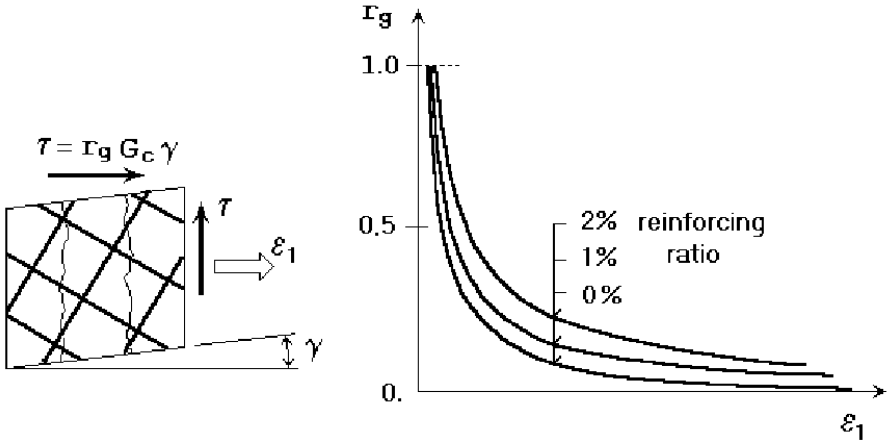


Fig. 2-18 Shear retention factor.

$$G = r_g\, G_c, \quad r_g = c_3 \frac{-\ln\left(\frac{1000\varepsilon_u}{c_1}\right)}{c_2} \tag{2.25}$$

$$c_1 = 7 + 333(p - 0.005), c_2 = 10 - 167(p - 0.005), 0 \le p \le 0.02$$

where r_g is the shear retention factor, G is the reduced shear modulus and G_c is the initial concrete shear modulus:

$$G_c = \frac{E_c}{2(1+\nu)} \tag{2.26}$$

where E_c is the initial elastic modulus and ν is the Poisson's ratio. The strain ε_υ is normal to the crack direction (the crack opening strain), c_1 and c_2 are parameters depending on the reinforcing crossing the crack direction, p is the transformed reinforcing ratio (all reinforcement is transformed on the crack plane) and c_3 is the user's scaling factor. By default, c_3=1. In ATENA the effect of reinforcement ratio is not considered, and p is assumed to be 0.0.

There is an additional constraint imposed on the shear modulus. The shear stress on the crack plane $\tau_{uv} = G\gamma$ is limited by the tensile strength f'_t. The secant and tangent shear moduli of cracked concrete are equal.

2.1.8 Compressive Strength of Cracked Concrete

A reduction of the compressive strength after cracking in the direction parallel to the cracks is done by a similar way as found from experiments of VECCHIO and COLLINS 1982 and formulated in the Compression Field Theory. However, a different function is used for the reduction of concrete strength here, to allow for user's adjustment of this effect. This function has the form of the Gauss's function, Fig. 2-19. The parameters of the function were derived from the experimental data published by KOLLEGER et al. 1988, which also included data of Collins and Vecchio (VECCHIO at al.1982)

$$f_c^{'ef} = r_c f_c^{'}, r_c = c + (1-c)e^{-(128\varepsilon_u)^2} \tag{2.27}$$

For the zero normal strain, $\varepsilon_{v,}$ there is no strength reduction, and for the large strains, the strength is asymptotically approaching to the minimum value $f_c^{'ef} = cf_c^{'}$.

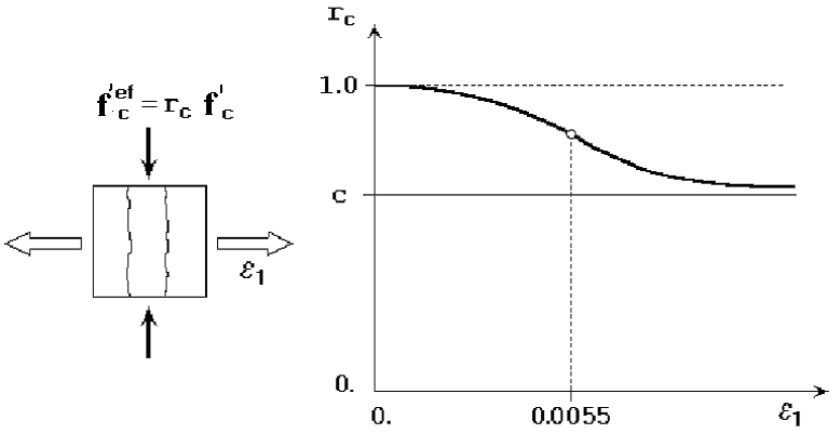


Fig. 2-19 Compressive strength reduction of cracked concrete.

The constant *c* represents the maximal strength reduction under the large transverse strain. From the experiments by KOLLEGGER et all. 1988, the value *c* = 0.45 was derived for the concrete reinforced with the fine mesh. The other researchers (DYNGELAND 1989) found the reductions not less than *c*=0.8. The value of *c* can be adjusted by input data according to the actual type of reinforcing.

However, the reduction of compressive strength of the cracked concrete does not have to be affected only by the reinforcing. In the plain concrete, when the strain localizes in one main crack, the compressive concrete struts can cross this crack, causing so-called "bridging effect". The compressive strength reduction of these bridges is also captured by the above model.

2.1.9 Tension Stiffening in Cracked Concrete

The tension stiffening effect can be described as a contribution of cracked concrete to the tensile stiffness of reinforcing bars. This stiffness is provided by the uncracked concrete or not fully opened cracks and is generated by the strain localization process. It was verified by simulation experiments of HARTL, G., 1977 and published in the paper (MARGOLDOVA et.al. 1998).

Including an explicit tension stiffening factor would result in an overestimation of this effect. Therefore, in the ATENA versions up to1.2.0 no explicit tension stiffening factor is possible in the input.

2.1.10 Summary of Stresses in SBETA Constitutive Model

In the case of uncracked concrete, the stress symbols have the following meaning:

σ_{c1} - maximal principal stress

σ_{c2} - minimal principal stress

(tension positive, compression negative)

In the case of cracked concrete, Fig. 2-16 stresses are defined on the crack plane:

σ_{c1} - normal stress normal to the cracks

σ_{c2} - normal stress parallel to the cracks

τ_c - shear stress on the crack plane

2.1.11 Material Stiffness Matrices

2.1.11.1 Uncracked Concrete

The material stiffness matrix for the uncracked concrete has the form of an elastic matrix of the isotropic material. It is written in the global coordinate system *x* and *y*.

$$\mathbf{D}_c = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \tag{2.28}$$

In the above E is the concrete elastic modulus derived from the equivalent uniaxial law. The Poisson's ratio ν is constant.

2.1.11.2 Cracked Concrete

For the cracked concrete, the matrix has the form of the elastic matrix for the orthotropic material. The matrix is formulated in a coordinate system *m1, m2,* Fig. 2-16 and Fig. 2-17*,* which is coincident with the crack direction. This local coordinate system is referred to the superscript *L* later. The direction 1 is normal to the crack and the direction 2 is parallel with the crack. The definition of the elastic constants for the orthotropic material in the plane stress state follows from the flexibility relation:

$$\begin{Bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \gamma \end{Bmatrix} = \begin{bmatrix} \frac{1}{E_1} & -\frac{\nu_{21}}{E_2} & 0 \\ -\frac{\nu_{12}}{E_1} & \frac{1}{E_2} & 0 \\ 0 & 0 & \frac{1}{G} \end{bmatrix} \begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \tau \end{Bmatrix} \tag{2.29}$$

First, we eliminate the orthotropic Poisson's ratios for the cracked concrete, because they are commonly not known. For this we use the symmetry relation $\nu_{12}E_2 = \nu_{21}E_1$. Therefore, in (2.29) there are only three independent elastic constants E_1, E_2, ν_{21}. Assuming that $\nu_{21} = \nu$ is the Poisson's ratio of the uncracked concrete and using the symmetry relation, we obtain

$$\nu_{12} = \frac{E_1}{E_2}\nu \tag{2.30}$$

The stiffness matrix $\mathbf{D}_c^{\,L}$ is found as the inverse of the flexibility matrix in (2.30):

$$\mathbf{D}_c^{\,L} = H\begin{bmatrix} \xi & \nu\xi & 0 \\ \nu\xi & 1 & 0 \\ 0 & 0 & G \end{bmatrix}, \qquad \xi = \frac{E_1}{E_2}, H = E_1(1-\xi\nu^2) \tag{2.31}$$

In the above relation E_2 must be nonzero. If E_2 is zero and E_1 is nonzero, then an alternative formulation is used with the inverse parameter $\frac{1}{\xi} = \frac{E_2}{E_1}$. In case that both elastic modules are zero, the matrix $\mathbf{D}_c^{\,L}$ is set equal to the null matrix.

The matrix $\mathbf{D}_c^{\,L}$ is transformed into the global coordinate system using the transformation matrix $\mathbf{T}_\varepsilon$ from (2.8).

$$\mathbf{D}_c = \mathbf{T}_\varepsilon^T \mathbf{D}_c^L \mathbf{T}_\varepsilon \tag{2.32}$$

The angle α is between the global axis x and the 1st material axis m_1, which is normal to the crack, Fig. 2-16.

2.1.11.3 Smeared Reinforcement

The material stiffness matrix of the i^{th} smeared reinforcement is

$$\mathbf{D}_{si} = p_i E_{si} \begin{bmatrix} \cos(\beta_i)^4 & \cos(\beta_i)^2 \sin(\beta_i)^2 & \cos(\beta_i)^3 \sin(\beta_i) \\ \cos(\beta_i)^2 \sin(\beta_i)^2 & \sin(\beta_i)^4 & \cos(\beta_i) \sin(\beta_i)^3 \\ \cos(\beta_i)^3 \sin(\beta_i) & \cos(\beta_i) \sin(\beta_i)^3 & \cos(\beta_i)^2 \sin(\beta_i)^2 \end{bmatrix} \tag{2.33}$$

The angle β is between the global axis x and the i^{th} reinforcement direction, and E_{si} is the elastic modulus of reinforcement. The reinforcing ratio $p_i = A_s/A_c$.

2.1.11.4 Material Stiffness of Composite Material

The total material stiffness of the reinforced concrete is the sum of material stiffness of concrete and smeared reinforcement:

$$\mathbf{D} = \mathbf{D}_c + \sum_{i=1}^{n} \mathbf{D}_{si} \tag{2.34}$$

The summation is over n smeared reinforcing components. In ATENA the smeared reinforcement is not added on the constitutive level, but it is modeled by separate layers of elements whose nodes are connected to those of the concrete elements. This corresponds to the assumption of perfect bond between the smeared reinforcement and concrete.

2.1.11.5 Secant and Tangent Material Stiffness

The material stiffness matrices in the above Subsections 2.1.11.1, 2.1.11.2, 2.1.11.3, 2.1.11.4 are either secant or tangent, depending on the type of elastic modulus used.

The secant material stiffness matrix is used to calculate the stresses for the given strains, as shown in Section 2.1.12.

The tangent material stiffness matrix is used to construct the element stiffness matrix.

2.1.12 Analysis of Stresses

The stresses in concrete are obtained using the actual secant component material stiffness matrix

$$\mathbf{s}_c = \mathbf{D}_c^{\,s} \mathbf{e} \tag{2.35}$$

where $\mathbf{D}_c^s$ is the secant material stiffness matrix from Section 2.1.11 for the uncracked or cracked concrete depending on the material state. The stress components are calculated in the global as well as in the local material coordinates (the principal stresses in the uncracked concrete and the stresses on the crack planes).

The stress in reinforcement and the associated tension stiffening stress is calculated directly from the strain in the reinforcement direction.

2.1.13 Parameters of Constitutive Model

Default formulas of material parameters:

Parameter:	*Formula:*
Cylinder strength	$f_c^{'} = -0.85 f_{cu}^{'}$
Tensile strength	$f_t^{'} = 0.24 f_{cu}^{'\frac{2}{3}}$
Initial elastic modulus	$E_c = (6000 - 15.5 f_{cu}^{'}) \sqrt{f_{cu}^{'}}$
Poisson's ratio	$\nu = 0.2$
Softening compression	$w_d = -0.0005 mm$
Type of tension softening	1 – exponential, based on G_F
Compressive strength in cracked concrete	$c = 0.8$
Tension stiffening stress	$\sigma_{st} = 0.$
Shear retention factor	variable (Sect.2.1.7)
Tension-compression function type	linear
Fracture energy G_f according to VOS 1983	$G_F = 0.000025 f_t^{'ef}$ *[MN/m]*
Orientation factor for strain localization	$\gamma_{\max} = 1.5$ (Sect.2.1.3)

The SBETA constitutive model of concrete includes 20 material parameters. These parameters are specified for the problem under consideration by user. In case of the parameters are not known automatic generation can be done using the default formulas given in the table above. In such a case, only the cube strength of concrete f'_{cu} (nominal strength) is specified and the remaining parameters are calculated as functions of the cube strength. The formulas for these functions are taken from the CEB-FIP Model Code 90 and other research sources.

Used units are MPa.

The parameters not listed in the table have zero default value.

The values of the material parameters can be also influenced by safety considerations. This is particularly important in cases of a design, where a proper safety margin should be met. For that reason, the choice of material properties depends on the purpose of analysis and the filed of an application. The typical examples of the application are the design, the simulation of failure and the research.

In case of the design application, according to most current standards, the material properties for calculation of structural resistance (failure load) are considered by minimal values with applied partial safety factors. The resulting maximum load can be directly compared with the design loads.

According to some researchers, more appropriate approach would be to consider the average material properties in nonlinear analysis and to apply a safety factor on the resulting integral response variable (force, moment). However, this safety format is not yet fully established.

In cases of the simulation of real behavior, the parameters should be chosen as close as possible to the properties of real materials. The best way is to determine these properties from mechanical tests on material sample specimens.

2.2 Fracture–Plastic Constitutive Model (CC3DCementitious, CC3DNonLinCementitious, CC3DNonLinCementitious2, CC3DNonLinCementitious2User, CC3DNonLinCementitious2Variable, CC3DNonLinCementitious2FRC, CC3DNonLinCementitious2SHCC, CC3DNonLinCementitious3)

2.2.1 Introduction

Fracture-plastic model combines constitutive models for tensile (fracturing) and compressive (plastic) behavior. The fracture model is based on the classical orthotropic smeared crack formulation and crack band model. It employs Rankine failure criterion, exponential softening, and it can be used as rotated or fixed crack model. The hardening/softening plasticity model is based on Menétrey-Willam failure surface. The model uses return mapping algorithm for the integration of constitutive equations. Special attention is given to the development of an algorithm for the combination of the two models. The combined algorithm is based on a recursive substitution, and it allows for the two models to be developed and formulated separately. The algorithm can handle cases when failure surfaces of both models are active, but also when physical changes such as crack closure occur. The model can be used to simulate concrete cracking, crushing under high confinement, and crack closure due to crushing in other material directions.

Although many papers have been published on plasticity models for concrete (for instance, PRAMONO, WILLAM 1989, MENETREY et al 1997, FEENSTRA 1993, 1998 ETSE 1992) or smeared crack models (RASHID 1968, CERVENKA and GERSTLE 1971, BAZANT and OH 1983, DE BORST 1986, ROTS 1989), there are not many descriptions of their successful combination in the literature. OWEN et al. (1983) presented a combination of cracking and visco-plasticity. Comprehensive treatise of the problem was provided also by de BORST (1986), and recently several works have been published on the combination of damage and plasticity (SIMO and JU 1987, MESCHKE et al. (1998). The presented model differs from the above formulations by ability to handle also physical changes like for instance crack closure, and it is not restricted to any shape of hardening/softening laws. Also, within the proposed approach it is possible to formulate the two models (i.e. plastic and fracture) entirely separately, and their combination can be provided in a different algorithm or model. From programming point of view such approach is well suited for object-oriented programming.

The method of strain decomposition, as introduced by DE BORST (1986), is used to combine fracture and plasticity models together. Both models are developed within the framework of return mapping algorithm by WILKINS (1964). This approach guarantees the solution for all magnitudes of strain increment. From an algorithmic point of view the problem is then transformed into finding an optimal return point on the failure surface.

The combined algorithm must determine the separation of strains into plastic and fracturing components, while it must preserve the stress equivalence in both models. The proposed algorithm is based on a recursive iterative scheme. It can be shown that such a recursive algorithm cannot reach convergence in certain cases such as, for instance, softening and dilating materials. For this reason, the recursive algorithm is extended by a variation of the relaxation method to stabilize convergence.

2.2.2 Material Model Formulation

The material model formulation is based on the strain decomposition into elastic ε_{ij}^{e}, plastic ε_{ij}^{p} and fracturing ε_{ij}^{f} components (DE BORST 1986).

$$\varepsilon_{ij} = \varepsilon_{ij}^{e} + \varepsilon_{ij}^{p} + \varepsilon_{ij}^{f} \tag{2.36}$$

The new stress state is then computed by the formula:

$$\sigma_{ij}^{n} = \sigma_{ij}^{n-1} + E_{ijkl}(\Delta\varepsilon_{kl} - \Delta\varepsilon_{kl}^{p} - \Delta\varepsilon_{kl}^{f}) \tag{2.37}$$

where the increments of plastic strain $\Delta\varepsilon_{ij}^{p}$ and fracturing strain $\Delta\varepsilon_{ij}^{f}$ must be evaluated based on the used material models.

2.2.3 Rankine-Fracturing Model for Concrete Cracking

Rankine criterion is used for concrete cracking

$$F_{i}^{f} = \sigma_{ii}'^{t} - f_{ti}' \leq 0 \tag{2.38}$$

It is assumed that strains and stresses are converted into the material directions, which in case of rotated crack model correspond to the principal directions, and in case of fixed crack model, are given by the principal directions at the onset of cracking. Therefore, $\sigma_{ii}'^{t}$ identifies the trial stress and f_{ti}' tensile strength in the material direction i. Prime symbol denotes quantities in the material directions. The trial stress state is computed by the elastic predictor.

$$\sigma_{ij}'^{t} = \sigma_{ij}'^{n-1} + E_{ijkl}\Delta\varepsilon_{kl}' \tag{2.39}$$

If the trial stress does not satisfy (2.38), the increment of fracturing strain in direction i can be computed using the assumption that the final stress state must satisfy (2.40).

$$F_{i}^{f} = \sigma_{ii}'^{n} - f_{ti}' = \sigma_{ii}'^{t} - E_{iikl}\Delta\varepsilon_{kl}'^{f} - f_{ti}' = 0 \tag{2.40}$$

This equation can be further simplified under the assumption that the increment of fracturing strain is normal to the failure surface, and that always only one failure surface is being checked. For failure surface k, the fracturing strain increment has the following form.

$$\Delta\varepsilon_{ij}'^{f} = \Delta\lambda \frac{\partial F_{k}^{f}}{\partial \sigma_{ij}} = \Delta\lambda\ \delta_{ik} \tag{2.41}$$

After substitution into (2.40) a formula for the increment of the fracturing multiplier λ is recovered.

$$\Delta\lambda = \frac{\sigma'^{t}_{kk} - f'_{tk}}{E_{kkkk}} = \frac{\sigma'^{t}_{kk} - f'_{t}(w_k^{\max})}{E_{kkkk}} \quad \text{and} \quad w_k^{\max} = L_t(\hat{\varepsilon}'^{f}_{kk} + \Delta\lambda) \tag{2.42}$$

This equation must be solved by iterations since for softening materials the value of current tensile strength $f'_t(w_k^{\max})$ is a function of the crack opening w, and is based on Hordijk's formula (defined in SBETA model).

The crack opening w is computed from the total value of fracturing strain $\hat{\varepsilon}'^{f}_{kk}$ in direction k, plus the current increment of fracturing strain $\Delta\lambda$, and this sum is multiplied by the characteristic length L_t. The characteristic length as a crack band size was introduced by BAZANT and OH. Various methods were proposed for the crack band size calculation in the framework of finite element method. FEENSTRA (1993) suggested a method based on integration point volume, which is not well suited for distorted elements. A consistent and rather complex approach was proposed by OLIVIER. In the presented work the crack band size L_t is calculated as a size of the element projected into the crack direction, Fig. 2-20. CERVENKA V. et al. (1995) showed that this approach is satisfactory for low order linear elements, which are used throughout this study. They also proposed a modification, which accounts for cracks that are not aligned with element edges.

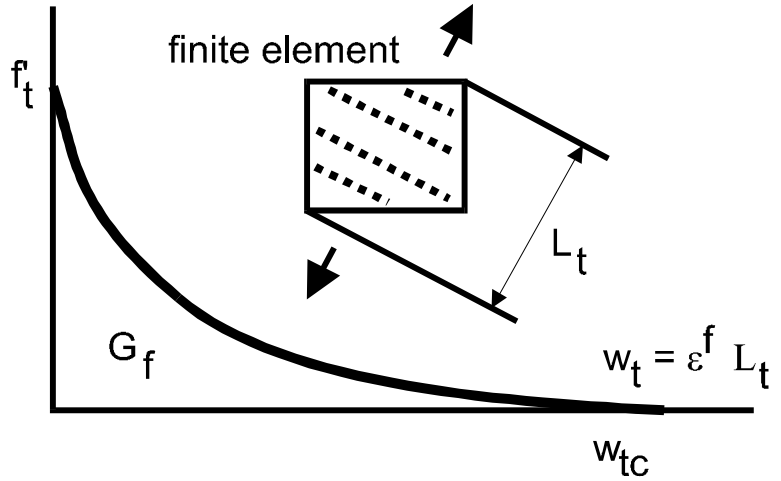


Fig. 2-20 Tensile softening and characteristic length

Equation (2.42) can be solved by recursive substitutions. It is possible to show by expanding $f'_t(w_k^{\max})$ into a Taylor series that this iteration scheme converges if:

$$\left| -\frac{\partial f'_t(w_k^{\max})}{\partial w} \right| < \frac{E_{kkkk}}{L_t} \tag{2.43}$$

Equation (2.43) is violated for softening materials only when snap back is observed in the stress-strain relationship, which can occur if large finite elements are used. In the standard displacement based finite element method, the strain increment is given, therefore, a snap back on the constitutive level cannot be captured. This means that the critical region, with snap back on the softening curve, will be skipped in a real calculation, which physically means, that the energy dissipated by the system will be over estimated. This is of course undesirable, and finite

elements smaller then $L < E_{kkkk} \Big/ \left| \frac{\partial f_t'(0)}{\partial w} \right|$ should be used, where $\frac{\partial f_t'(0)}{\partial w}$ denotes the initial slope of the crack softening curve.

It is important to distinguish between total fracturing strain $\hat{\varepsilon}_{ij}'^{f}$, which corresponds to the maximal fracturing strain reached during the loading process, and current fracturing strain $\varepsilon_{ij}'^{f}$, which can be smaller due to crack closure, and is computed using (2.44) derived by ROTS and BLAUWENDRAAD.

$$\varepsilon_{kl}'^{f} = (E_{ijkl} + E_{ijkl}'^{cr})^{-1} E_{klmn} \varepsilon_{mn}' \text{ , and } E_{ijlk}'^{cr} \text{ is defined by } \sigma_{ij}' = E_{ijkl}'^{cr} \varepsilon_{kl}'^{f} \tag{2.44}$$

The fourth order crack tensor $E_{ijkl}'^{cr}$ represents the cracking stiffness in the local material directions. In the current formulation, it is assumed, that there is no interaction between normal and shear components. Thus, the crack tensor is given by the following formulas.

$$E_{ijkl}'^{cr} = 0 \quad \text{for} \quad i \neq k \quad \text{and} \quad j \neq l \tag{2.45}$$

Mode I crack stiffness equals

$$E_{iiii}'^{cr} = \frac{f_t'(w_i^{\max})}{\hat{\varepsilon}_{ii}'^{f}}, \text{ (no summation of indices)} \tag{2.46}$$

and mode II and III crack stiffness is assumed as:

$$E_{ijij}'^{cr} = s_F \min\left(E_{iiii}'^{cr}, E_{jjjj}'^{cr}\right), \text{ (no summation of indices)} \tag{2.47}$$

where $i \neq j$, and s_F is a shear factor coefficient that defines a relationship between the normal and shear crack stiffness. The default value of s_F is 20.

Shear strength of a cracked concrete is calculated using the Modified Compression Field Theory of VECHIO and COLLINS (1986).

$$\sigma_{ij} \leq \frac{0.18\sqrt{f_c'}}{0.31 + \frac{24w}{a_g + 16}}, \quad i \neq j \tag{2.48}$$

Where f_c' is the compressive strength in MPa, a_g is the maximum aggregate size in mm and w is the maximum crack width in mm at the given location. This model is activated by specifying the maximum aggregate size a_g otherwise the default behavior is used where the shear stress on a crack surface cannot exceed the tensile strength.

The secant constitutive matrix in the material direction was formulated by ROTS and BLAUWENDRAAD in the matrix format.

$$\mathbf{E}'^{s} = \mathbf{E} - \mathbf{E}(\mathbf{E}'^{cr} + \mathbf{E})^{-1}\mathbf{E} \tag{2.49}$$

Strain vector transformation matrix $\mathbf{T}^{\varepsilon}$ (i.e. global to local strain transformation matrix) can be used to transform the local secant stiffness matrix to the global coordinate system.

$$\mathbf{E}^{s} = \mathbf{T}^{\varepsilon T}\mathbf{E}'^{s}\mathbf{T}^{\varepsilon} \tag{2.50}$$

It is necessary to handle the special cases before the onset of cracking, when the crack stiffness approaches infinity. Large penalty numbers are used for crack stiffness in these cases.

2.2.3.1 Unloading Direction

Crack closure stiffness is controlled by the unloading factor (material parameter) $0 \leq f_U < 1$.

The value of 0 corresponds to unloading to origin (default value for backward compatibility), f_U =1 means unloading direction parallel to the initial elastic stiffness.

2.2.4 Plasticity Model for Concrete Crushing

New stress state in the plastic model is computed using the predictor-corrector formula.

$$^{(n)}\sigma_{ij} = {}^{(n-1)}\sigma_{ij} + E_{ijkl}(\Delta\varepsilon_{kl} - \Delta\varepsilon_{kl}^{p}) = \sigma_{ij}^{t} - E_{ijkl}\Delta\varepsilon_{kl}^{p} = \sigma_{ij}^{t} - \sigma_{ij}^{p} \tag{2.51}$$

The plastic corrector σ_{ij}^{p} is computed directly from the yield function by return mapping algorithm.

$$F^{p}(\sigma_{ij}^{t} - \sigma_{ij}^{p}) = F^{p}(\sigma_{ij}^{t} - \Delta\lambda l_{ij}) = 0 \tag{2.52}$$

The crucial aspect is the definition of the return direction l_{ij}, which can be defined as

$$l_{ij} = E_{ijkl}\frac{\partial G^{p}(\sigma_{kl}^{t})}{\partial\sigma_{kl}} \quad \text{then} \quad \Delta\varepsilon_{ij}^{p} = \Delta\lambda\frac{\partial G^{p}(\sigma_{ij}^{t})}{\partial\sigma_{ij}} \tag{2.53}$$

where $G(\sigma_{ij})$ is the plastic potential function, whose derivative is evaluated at the predictor stress state σ_{ij}^{t} to determine the return direction.

The failure surface of MENETREY, WILLAM is used in the current version of the material model.

$$F_{3P}^{p} = \left[\sqrt{1.5}\frac{\rho}{f_c^{'}}\right]^{2} + m\left[\frac{\rho}{\sqrt{6}f_c^{'}}r(\theta,e) + \frac{\xi}{\sqrt{3}f_c^{'}}\right] - c = 0 \tag{2.54}$$

where

$$m = 3\frac{f_c^{'2} - f_t^{'2}}{f_c^{'}f_t^{'}}\frac{e}{e+1}, \quad r(\theta,e) = \frac{4(1-e^{2})\cos^{2}\theta + (2e-1)^{2}}{2(1-e^{2})\cos\theta + (2e-1)\left[4(1-e^{2})\cos^{2}\theta + 5e^{2} - 4e\right]^{\frac{1}{2}}}$$

In the above equations (ξ,ρ,θ) are Heigh-Vestergaard coordinates, f_c' and f_t' is compressive strength and tensile strength respectively. Parameter $e \in \langle 0.5,1.0\rangle$ defines the roundness of the failure surface. The failure surface has sharp corners if $e = 0.5$, and is fully circular around the hydrostatic axis if $e = 1.0$.

The position of failure surfaces is not fixed but it can move depending on the value of strain hardening/softening parameter. The strain hardening is based on the equivalent plastic strain, which is calculated according to the following formula.

$$\Delta\varepsilon_{eq}^{p} = \min(\Delta\varepsilon_{ij}^{p}) \tag{2.55}$$

For Menétrey-Willam surface the hardening/softening is controlled by the parameter $c \in \langle 0,1\rangle$, which evolves during the yielding/crushing process by the following relationship:

$$c = \left(\frac{f_c'(\varepsilon_{eq}^p)}{f_c'} \right)^2 \tag{2.56}$$

In the above two formulas the expression $f_c'(\varepsilon_{eq}^p)$ indicates the hardening/softening law, which is based on the uniaxial compressive test. The law is shown in Fig. 2-21, where the softening curve is linear, and the elliptical ascending part is given by the following formula:

$$\sigma = f_{co} + (f_c - f_{co}) \sqrt{1 - \left(\frac{\varepsilon_c - \varepsilon_{eq}^p}{\varepsilon_c} \right)^2} \tag{2.57}$$

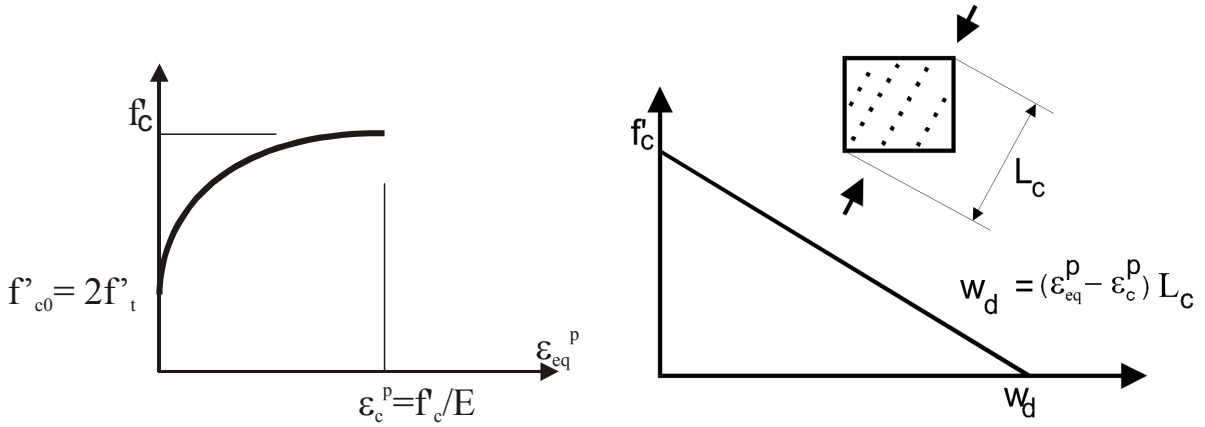


Fig. 2-21. Compressive hardening/softening and compressive characteristic length. Based on experimental observations by VAN MIER.

The law on the ascending branch is based on strains, while the descending branch is based on displacements to introduce mesh objectivity into the finite element solution, and its shape is based on the work of VAN MIER. The onset of nonlinear behavior $f_{c0}^{'}$ is an input parameter as well as the value of plastic strain at compressive strength ε_c^p. The Fig. 2-21 shows typical values of these parameters. In general case, however, ε_c^p should be calculated from the total strain at the peak by subtracting the elastic part $\varepsilon_c^p = \varepsilon_1 - \frac{f_c^{'}}{E}$, where ε_1 is the compressive strain when the compressive strength $f_c^{'}$ is reached. Especially the choice of the parameter $f_{c0}^{'}$ should be selected with care, since it is important to ensure that the fracture and plastic surfaces intersect each other in all material stages. On the descending curve the equivalent plastic strain is transformed into displacements through the length scale parameter L_c. This parameter is defined by analogy to the crack band parameter in the fracture model in Sec. 2.2.3, and it corresponds to the projection of element size into the direction of minimal principal stresses. The square in (2.56) is due to the quadratic nature of the Menétry-Willam surface.

Return direction is given by the following plastic potential

$$G^p(\sigma_{ij}) = \beta \frac{1}{\sqrt{3}} I_1 + \sqrt{2J_2} \tag{2.58}$$

where β determines the return direction. If $\beta < 0$ material is being compacted during crushing, if $\beta = 0$ material volume is preserved, and if $\beta > 0$ material is dilating. In general, the plastic model is non-associated, since the plastic flow is not perpendicular to the failure surface

The return mapping algorithm for the plastic model is based on predictor-corrector approach as is shown in Fig. 2-22. During the corrector phase of the algorithm the failure surface moves along the hydrostatic axis to simulate hardening and softening. The final failure surface has the apex located at the origin of the Haigh-Vestergaard coordinate system. Secant method-based Algorithm 1 is used to determine the stress on the surface, which satisfies the yield condition and also the hardening/softening law.

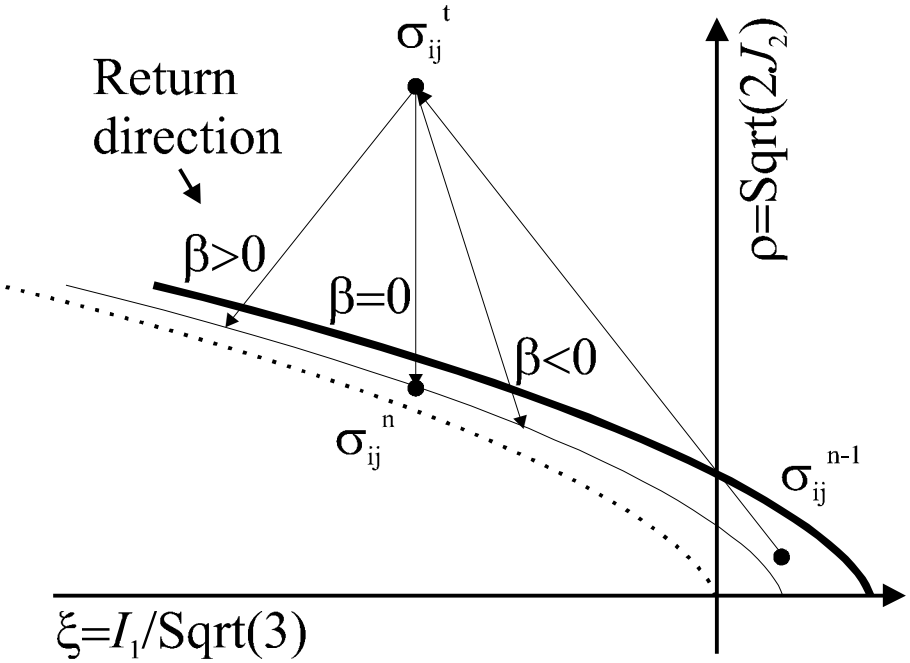


Fig. 2-22 Plastic predictor-corrector algorithm.

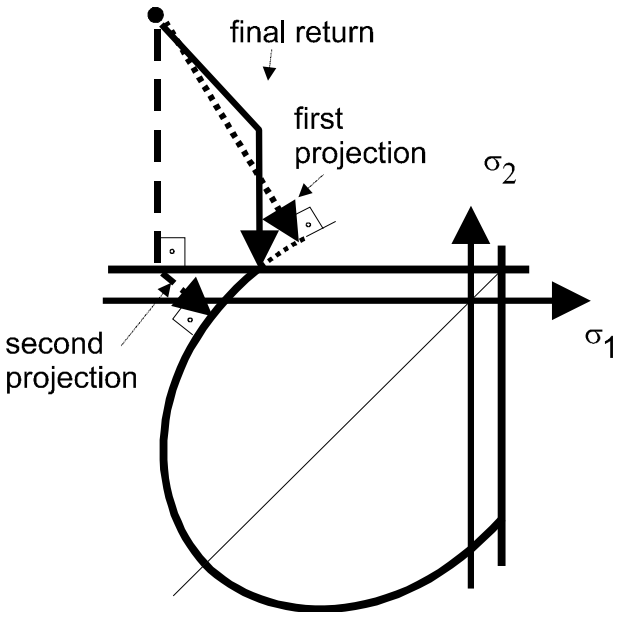


Fig. 2-23. Schematic description of the iterative process (2.73). For clarity shown in two dimensions.

Algorithm 1: (Input is ${}^{(n-1)}\sigma_{ij}, {}^{(n-1)}\varepsilon_{ij}^{p}, \Delta^{(n)}\varepsilon_{ij}$)

Elastic predictor: $$\sigma_{ij}^{t} = {}^{(n-1)}\sigma_{ij} + E_{ijkl}\ \Delta^{(n)}\varepsilon_{kl} \tag{2.59}$$

Evaluate failure criterion: $$f_A^p = F^p(\sigma_{ij}^t, {}^{(n-1)}\varepsilon_{ij}^p),\ \ \Delta\lambda_A = 0 \tag{2.60}$$

If failure criterion is violated i.e. $f_A^p > 0$

Evaluate return direction: $$m_{ij} = \frac{\partial G^p(\sigma_{ij}^t)}{\partial \sigma_{ij}} \tag{2.61}$$

Return mapping: $$F^p(\sigma_{ij}^t - \Delta\lambda_B E\, m_{ij}, {}^{(n-1)}\varepsilon_{ij}^p) = 0 \ \ \Rightarrow \ \ \Delta\lambda_B \tag{2.62}$$

Evaluate failure criterion: $$f_B^p = F^p(\sigma_{ij}^t - \Delta\lambda_B E\, m_{ij}, {}^{(n-1)}\varepsilon_{ij}^p + \Delta\lambda_B m_{ij}) \tag{2.63}$$

Secant iterations (i) as long as $$\left|\Delta\lambda_A - \Delta\lambda_B\right| > e \tag{2.64}$$

New plastic multiplier increment: $$\Delta\lambda = \Delta\lambda_A - f_A^p \frac{\Delta\lambda_B - \Delta\lambda_A}{f_B^p - f_A^p} \tag{2.65}$$

New return direction: $${}^{(i)}m_{ij} = \frac{\partial G^p(\sigma_{ij}^t - \Delta\lambda\, E\ {}^{(i-1)}m_{ij})}{\partial \sigma_{ij}} \tag{2.66}$$

Evaluate failure criterion: $$f^p = F^p(\sigma_{ij}^t - \Delta\lambda\, E\ {}^{(i)}m_{ij}, {}^{(n-1)}\varepsilon_{ij}^p + \Delta\lambda\ {}^{(i)}m_{ij}) \tag{2.67}$$

New initial values for secant iterations:

$$f_B^p < 0 \ \ \Rightarrow \ \ f_B^p = f^p, \ \ \Delta\lambda_B = \Delta\lambda \tag{2.68}$$

$$f_B^p \geq 0 \ \ \Rightarrow \ \ f_A^p = f_B^p, \ \ \Delta\lambda_A = \Delta\lambda_B, \ \ f_B^p = f^p, \ \ \Delta\lambda_B = \Delta\lambda \tag{2.69}$$

End of secant iteration loop

End of algorithm update stress and plastic strains.

$${}^{(n)}\varepsilon_{ij}^p = {}^{(n-1)}\varepsilon_{ij}^p + \Delta\lambda_B\, {}^{(i)}m_{ij}, \qquad {}^{(n)}\sigma_{ij} = \sigma_{ij}^t - \Delta\lambda_B E\, {}^{(i)}m_{ij} \tag{2.70}$$

2.2.5 Combination of Plasticity and Fracture model

The objective is to combine the above models into a single model such that plasticity is used for concrete crushing and the Rankine fracture model for cracking. This problem can be generally stated as a simultaneous solution of the two following inequalities.

$$F^p({}^{(n-1)}\sigma_{ij} + E_{ijkl}(\Delta\varepsilon_{kl} - \Delta\varepsilon_{kl}^f - \Delta\varepsilon_{kl}^p)) \leq 0 \quad \text{solve for } \Delta\varepsilon_{kl}^p \tag{2.71}$$

$$F^f({}^{(n-1)}\sigma_{ij} + E_{ijkl}(\Delta\varepsilon_{kl} - \Delta\varepsilon_{kl}^p - \Delta\varepsilon_{kl}^f)) \leq 0 \quad \text{solve for } \Delta\varepsilon_{kl}^f \tag{2.72}$$

Each inequality depends on the output from the other one, therefore the following iterative scheme is developed.

Algorithm 2:

Step 1: $F^p({}^{(n-1)}\sigma_{ij} + E_{ijkl}(\Delta\varepsilon_{kl} - \Delta^{(i-1)}\varepsilon_{kl}^f + b\Delta^{(i-1)}\varepsilon_{kl}^{cor} - \Delta^{(i)}\varepsilon_{kl}^p)) \le 0$ solve for $\Delta^{(i)}\varepsilon_{kl}^p$

Step 2: $F^f({}^{(n-1)}\sigma_{ij} + E_{ijkl}(\Delta\varepsilon_{kl} - \Delta^{(i)}\varepsilon_{kl}^p - \Delta^{(i)}\varepsilon_{kl}^f)) \le 0$ solve for $\Delta^{(i)}\varepsilon_{kl}^f$

Step 3: $\Delta^{(i)}\varepsilon_{ij}^{cor} = \Delta^{(i)}\varepsilon_{ij}^f - \Delta^{(i)}\varepsilon_{ij}^f$ (2.73)

Iterative correction of the strain norm between two subsequent iterations can be expressed as

$$\left\|\Delta^{(i)}\varepsilon_{ij}^{cor}\right\| = (1-b)\,\alpha^f\alpha^p\left\|\Delta^{(i)}\varepsilon_{ij}^{cor}\right\| \tag{2.74}$$

$$\text{where } \alpha^f = \frac{\left\|\Delta^{(i)}\varepsilon_{ij}^f - \Delta^{(i-1)}\varepsilon_{ij}^f\right\|}{\left\|\Delta^{(i)}\varepsilon_{ij}^p - \Delta^{(i-1)}\varepsilon_{ij}^p\right\|},\ \alpha^p = \frac{\left\|\Delta^{(i)}\varepsilon_{ij}^p - \Delta^{(i)}\varepsilon_{ij}^p\right\|}{\left\|\Delta\varepsilon_{ij}^{cor}\right\|}$$

and b is an iteration correction or relaxation factor, which is introduced to guarantee convergence. It is to be determined based on the run-time analysis of α^f and α^p, such that the convergence of the iterative scheme can be assured. The parameters α^f and α^p characterize the mapping properties of each model (i.e. plastic and fracture). It is possible to consider each model as an operator, which maps strain increment on the input into a fracture or plastic strain increment on the output. The product of the two mappings must be contractive to obtain a convergence. The necessary condition for the convergence is:

$$\left|(1-b)\alpha^f\alpha^p\right| < 1 \tag{2.75}$$

If b equals 0, an iterative algorithm based on recursive substitution is obtained. The convergence can be guaranteed only in two cases:

One of the models is not activated (i.e. implies α^f or $\alpha^p = 0$),

There is no softening in either of the two models and dilating material is not used in the plastic part, which for the plastic potential in this work means $\beta \le 0$, (2.58). This is a sufficient but not necessary condition to ensure that α^f and $\alpha^p < 1$.

It can be shown that the values of α^f and α^p are directly proportional to the softening rate in each model. Since the softening model remains usually constant for a material model and finite element, their values do not change significantly between iterations. It is possible to select the scalar b such that the inequality (2.75) is satisfied always at the end of each iteration based on the current values of α^f and α^p. There are three possible scenarios, which must be handled, for the appropriate calculation of b:

$\left|\alpha^f\alpha^p\right| \le \chi$, where χ is related to the requested convergence rate. For linear rate it can be set to $\chi = 1/2$. In this case the convergence is satisfactory and $b = 0$.

$\chi < \left|\alpha^f\alpha^p\right| < 1$, then the convergence would be too slow. In this case b can be estimated as $b = 1 - \dfrac{\left|\alpha^f\alpha^p\right|}{\chi}$, in order to increase the convergence rate.

$1 \le \left|\alpha^f\alpha^p\right|$, then the algorithm is diverging. In this case b should be calculated as $b = 1 - \dfrac{\chi}{\left|\alpha^f\alpha^p\right|}$ to stabilize the iterations.

This approach guarantees convergence as long as the parameters α^p, α^f do not change drastically between the iterations, which should be satisfied for smooth and correctly formulated models. The rate of convergence depends on material brittleness, dilating parameter β and finite element size. It is advantageous to further stabilize the algorithm by smoothing the parameter b during the iterative process:

$$b = ({}^{(i)}b + {}^{(i-1)}b)/2 \tag{2.76}$$

where the superscript i denotes values from two subsequent iterations. This will eliminate problems due to the oscillation of the correction parameter b. Important condition for the convergence of the above Algorithm 2 is that the failure surfaces of the two models are intersecting each other in all possible positions even during the hardening or softening.

Additional constraints are used in the iterative algorithm. If the stress state at the end of the first step violates the Rankine criterion, the order of the first two steps in Algorithm 2 is reversed. Also, concrete crushing in one direction influences the cracking in other directions. It is assumed that after the plasticity yield criterion is violated, the tensile strength in all material directions is set to zero.

On the structural level secant matrix is used to achieve a robust convergence during the strain localization process.

The proposed algorithm for the combination of plastic and fracture models is graphically shown in Fig. 2-23. When both surfaces are activated, the behavior is quite like the multi-surface plasticity (SIMO et al. 1988). Contrary to the multi-surface plasticity algorithm the proposed method is more general in the sense that it covers all loading regimes including physical changes such as for instance crack closure. Currently, it is developed only for two interacting models, and its extension to multiple models is not straightforward.

There are additional interactions between the two models that need to be considered to properly describe the behavior of a concrete material:

(a) After concrete crushing the tensile strength should decrease as well

(b) According to the research work of Collins (VECHIO and COLLINS (1986)) and coworkers it was established the also compressive strength should decrease when cracking occurs in the perpendicular direction. This theory is called compression field theory and it is used to explain the shear failure of concrete beams and walls.

The interaction (a) is resolved by adding the equivalent plastic strain to the maximal fracturing strain in the fracture model to automatically increase the tensile damage based on the compressive damage such that the fracturing strains satisfies the following condition:

$$\hat{\varepsilon}'^{f}_{kk} \geq \frac{f_t'}{f_c'} \varepsilon^p_{eq} \tag{2.77}$$

The compressive strength reduction (b) is based on the following formula based proposed by Collins:

$$\sigma_c = r_c f_c'$$

$$r_c = \frac{1}{0.8 + 170\,\varepsilon_1}, \; r_c^{\lim} \leq r_c \leq 1.0 \tag{2.78}$$

Where ε_1 is the tensile strain in the crack. In ATENA the largest maximal fracturing strain is used for ε_1 and the compressive strength reduction is limited by $r_c^{\lim}$. If $r_c^{\lim}$ is not specified, then no compression reduction is considered.

2.2.6 Variants of the Fracture Plastic Model

The several ATENA material models are based on the above theories:

CC3DCementitious,

CC3DNonLinCementitious,

CC3DNonLinCementitious2,

CC3DNonLinCementitious2Variable,

CC3DNonLinCementitious2Fatigue (described in section 2.2.10),

CC3DNonLinCementitious2User,

CC3DNonLinCementitious2FRC (described in section 2.2.11),

CC3DNONLINCEMENTITIOUS2SHCC,

CC3DNONLINCEMENTITIOUS2HPFRC (described in section 2.2.12),

and *CC3DNonLinCementitious3* (described in section 2.2.13),

with the following differences: *CC3DCementitious* assumes linear response up to the point when the failure envelope is reached both in tension and compression. This means that there is no hardening regime in Fig. 2-21. The material *CC3DNonLinCementitious* on the contrary assumes a hardening regime before the compressive strength is reached. The material *CC3DNonLinCementitious2* is equivalent to *CC3DNonLinCementitious* but purely incremental formulation is used (in *CC3DNonLinCementitious* a total formulation is used for the fracturing part of the model), therefore this material can be used in creep calculations or when it is necessary to change material properties during the analysis. The material *CC3DNonLinCementitious2Variable* is based on the material *CC3DNonLinCementitious2* and it allows to define history evolution laws for selected material parameters. The following material parameters can be defined using an arbitrary evolution laws: young modulus E, tensile strength $f_t^{'}$, compressive strength $f_c^{'}$ and $f_{c0}^{'}$. It is the responsibility of the user to define the parameters in a meaningful way. It means that at any time (please note compressive strength parameters $f_c^{'}$ and $f_{c0}^{'}$ are defined as negative values in ATENA):

$$f_t^{'} \le \tfrac{1}{2}\left|f_{c0}^{'}\right| \tag{2.79}$$

$$f_{c0}^{'} < f_{c0}^{'}, \quad f_{c0}^{'} < 0 \tag{2.80}$$

The material *CC3DNonLinCementitious2User* allows for user defined laws for selected material laws such as: diagrams for tensile and softening behavior (see Fig. 2-24 and Fig. 2-25), shear retention factor (Fig. 2-26) and the effect of lateral compression on tensile strength (Fig. 2-27).

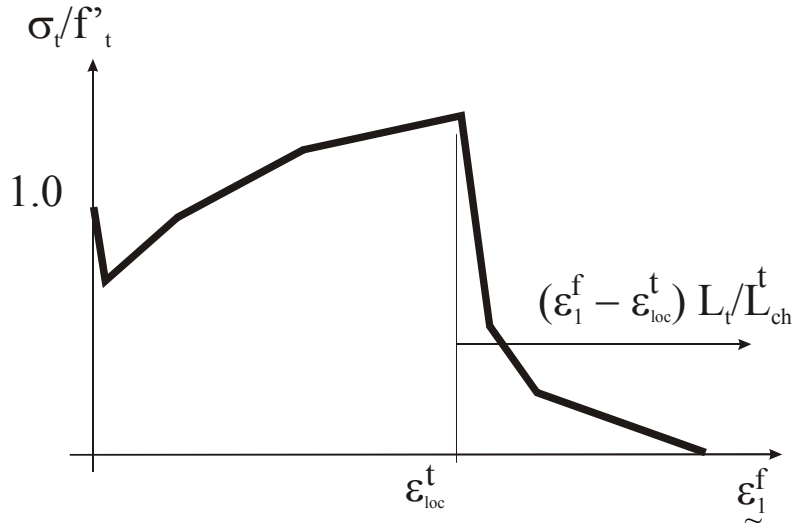


Fig. 2-24. An example of a user defined tensile behavior for CC3DNonLinCementitious2User material.

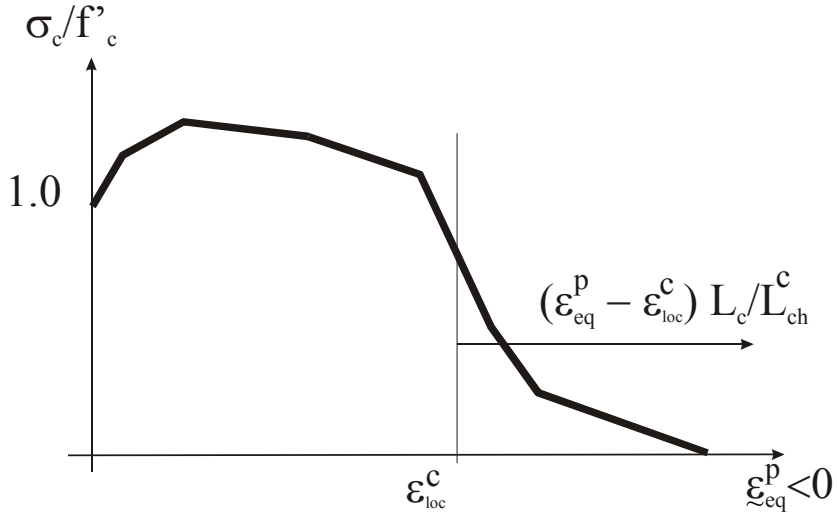


Fig. 2-25. An example of a user defined compressive behavior for CC3DNonLinCementitious2User material.

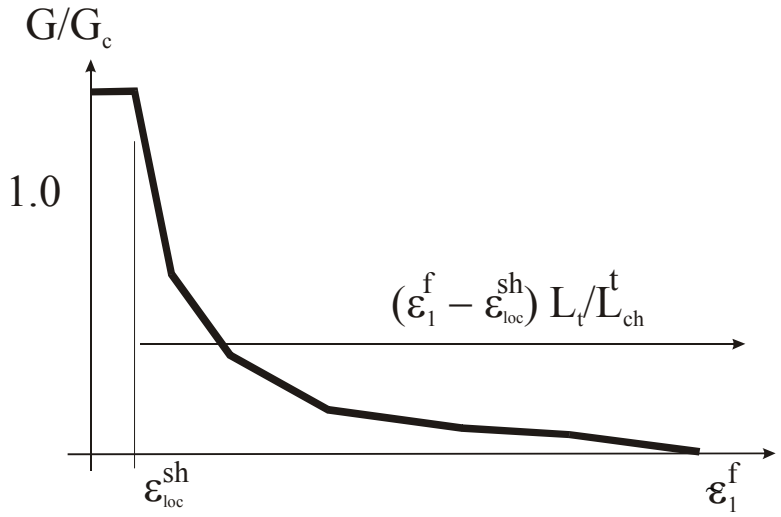


Fig. 2-26. An example of a user defined shear retention factor for shear stiffness degradation after cracking.

In the user defined material mode II and III crack stiffness are evaluated with the help of the shear retention factor r_g as:

$$E'^{cr}_{ijij} = \frac{r_g\, G}{1 - r_g} \tag{2.81}$$

where $i \neq j$, $r_g = \min(r_g^i, r_g^j)$ is the minimum of shear retention factors on cracks in directions i, j, and G is the elastic shear modulus. Shear retention factor on a crack in direction i is evaluated from the user specified diagram as shown in Fig. 2-26.

In the above diagrams L_t and L_c represents the crack band size and crush band size respectively as it is defined Section 2.1.3. L^t_{ch} and L^c_{ch} represents a size for which the tensile and compression diagram respectively is valid. For instance, it represents the measuring base that was used in an experiment to determine the strain values in the diagrams above. ε_{loc} represents the strain value, after which strain localization can be expected. Usually, this is the strain after which the diagram is entering into the softening regime. For instance, the strain value that is used to determine the tensile strength is calculated based on the following assumptions:

if $\varepsilon_1^f < \varepsilon_{loc}^f$

$$\tilde{\varepsilon}_1^f = \varepsilon_1^f$$

else

$$\tilde{\varepsilon}_1^f = \varepsilon_{loc}^f + (\varepsilon_1^f - \varepsilon_{loc}^f) \Big/ \frac{L_t}{L^t_{ch}} \tag{2.82}$$

The calculation of the strain value for graphs in Fig. 2-25 and Fig. 2-26 is analogical to Eq. (2.82) but the appropriate values of ε_{loc}, L and L_{ch} should be used. It should be noted that the strain ε_1^f is the strain that is calculated from the strain tensor at the finite element integration points, while the strain $\tilde{\varepsilon}_1^f$ is used to determine the current tensile strength from the provided stress-strain diagram (see Fig. 2-24). The equation (2.82) then represents a scaling that considers the difference between the experimental size and the size of the integration point. This approach guarantees that the same amount of energy is dissipated when using large and small finite elements.

It is also possible to define a material law for the shear strength of a cracked concrete and for the compressive strength reduction after cracking.

Compressive strength of cracked concrete $\qquad \sigma_c = r_c(\varepsilon_1^f) f_c' \qquad (2.83)$

Shear strength of cracked concrete $\qquad \sigma_{ij} \leq f_{sh}(\tilde{\varepsilon}_1^f) f_t' \qquad (2.84)$

It should be realized that the compressive strength of the cracked concrete i.e. (2.83) is a function of the maximal fracturing strain, i.e. maximal tensile damage at the given point. The shear strength should be a function of the crack opening. Because of that the shear strength is specified as a function of the fracturing strain $\tilde{\varepsilon}_1^f$ after the localization transformation (2.82). The shear strength law is specified as a value relative to f_t'. The compressive strength reduction is specified as a function relative to f_c'.

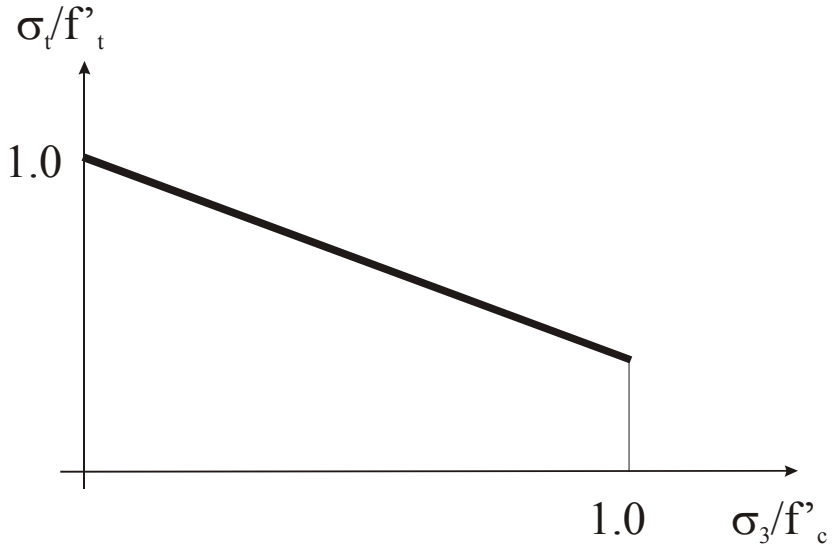


Fig. 2-27. An example of a user defined tensile strength degradation law due to lateral compressive stress.

2.2.7 Tension Stiffening

In heavily reinforced concrete structures, the cracks cannot fully developed and concrete contributes to the steel stiffness. This effect is called tension stiffening and in CC3DNonLinCementitious2 material it can be simulated by specifying a tension stiffening factor c_{ts}. This factor represents the relative limiting value of tensile strength in the tension softening diagram. The tensile stress cannot drop below the value given by the product of $c_{ts} f_t$ (see Fig. 2-28). The recommended default value for c_{ts} is 0.4 as recommended by CEB-FIP Model Code 1990.

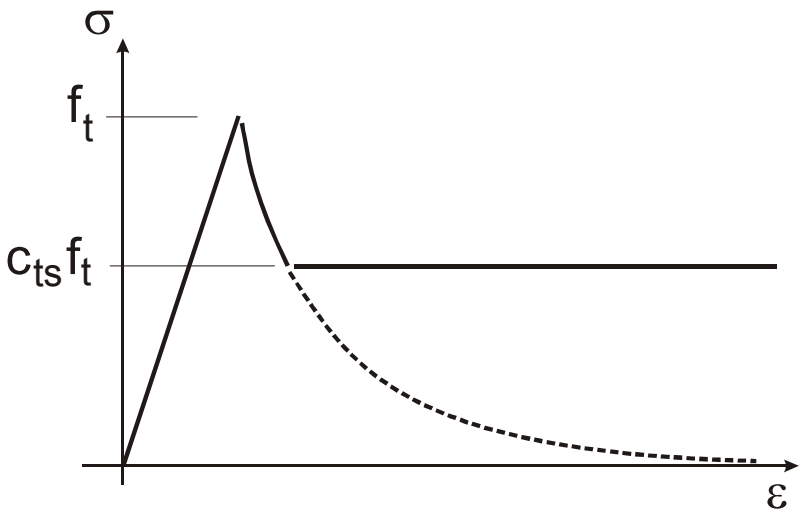


Fig. 2-28: Tension stiffening.

2.2.8 Crack Spacing

In heavily reinforced concrete structures, or structures with large finite elements, when many reinforcement bars are crossing each finite element, the crack band approach described in Section 2.1.3 will provide too conservative results, and the calculated crack widths may be

overestimated. This is the consequence of the fact that the crack band approach assumes that the crack spacing is larger than a finite element size. In heavily reinforced structures, or if large finite elements are used, it may occur that the crack spacing will be smaller than finite element size. This is especially true if shell/plate elements are used. In this case, typically large finite elements can be used, and they usually contain significant reinforcement. In these cases, it is useful to provide the crack spacing manually, since otherwise the program will overestimate the cracking and due to that also larger deflections may be calculated. The program ATENA allows the user to manually define the crack spacing. This user defined spacing is used as crack band size L_t in cases when the user defined crack spacing is smaller than the L_t that would be calculated by formulas presented in Section 2.1.3.

2.2.9 Fixed or Rotated Cracks

Similarly, to the SBETA material, the Cementitious material family offers the choice of fixed and rotated crack models (see section 2.1.6). The fixed crack material parameter determines at which maximum residual tensile stress level the crack direction gets fixed. In other words, 0.0 means fully rotated crack model (as 0 in SBETA), 1.0 means fixed crack model (as 1 in SBETA), values between 0.0 and 1.0 determine the crack direction locking level, e.g., 0.7 fixes the crack direction as soon it opens so far that the softening law drops to 0.7 times the [initial] tensile strength.

2.2.10 Fatigue

For modelling fatigue behavior of concrete (CEB 1988 and SAE AE-4) under tensile load, a new material has been implemented in ATENA. The new material (*CC3DNonLinCementitious2Fatigue*) is based on the existing three-dimensional fracture plastic material (*CC3DNonLinCementitious2*) and uses a stress-based model (2.2.10.1). It has an additional parameter, $\beta_{fatigue}$, and additional data attributes for σ_{base}, N, and $\varepsilon_{fatigue}$, used in the damage calculation as described in section 2.2.10.2. For details and validation against tests conducted by KESSLER-KRAMER (2002) see ČERVENKA, PRYL (2007) or PRYL, CERVENKA, PUKL (2010). Modelling 3-point bending tests with this material is presented in PRYL, PUKL, CERVENKA (2013) and PRYL, D., MIKOLÁŠKOVÁ, J., PUKL, R. (2014).

2.2.10.1 Stress Based Models

In this approach the fatigue is represented by the so-called S-N curves relating the applied stress, S, and the number of cycles, N, to failure. Such curves must be determined by tests, see Fig. 2-29.

For steel reinforcement bars the performance can be normally expressed as a simple power law by BASQUIN (1910).

$$\Delta\sigma_r^m N = C \tag{2.85}$$

where $\Delta\sigma_r$ is the stress range, N is the number of cycles to failure and m and C are constants. This means a linear relationship between $\Delta\sigma$ and N in a full logarithmic diagram. The equation (2.85) is generally valid for the high-cycle range.

For plain concrete, the performance can normally be expressed as a straight line in a semi-logarithmic diagram of the form:

$$\frac{\sigma_{\max}}{f} = 1 - \beta\left(1 - R\right)\log N \tag{2.86}$$

where σ_{max} is the maximum stress, f is static concrete strength, $R = \frac{\sigma_{min}}{\sigma_{max}}$, σ_{min} is minimum stress and β is a material constant. The equation (2.86) holds for both compressive and tensile stresses, however, the value of β is not necessarily the same for tensile and compressive behavior of a material. The value should be determined from experiments. For example, β=0.052 was used based on the experimental results for load levels 0.7 and 0.9 F_{stat} when modelling the test on a probe sealed during curing with a notch from section 3.5.2.4 of KESSLER-KRAMER (2002) for validation.

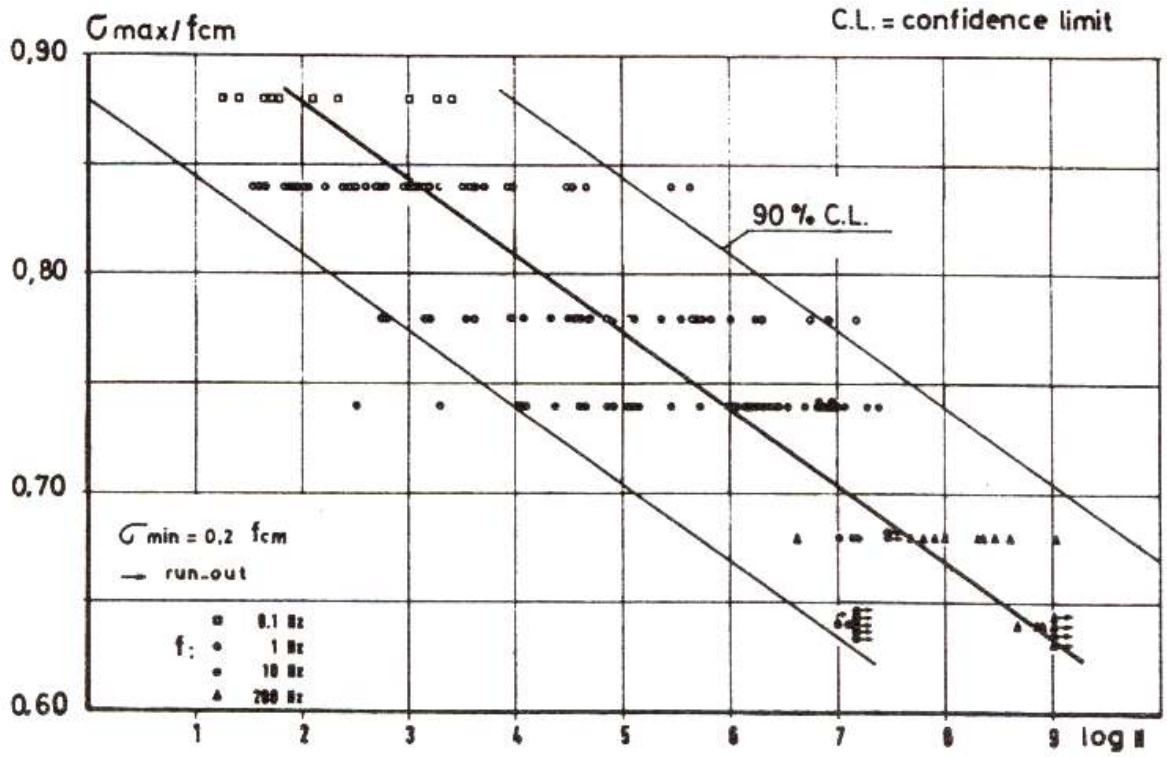


Fig. 2-29: Typical S-N line for concrete in compression (KLAUSEN (1978))

The S-N relations mentioned above are mainly obtained by constant amplitude tests. However, in real structures the stresses are varying. One method which can be of help in this context is the well-known Palmgren-Miner hypothesis PALMGREN (1924), MINER (1945).

$$\sum_{i=1}^{k} \frac{n_i}{N_i} = 1 \tag{2.87}$$

where n_i is the number of constant amplitude cycles at stress level i, N_i is the number of cycles to failure at stress level i, and k is the number of stress levels. As a rough tool this hypothesis is useful, especially concerning steel. It can also be used for concrete although some investigations have suggested that a value lower than 1 should be used.

2.2.10.2 Fatigue Damage Calculation

In the implemented model, fatigue damage consists of a contribution based on cyclic stress (2.2.10.2.1), and an additional contribution from crack opening and closing in each cycle (2.2.10.2.3). The former is dominant before cracking occurs, the latter in already cracked regions.

2.2.10.2.1 Stress Based Contribution

The number of cycles to failure N is determined from a simple stress based model, so called S-N or Wöhler curve as described in the previous section 2.2.10.1.

$\frac{\sigma_{upper}}{f} = 1 - \beta_{fatigue}(1-R)\log N$, i.e., $N = 10^{\left(\frac{1-\frac{\sigma_{upper}}{f}}{\beta_{fatigue}(1-R)}\right)}$, where σ_{upper} stands for the maximum tensile or compressive stress and f for the corresponding strength, f_t or f_c, $R = \frac{\sigma_{base}}{\sigma_{upper}}$.

Then, the damage due to fatigue after n cycles is calculated as an increase of the maximum fracturing strain $\hat{\varepsilon}_{ij}^{\prime f}$ (see section 2.2.3). The maximum fracturing strain in each principal direction is adjusted by adding

$\varepsilon_{fatigue} = \frac{w_{fatigue}}{ElemSize}$, where $w_{fatigue} = \frac{n}{N} w_{fail}$ and the failing displacement for the given stress $w_{fail} = invert\_soft\_law(\sigma_{upper})$ (see Fig. 2-30).

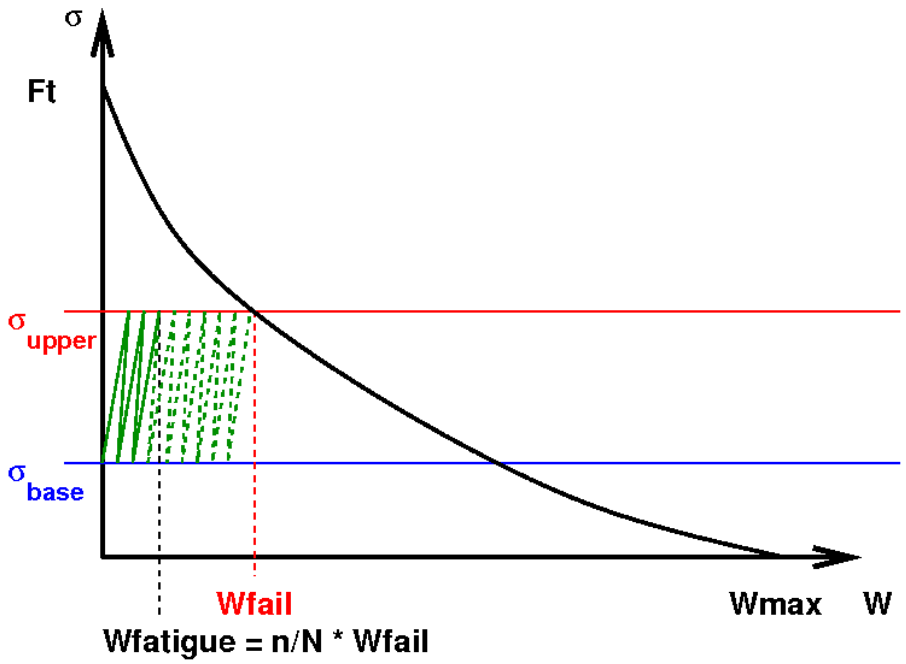


Fig. 2-30: Softening law and fatigue damage.

In ATENA 4.0, a single value of $\beta_{fatigue}$ is used to calculate fatigue damage caused by both tensile and compressive stresses. So far, there is also no special provision implemented for loads crossing zero, i.e., changing from tension to compression and back in each cycle, which lead to faster damage according to experimental results presented in CEB 1988 and SAE AE-4. In that situation, the damage is calculated separately for cyclic loading from 0 to max. compression and from 0 to max. tension, and then the worse of both damage values is considered. It should be also noted that the damage is only introduced in form of maximum fracturing strain, which has no direct impact on compressive material properties, i.e., the fatigue damage effectively only has influence on tensile behaviour of the material.

2.2.10.2.2 Stress Based Contribution with Trilinear Damage [1]

Hardcoded definition of damage evolution during the fatigue process, with the breakpoints

wf1 = wfr_1 * w_{fail} and wf2 = wfr_2 * w_{fail}.

$w_{fatigue}$ =

n * wf1 / N1	for n_tot < N1
wf1 + ((n_tot - N1) * (wf2 - wf1) / (N2 - N1)) - wf_curr	for N1 <= n_tot < N2
wf2 + ((n_tot - N2) * (w_{fail} - wf2) / (N - N2)) - wf_curr	for N2 <= n_tot < N
w_{fail} * n_tot / N - wf_curr	for N <= n_tot

where

n_tot = n + N_beg, N_curr = N - N_beg,

N_beg =

wf_curr * N1 / wf1	for wf_curr < wf1
N1 + (wf_curr - wf1) * (N2 - N1)/(wf2 - wf1)	for wf1 <= wf_curr < wf2
N2 + (wf_curr - wf2) * (N - N2)/(w_{fail} - wf2)	for wf2 <= wf_curr

0 < N_beg < N

and

wfr_1 = 0.1, Nr_1 = 0.1, wfr_2 = 0.5, Nr_2 = 0.9, N1 = Nr_1 * N, N2 = Nr_2 * N.

2.2.10.2.3 Crack Opening Based Contribution

The damage due to cracks that open and close during the cyclic loading is determined as $\varepsilon_{fatigueCOD} = \frac{w_{fatigueCOD}}{ElemSize}$, where $w_{fatigueCOD} = n\,\xi_{fatigue} / R_{COD}\, c_{fatigueCODload}\, \Delta COD$ [2], R_{COD} is the crack opening ratio (similar to the cycle asymmetry ratio R used in the stress based contribution; with a bottom limit of 0.01), and ΔCOD denotes the difference between the maximum and minimum crack opening during a cycle. The resulting $\varepsilon_{fatigueCOD}$ is added to $\varepsilon_{fatigue}$ before the fatigue damage is introduced into the material.

[1] Available since version 5.3.0

[2] In ATENA versions prior to 5.1.3 and 5.3.4: $w_{fatigueCOD} = n\,\xi_{fatigue}\, c_{fatigueCODload}\, \Delta COD$

2.2.10.3 Bringing in Fatigue Damage

It is recommended to introduce the fatigue induced damage into the unloaded structure (i.e., at the lower stress level). Several other approaches of introducing the damage into the model were also tested, i.e., introducing the damage at the upper load level or during reloading, but they usually bring more convergence problems, especially during unloading.

2.2.11 Fiber Reinforced Concrete (FRC) Material

The CC3DNonLinCementitious2FRC material model is based on CC3DNonLinCementitious2 as described above in Sections 2.2.1 - 2.2.6. In case of FRC, the fibers added to the concrete mixture increase the residual strength and ductility of the material, which is reflected by the tension softening law. In the FRC material model, the added fractural energy approach proposed by Juhász (2013) is implemented in the stress-crack width diagram. The total fractural energy of the fiber reinforced concrete reads:

$$G_{FFRC} = G_F + G_{Ff} \quad , G_{FFRC} - G_F + G_{Ff}$$

where G_{FFRC} and G_F, are the fractural energies of the fiber reinforced concrete and the plain concrete matrix, respectively, and G_{Ff} is the additional fractural energy, which corresponds to the pull-out energy of the fibers.

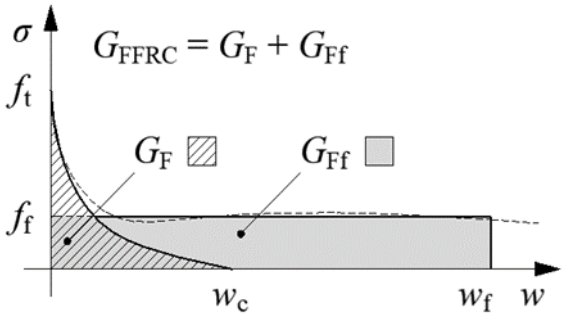


Fig. 2-31: Crack opening law for FRC using the added fractural energy approach.

The fracture energy added by the fibers is assumed as:

$$G_{Ff} = w_f \cdot f_t \ ,$$

where w_f is the maximum crack opening width of the FRC, which depends on the type and length of the fibers, and f_t is the post-cracking residual tension strength. It should be noted that the value of f_t defined as f_{Ftu} in the fib model code 2010 (Taerwe and Matthys, 2013).

2.2.12 Strain Hardening Cementitious Composite (SHCC, HPFRCC) Material

The CC*3DNONLINCEMENTITIOUS2SHCC* is suitable for fibre reinforced concrete, such as SHCC (Strain Hardening Cementitious Composites) and HPFRCC or UHPFRC (high and ultra-high performance fiber reinforced concrete) materials. The theory of this material model is identical to those described in Sections 2.2.1 - 2.2.6. The tensile softening regime (Fig. 2-33) and the shear retention factor (Eq. (2.94)) are modified based on the model, proposed in KABELE, P. (2002). This model is based on a notion of a representative volume element (RVE), which contains distributed multiple cracks (hardening) as well as localized cracks (softening) – see Fig. 2-32.

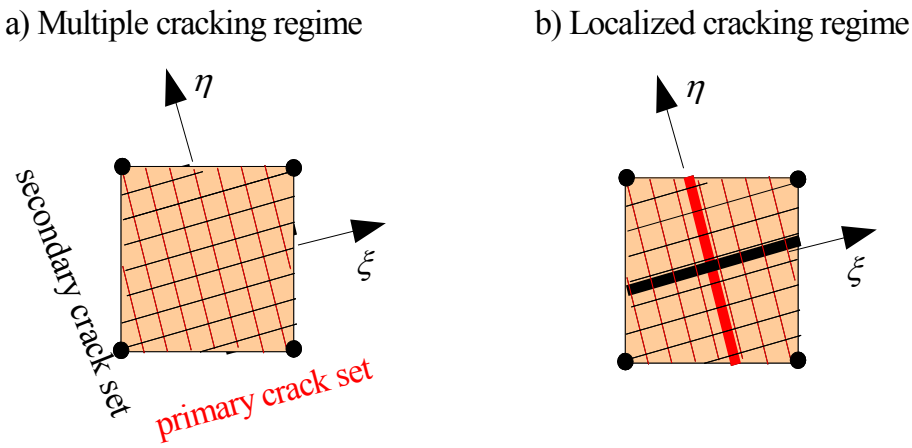


Fig. 2-32: Representative volume element with cracks.

2.2.12.1 Basic Assumptions

a) multiple cracking regime (hardening)

- A set of parallel planar multiple cracks forms when maximum principal stress $\sigma_{max} = \sigma_{fc}$ (first crack strength).
- Crack planes are perpendicular to the direction of σ_{max} (ξ-axis).
- The direction of a crack set is fixed.
- Secondary crack set may form in direction perpendicular to primary set if the maximum normal stress in the corresponding direction (η-axis) exceeds σ_{fc}.
- Cracks may slide if the direction of principal stress changes.
- Crack opening and sliding are resisted by fiber bridging.
- Crack opening and sliding displacements are averaged over the RVE as cracking strains $\varepsilon_{ij}^{mc,\xi}, \varepsilon_{ij}^{mc,\eta}$ (notation: lower indices – components of tensor or vector, upper indices – multiple or localized crack *mc*, *lc* and association with primary or secondary crack direction ξ, η)

b) localized cracking regime (softening)

- A localized crack forms within a set of multiple cracks if the corresponding normal cracking strain exceeds the level of ε^{mc}_{mb} (cracking strain capacity, a material constant).
- Opening and sliding displacements of the $\Delta_i^{\xi}, \Delta_i^{\eta}$ localized cracks are treated by the crack band model (i.e. they are transformed into cracking strains $\varepsilon_{ij}^{lc,\xi}, \varepsilon_{ij}^{lc,\eta}$ by dividing them with corresponding band width w^{ξ}_c or w^{η}_c).

The overall strain of the RVE is then obtained as a sum of strain of material between cracks (which may possibly contain nonlinear plastic strain due to compressive yielding), cracking strains due to multiple cracks, and cracking strains due to localized cracks:

$$\varepsilon_{ij} = \varepsilon_{ij}^{s} + \varepsilon_{ij}^{mc,\xi} + \varepsilon_{ij}^{mc,\eta} + \varepsilon_{ij}^{lc,\xi} + \varepsilon_{ij}^{lc,\eta} \tag{2.88}$$

where ε_{ij}^{s} represents the strain of the continuous material between cracks.

2.2.12.2 Crack Opening Model

The crack-normal stress components are related to cracking strains corresponding to opening of multiple and localized cracks by piecewise linear relations depicted in Fig. 2-33 [although linear hardening and softening are shown, a user should be allowed to input piecewise linear curves]. Note that for multiple cracks, it is assumed that they do not close unless exposed to crack-normal compression (plasticity-like unloading) while a localized crack is assumed to close so that normal stress decreases linearly to reach zero at zero COD [these assumptions may need to be revised in the future to some combination of plasticity and damage-like closure]. See also section 2.2.3.

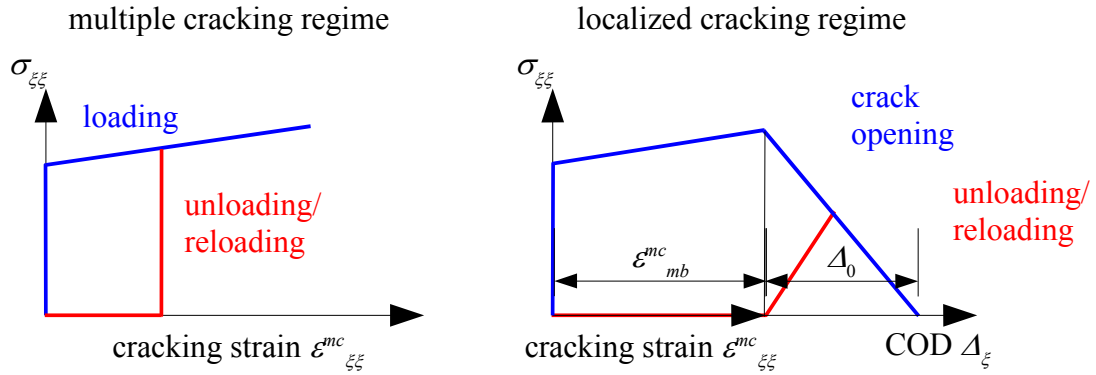


Fig. 2-33: Stress vs. cracking strain relations in crack-normal direction.

2.2.12.3 Crack Sliding Model

The model for crack sliding phenomena is implemented by means of a variable shear retention factor β. The shear retention factor is defined as a ratio of the material post-cracking shear stiffness G^c to its elastic shear stiffness G,

$$\beta = \frac{G^c}{G}. \tag{2.89}$$

Let us determine stiffness G^c, while considering the most general 2-D case of an element, which contains two perpendicular sets of multiple cracks and two perpendicular localized cracks. If the problem is defined in plane ξ–η, then the total engineering shear strain has only one non-zero component, which is obtained as:

$$\gamma_{\xi\eta} = 2\epsilon^{s}_{\xi\eta} + 2\epsilon^{mc,\xi}_{\xi\eta} + 2\epsilon^{mc,\eta}_{\xi\eta} + 2\epsilon^{lc,\xi}_{\xi\eta} + 2\epsilon^{lc,\eta}_{\xi\eta}, \tag{2.90}$$

which can be rewritten with use of the shear bridging model (Kabele, 2000) as:

$$\gamma_{\xi\eta} = \left[\frac{1}{G} + \frac{1}{M(\epsilon^{mc,\xi}_{\xi\xi})} + \frac{1}{M(\epsilon^{mc,\eta}_{\eta\eta})} + \frac{1}{w^{\xi}_{c} L(\Delta^{\xi}_{\xi})} + \frac{1}{w^{\eta}_{c} L(\Delta^{\eta}_{\eta})}\right]\sigma_{\xi\eta} = \frac{1}{G^c}\sigma_{\xi\eta} \tag{2.91}$$

Functions M and L are defined by

$$M(\varepsilon) = \frac{V_f\, k\, G_f}{2\varepsilon} \tag{2.92}$$

$$L(\Delta)=\left(\frac{1}{2}-\frac{\Delta}{2\Delta_0}\right)\frac{V_f\,k\,G_f}{\Delta\left[1+\dfrac{4\,k\,G_f}{3\,E_f}\left(\dfrac{\Delta}{d_f}\right)^2\right]},\ \text{for}\ \ \Delta\le\Delta_0$$

$$L(\Delta)=0\,,\ \text{for}\ \ \Delta>\Delta_0 \tag{2.93}$$

Here V_f is the fiber volume fraction, G_f is the fiber shear modulus, E_f is the fiber Young's modulus, d_f is the fiber diameter, and k is the fiber cross-section shape correction factor. The quantity Δ_ξ and Δ_η indicates the crack opening in direction ξ and η respectively. The parameter Δ_0 represents the limiting value of the crack opening displacement, when no tensile stress can be transferred across the crack, i.e. the point when the stress-displacement diagram in Fig. 2-33 drops to zero. These parameters are to be supplied by the user except for the parameter Δ_0, which is automatically extracted from the provided stress-strain law for tension. The shear retention factor is then expressed as

$$\beta=\frac{1}{1+G\left[\dfrac{1}{M(\varepsilon_{\xi\xi}^{mc,\xi})}+\dfrac{1}{M(\varepsilon_{\eta\eta}^{mc,\eta})}+\dfrac{1}{w_c^{\xi}L(\Delta_\xi^{\xi})}+\dfrac{1}{w_c^{\eta}L(\Delta_\eta^{\eta})}\right]} \tag{2.94}$$

Note that for an element containing only multiple cracks (before localization) $\Delta_\xi^\xi=\Delta_\eta^\eta=0$ and $1/L$ terms approach zero. For an uncracked element, $\epsilon_{\xi\xi}^{mc,\xi}=\epsilon_{\eta\eta}^{mc,\eta}=\Delta_\xi^\xi=\Delta_\eta^\eta=0$ and $1/M$ and $1/L$ approach zero, giving β=1.

2.2.13 Confinement-Sensitive Constitutive Model

The *CC3DNonLinCementitious3* fracture-plastic constitutive model is an advanced version of the *CC3DNonLinCementitious2* material that can handle the increased deformation capacity of concrete under triaxial compression. It is suitable for problems including confinement effects such as confined reinforced concrete members (columns, bridge piers), nuclear vessels and triaxial compression tests of plain concrete. A detailed description of the model formulation is presented in PAPANIKOLAOU and KAPPOS (2007). In this section, only the main differences between the *CC3DNonLinCementitious3* and the *CC3DNonLinCementitious2* model are described, which are mainly focused on the plasticity part of the model (section 2.2.4).

2.2.13.1 Hardening and Softening Function

The position of failure surface can expand and move along the hydrostatic axis (simulating the hardening and softening stages), based on the value of the hardening/softening parameter (κ). In the present model, this parameter identifies with the volumetric plastic strain (GRASSL et al., 2002) :

$$d\kappa=d\varepsilon_v^p=d\varepsilon_1^p+d\varepsilon_2^p+d\varepsilon_3^p \tag{2.95}$$

The instantaneous shape and location of the loading surface during hardening is defined by a hardening function (k), which depends on the hardening/softening parameter (κ). This function is directly incorporated in the Menétrey-Willam failure surface equations (2.54), operating as a

scaling factor on the compressive concrete strength (f_c). It has the same elliptic form with *CC3DNonLinCementitious2* (2.57), but herein in terms of the plastic volumetric strain:

$$k(\kappa) = k(\varepsilon_v^p) = k_o + (1 - k_o) \cdot \sqrt{1 - \left(\frac{\varepsilon_{v,t}^p - \varepsilon_v^p}{\varepsilon_{v,t}^p} \right)^2} \tag{2.96}$$

where $\varepsilon_{v,t}^p$ is the plastic volumetric strain at uniaxial concrete strength (onset of softening) and k_o is the value that defines the initial yield surface that bounds the initial elastic regime (onset of plasticity). At the end of the hardening process, the hardening function retains a constant value of unity and the material enters the softening regime, which is controlled by the softening function (c). This function simulates the material decohesion by shifting the loading surface along the negative hydrostatic axis. It is assumed that it follows the softening function originally proposed by VAN GYSEL and TAERWE (1996) for uniaxial compression:

$$c(\kappa) = c(\varepsilon_v^p) = \left(\frac{1}{1 + \left(\frac{n_1 - 1}{n_2 - 1} \right)^2} \right)^2 \tag{2.97}$$

where:

$$n_1 = \varepsilon_v^p / \varepsilon_{v,t}^p \tag{2.98}$$

$$n_2 = (\varepsilon_{v,t}^p + t) / \varepsilon_{v,t}^p \tag{2.99}$$

Parameter t in equation (2.99) controls the slope of the softening function and the outmost square is necessary due to the quadratic nature of the loading surface. The softening function value starts from unity and complete material decohesion is attained at c = 0. The evolution of both hardening and softening functions with respect to the hardening/softening parameter is schematically shown in Fig. 2-34.

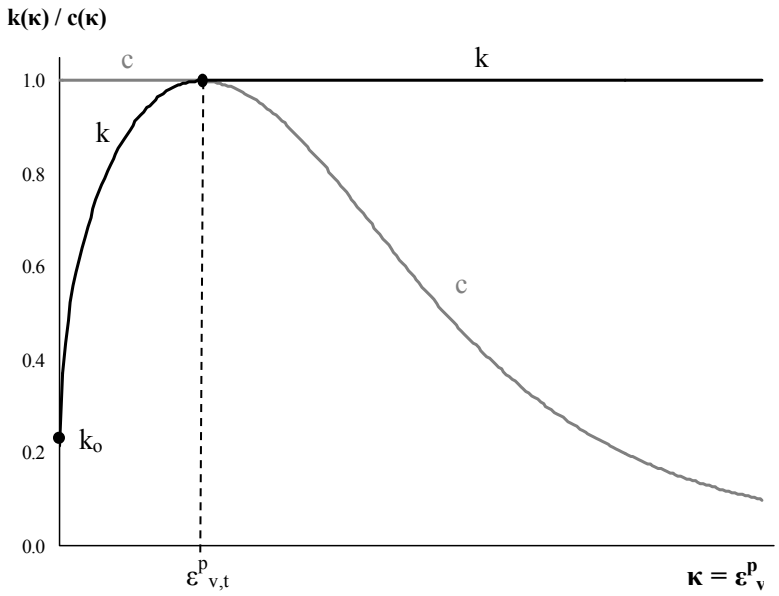


Fig. 2-34: Evolution of hardening (k) and softening (c) functions with respect to the plastic volumetric strain.

2.2.13.2 Plastic Potential Function

The present plasticity model incorporates a non-associated flow rule using a polynomial plastic potential function (g), with Lode angle (θ) dependency and adjustable order (n):

$$g = A \cdot \left(\frac{\rho}{k \cdot \sqrt{c} \cdot f_c} \right)^n + \left[C + \frac{1}{2}(B - C)(1 - \cos 3\theta) \right] \cdot \frac{\rho}{k \cdot \sqrt{c} \cdot f_c} + \frac{\xi}{k \cdot \sqrt{c} \cdot f_c} - a \qquad (2.100)$$

Parameters A, B and C define the shape of the plastic potential function in stress space and their calibration is based on the assumption that the inclination (ψ) of the incremental plastic strain vector identifies with the inclination of the total plastic strain vector at three distinct stress states, namely the uniaxial, biaxial and triaxial compressive concrete strength (Fig. 2-35). The attraction constant (a) is included for mathematical clarity and is not a user parameter, due to plastic potential function differentiation in the flow rule.

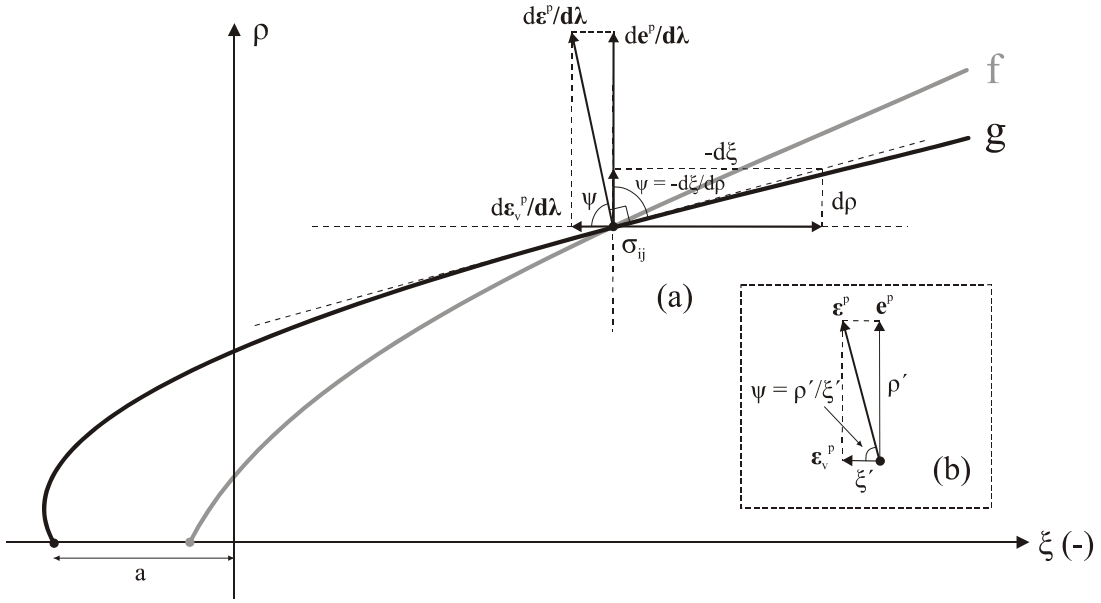


Fig. 2-35: Direction (ψ) of the incremental (a) and total (b) plastic strain vectors.

2.2.13.3 Suggested Model Parameters

A detailed calibration scheme for the plasticity model parameters, based on and extensive experimental database can be found in PAPANIKOLAOU and KAPPOS (2007) and suggested values (including the fracture model parameters) for various uniaxial compressive concrete strengths (f_c) are shown in the following table (see *Atena Input File Format* document for the material definition details):

Table 2.2-1 Suggested parameters for the fracture and plasticity models

f_c (MPa)	20	30	40	50	60	70
E_c (MPa)	24377	27530	30011	32089	33893	35497
ν	0.2	0.2	0.2	0.2	0.2	0.2
f_t (MPa)	1.917	2.446	2.906	3.323	3.707	4.066
λ_t	1.043	1.227	1.376	1.505	1.619	1.722
e	0.5281	0.5232	0.5198	0.5172	0.5151	0.5133
f_{co} (MPa)	-4.32	-9.16	-15.62	-23.63	-33.14	-44.11
$\varepsilon^p_{v,t}$	$4.92\cdot10^{-4}$	$6.54\cdot10^{-4}$	$8.00\cdot10^{-4}$	$9.35\cdot10^{-4}$	$1.06\cdot10^{-3}$	$1.18\cdot10^{-3}$
t	$1.33\cdot10^{-3}$	$2.00\cdot10^{-3}$	$2.67\cdot10^{-3}$	$3.33\cdot10^{-3}$	$4.00\cdot10^{-3}$	$4.67\cdot10^{-3}$
A	7.342177	5.436344	4.371435	3.971437	3.674375	3.43856
B	-8.032485	-6.563421	-5.73549	-5.430334	-5.202794	-5.021407
C	-3.726514	-3.25626	-3.055953	-2.903173	-2.797059	-2.719067
n	3	3	3	3	3	3
G_f (MN/m)	$4.87\cdot10^{-5}$	$6.47\cdot10^{-5}$	$7.92\cdot10^{-5}$	$9.26\cdot10^{-5}$	$1.05\cdot10^{-4}$	$1.17\cdot10^{-4}$

f_c (MPa)	80	90	100	110	120
E_c (MPa)	36948	38277	39506	40652	41727
ν	0.2	0.2	0.2	0.2	0.2
f_t (MPa)	4.405	4.728	5.036	5.333	5.618
λ_t	1.816	1.904	1.986	2.063	2.136
e	0.5117	0.5104	0.5092	0.5081	0.5071
f_{co} (MPa)	-56.50	-70.30	-85.48	-102.01	-114.00
$\varepsilon^p_{v,t}$	$1.30\cdot10^{-3}$	$1.41\cdot10^{-3}$	$1.52\cdot10^{-3}$	$1.62\cdot10^{-3}$	$1.73\cdot10^{-3}$
t	$5.33\cdot10^{-3}$	$6.00\cdot10^{-3}$	$6.67\cdot10^{-3}$	$7.33\cdot10^{-3}$	$8.00\cdot10^{-3}$
A	3.245006	3.082129	2.942391	2.820644	2.713227
B	-4.871993	-4.745867	-4.637358	-4.542587	-4.458782
C	-2.659098	-2.611426	-2.572571	-2.540158	-2.512681
n	3	3	3	3	3
G_f (MN/m)	$1.29\cdot10^{-4}$	$1.40\cdot10^{-4}$	$1.50\cdot10^{-4}$	$1.61\cdot10^{-4}$	$1.71\cdot10^{-4}$

2.3 Von Mises Plasticity Model

Von Mises plasticity model called also as J_2 plasticity is based only on one parameter k. The yield function is defined as:

$$F^p(\sigma_{ij}) = \sqrt{J_2} - k\left(\varepsilon^p_{eq}\right) = 0 \tag{2.101}$$

where J_2 denotes the second invariant of stress deviator tensor. The parameter $k\left(\varepsilon^p_{eq}\right) = \sqrt{1/3}\,\sigma_y\left(\varepsilon^p_{eq}\right)$ is the maximal shear stress and σ_y is the uniaxial yield stress. This parameter controls the isotropic hardening of the yield criterion.

$$\sigma_y\left(\varepsilon^p_{eq}\right) = \sigma_y + H\varepsilon^p_{eq}, \qquad \varepsilon^p_{eq} = \sum_{i=1}^{N_{inc}} \sqrt{2/3\left(\Delta\boldsymbol{\varepsilon}^p : \Delta\boldsymbol{\varepsilon}^p\right)} \tag{2.102}$$

σ_y is the yield stress, H the hardening modulus and ε^p_{eq} is the equivalent plastic strain calculated as a summation of equivalent plastic strains during the loading history.

In case of von Mises plasticity the plastic potential function is identical with the yield function:

$$G^p(\sigma_{ij}) = F^P(\sigma_{ij}) \tag{2.103}$$

The associated flow rule is assumed. The background information can be found in (CHEN, SALEEB 1982, Sec.5.4.2).

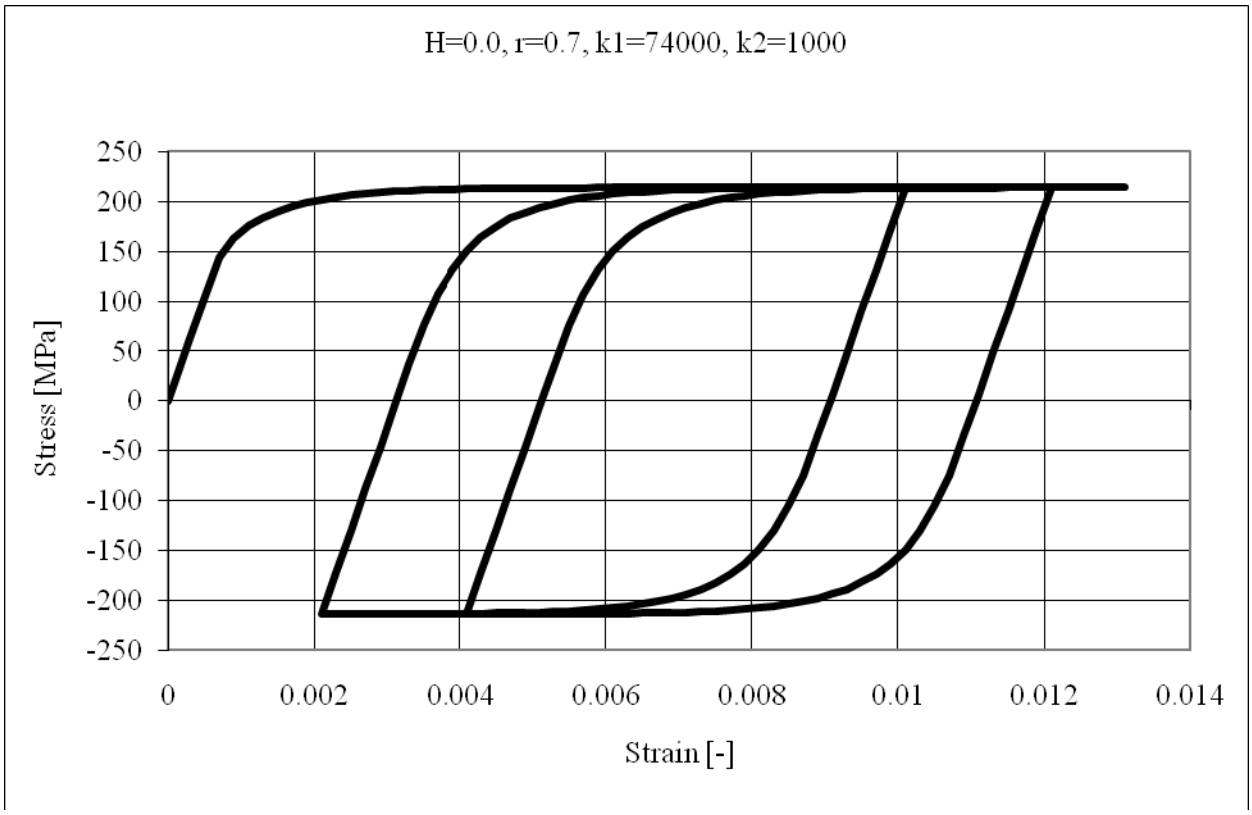
The Von Mises model could be used to model cyclic steel behavior including Bauschinger effect. In this case the yield function is modified as:

$$\sqrt{\tfrac{1}{2}(\boldsymbol{\sigma}' - \mathbf{X}) : (\boldsymbol{\sigma}' - \mathbf{X})} - k\left(\varepsilon_{eq}^{p}\right) - (r-1)k_0 = 0 \tag{2.104}$$

where $\boldsymbol{\sigma}'$ is the deviatoric stress, k_0 is an initial value of $k(\varepsilon_{eq}^{p})$ according to (2.102), $\mathbf{X}$ is the so called back stress controlling the kinematic hardening:

$$\Delta\mathbf{X} = \tfrac{2}{3} k_1 \Delta\boldsymbol{\varepsilon}^{p} - k_2 \mathbf{X} \Delta\varepsilon_{eq}^{p} \tag{2.105}$$

In equations (2.104) and (2.105) quantities r, k_1, k_2 are material parameters for the cyclic response. If r is non-zero, the cyclic model is activated, and it controls the radius of the Von Mises surface. If $r = 1$ the yielding will start exactly when σ_y is reached. For lower values, the non-linear behavior starts earlier, and the slope of the response is mainly affected by parameter k_1 (larger value – higher slope). Parameter k_2 on the other hand affects the memory of the cyclic response. Some examples of various parameter combinations are shown at Fig. 2-36.



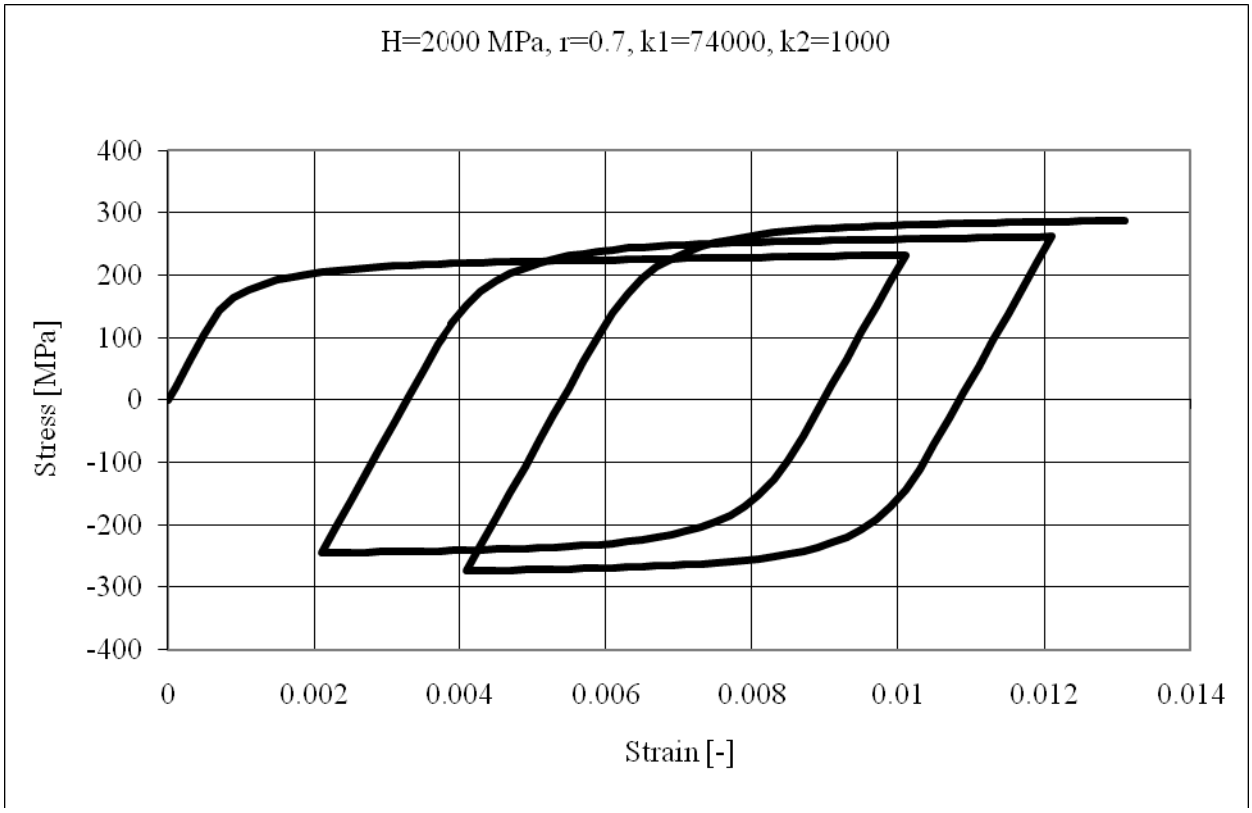


Fig. 2-36: Effect of material parameter choice on cyclic response for E=210 GPa and σ_y = 200 MPa.

2.4 Drucker-Prager Plasticity Model

Drucker-Prager plasticity model is based on a general plasticity formulation that is described in Section 2.2.4. The yield function is defined as:

$$F_{DP}^{p}(\sigma_{ij}) = \alpha I_1 + \sqrt{J_2} - k = 0 \tag{2.106}$$

Where α and k are parameters defining the shape of the failure surface. They can be estimated by matching with the Mohr-Coulomb surface. If the two surfaces are to agree along the compressive meridian, i.e. $\theta = 0^0$, the formulas are:

$$\alpha = \frac{2\sin\phi}{\sqrt{3}\left(3-\sin\phi\right)}, \qquad k = \frac{6c\cos\phi}{\sqrt{3}\left(3-\sin\phi\right)} \tag{2.107}$$

This corresponds to a outer cone to the Mohr-Coulomb surface. The inner cone, which passes through the tensile meridian where $\theta = 60^0$ has the constants given by the following expressions:

$$\alpha = \frac{2\sin\phi}{\sqrt{3}\left(3+\sin\phi\right)}, \qquad k = \frac{6c\cos\phi}{\sqrt{3}\left(3+\sin\phi\right)} \tag{2.108}$$

The position of failure surfaces is not fixed but it can move depending on the value of strain hardening/softening parameter. The strain hardening is based on the equivalent plastic strain, which is calculated according to the following formula.

$$\Delta\varepsilon_{eq}^{p} = \min(\Delta\varepsilon_{ij}^{p}) \tag{2.109}$$

Hardening/softening in the Drucker-Prager model is controlled by the parameter k. This parameter is selected such that the surface at the peak passes through the uniaxial compressive strength, and it changes according to the following expression.

$$k' = k\frac{f_c'(\varepsilon_{eq}^{p})}{f_c'} \tag{2.110}$$

The symbol k' in the above formula replaces k in (2.106). In the above two formulas the expression $f_c'(\varepsilon_{eq}^{p})$ indicates the hardening/softening law, which is based on the uniaxial compressive test. The law is shown in Fig. 2-37.

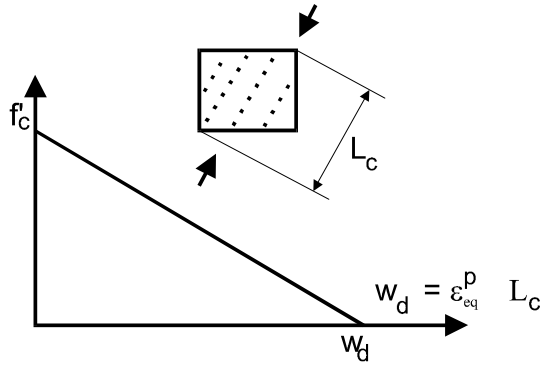


Fig. 2-37. Linear softening in the Drucker-Prager material model

Return direction is given by the following plastic potential:

$$G^{p}(\sigma_{ij}) = \beta\frac{1}{\sqrt{3}}I_1 + \sqrt{2J_2} \tag{2.111}$$

where β determines the return direction. If $\beta < 0$ material is being compacted during crushing, if $\beta = 0$ material volume is preserved, and if $\beta > 0$ material is dilating. In general, the plastic model is non-associated, since the plastic flow is not perpendicular to the failure surface

The return mapping algorithm for the plastic model is based on predictor-corrector approach as is shown in Fig. 2-22. During the corrector phase of the algorithm the failure surface moves along the hydrostatic axis to simulate hardening and softening. The final failure surface has the apex located at the origin of the Haigh-Vestergaard coordinate system. Secant method-based Algorithm 1 is used to determine the stress on the surface, which satisfies the yield condition and also the hardening/softening law.

2.5 User Material Model

In some situations, none of the standard material models available in ATENA can describe the behavior sufficiently. Many such cases can be handled by defining user laws in the fracture-plastic material model (see *CC3DNonLinCementitious2User* described in section 2.2.6), in the others the user can provide a dynamic link library implementing his own material model. The user material is based on the elastic isotropic material, adding new material parameters and state variables (both limited to floating point values). See the *User Material DLL Manual* for description and reference, and the `CCUserMaterialExampleDLL` directory in *Atena Science Examples* for an example project including the source code in **C** and a window help file version of the manual, `AtenaV4_UserMaterialDLL.chm`. Please note that the behavior of the user model may have influence on convergence of the analysis.

2.6 Interface Material Model

The interface material model can be used to simulate contact between two materials such as for instance a construction joint between two concrete segments or a contact between foundation and concrete structure. The interface material is based on Mohr-Coulomb criterion with tension cut off. The constitutive relation for a general three-dimensional case is given in terms of tractions on interface planes and relative sliding and opening displacements.

$$\begin{Bmatrix} \tau_1 \\ \tau_2 \\ \sigma \end{Bmatrix} = \begin{bmatrix} K_{tt} & 0 & 0 \\ 0 & K_{tt} & 0 \\ 0 & 0 & K_{nn} \end{bmatrix} \begin{Bmatrix} \Delta v_1 \\ \Delta v_2 \\ \Delta u \end{Bmatrix} \tag{2.112}$$

For two-dimensional problems second row and column are omitted.

The initial failure surface corresponds to Mohr-Coulomb condition (2.113) with ellipsoid in tension regime. After stresses violate this condition, this surface collapses to a residual surface which corresponds to dry friction.

$$|\tau| \le c - \sigma \cdot \phi, \quad \sigma \le 0 \tag{2.113}$$

$$\tau = \tau_0 \sqrt{1 - \frac{(\sigma - \sigma_c)^2}{(f_t - \sigma_c)^2}}, \quad \tau_0 = \frac{c}{\sqrt{1 - \frac{\sigma_c^2}{(f_t - \sigma_c)^2}}}, \; \sigma_c = -\frac{f_t^2 \phi}{c - 2 f_t \phi}, \quad 0 < \sigma \le f_t$$

$$\tau = 0, \quad \sigma > f_t$$

In tension the failure criterion is replaced by an ellipsoid, which intersect the normal stress axis at the value of f_t with the vertical tangent and the shear axis is intersected at the value of c (i.e. cohesion) with the tangent equivalent to $-\phi$.

The parameters for the interface model cannot be defined arbitrarily; there is certain dependence of some parameters on the others. When defining the interface parameters, the following rules should be observed:

$$f_t < \frac{c}{\phi}, \quad f_t < c$$
$$c > 0, \quad f_t > 0, \quad \phi > 0 \tag{2.114}$$

It is recommended that parameters c, f_t, ϕ are always greater than zero. In cases when no cohesion or no tensile strength is required, some very small values should be prescribed.

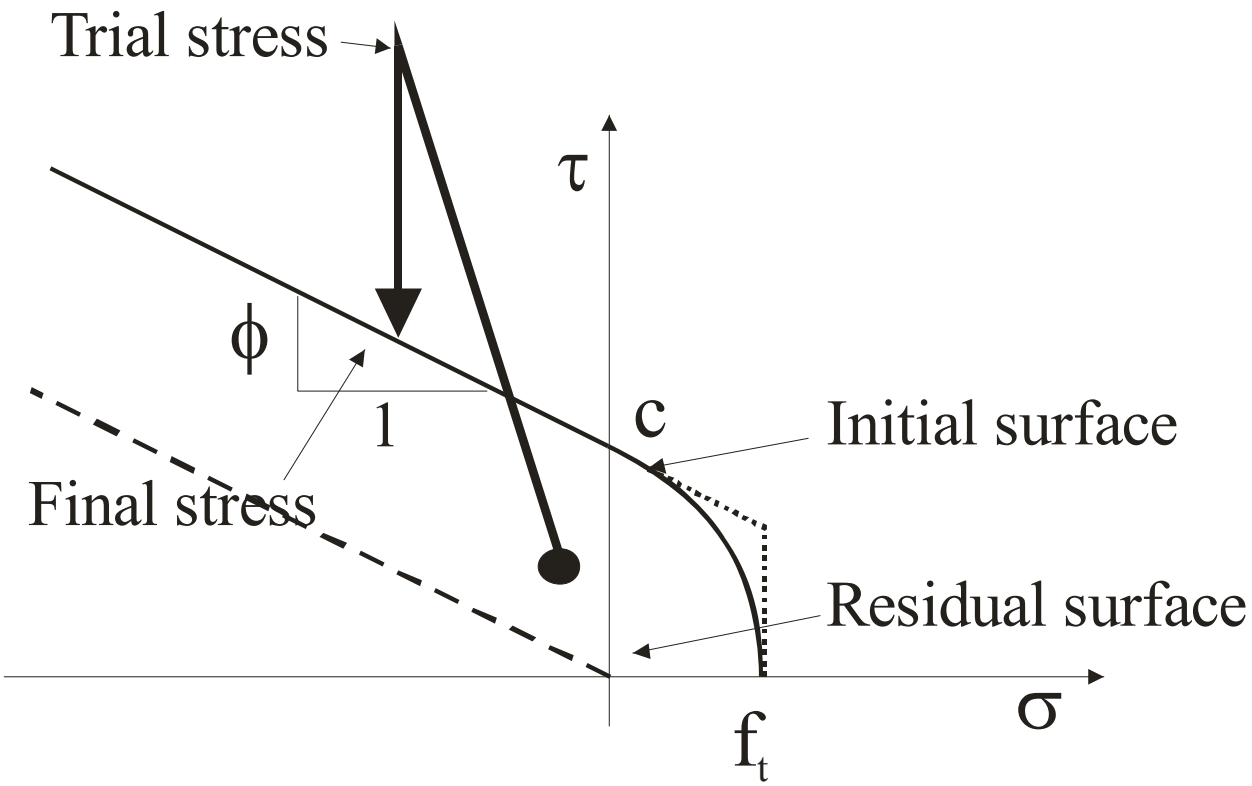


Fig. 2-38: Failure surface for interface elements.

In general three-dimensional case τ in Fig. 2-38 and equation (2.113) is calculated as:

$$\tau = \sqrt{\tau_1^2 + \tau_2^2} \tag{2.115}$$

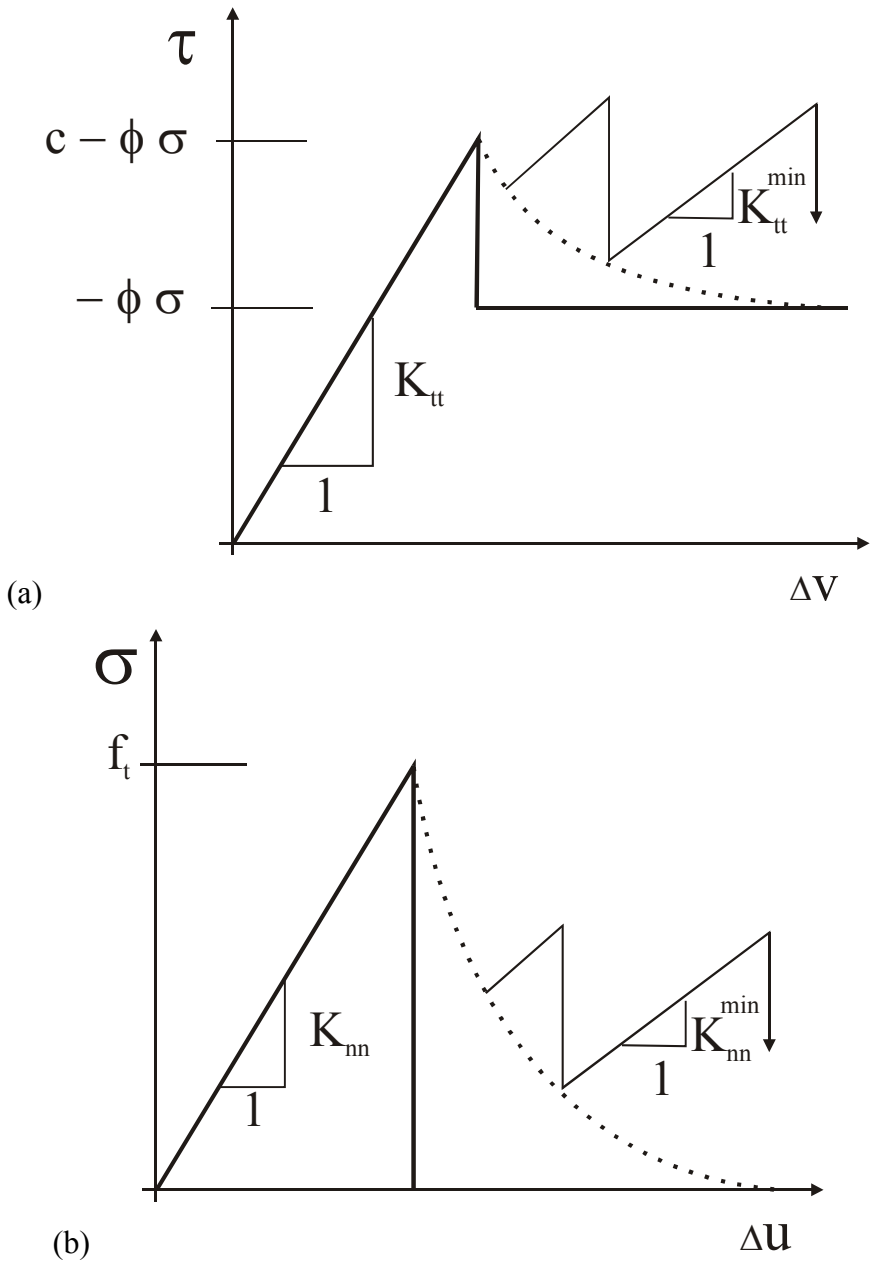


Fig. 2-39: Typical interface model behavior in shear (a) and tension (b)

The K_{nn}, K_{tt} denote the initial elastic normal and shear stiffness, respectively. Typically for zero thickness interfaces, the value of these stiffnesses correspond to a high penalty number. It is recommended not to use extremely high values as this may result in numerical instabilities. It is recommended to estimate the stiffness value using the following formulas

$$K_{nn} = \frac{E}{t}, \qquad K_{tt} = \frac{G}{t} \tag{2.116}$$

where E and G is minimal elastic modulus and shear modulus respectively of the surrounding material. t is the width of the interface zone. Its value can be selected either based on the reality. For instance, for mortar between masonry bricks the value is typically 10-20 mm. Alternatively, it can be estimated as a dimension, which can be considered negligible with respect to the structural size. For instance, in case of a dam analysis, where the dam dimensions are typically in the order of 100 meters, the width of the interface zone can be estimated to be 0.5 meters. It is

suitable due to numerical reasons if stiffness is about 10 times of the stiffness of adjacent finite elements.

There are two additional stiffness values that need to be specified in the ATENA input. They are denoted in Fig. 2-39 as $K_{nn}^{\min}$ and $K_{tt}^{\min}$. They are used only for numerical purposes after the failure of the element to preserve the positive definiteness of the global system of equations. Theoretically, after the interface failure the interface stiffness should be zero, which would mean that the global stiffness will become indefinite. These minimal stiffnesses should be about 0.001 times of the initial ones.

It is possible to define evolution laws for tensile as well as shear softening by arbitrary multilinear laws. Examples of such laws are shown in Fig. 2-40. The figure describes bi-linear softening laws. The break point of this law can be determined for instance by the formula proposed by Bruehwiler and Wittman (1990).

$$s_1 = \frac{f_t}{4}, \qquad v_1 = 0.75\frac{G_F}{f_t} \tag{2.117}$$

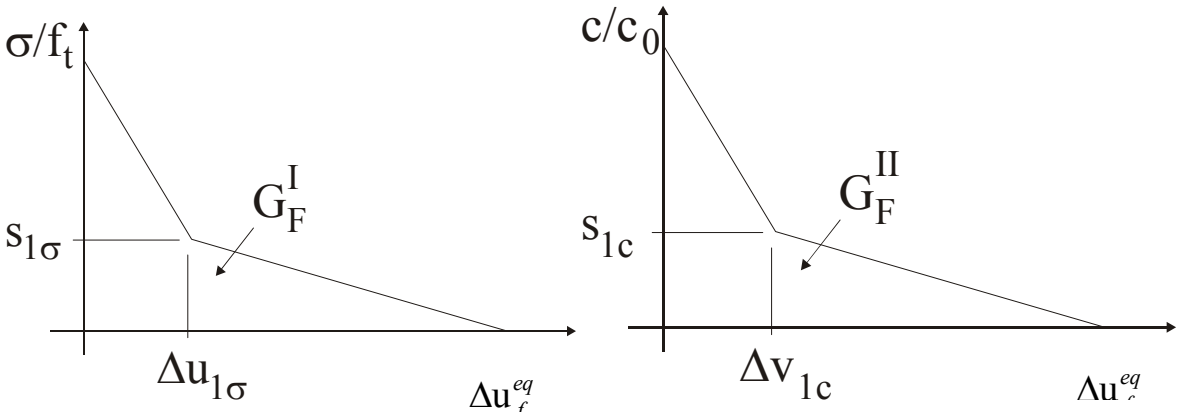


Fig. 2-40: Example of a softening law for tension and cohesion.

The evolution law depends on the equivalent nonlinear interface relative displacement

$$\Delta u_{eq}^{f} = \sqrt{\Delta u_f^2 + \Delta v_{f1}^2 + \Delta v_{f2}^2} \quad \text{in 3D and} \quad \Delta u_{eq}^{f} = \sqrt{\Delta u_f^2 + \Delta v_f^2} \quad \text{in 2D} \tag{2.118}$$

Where Δu_f and Δv_{fi} are the inelastic components of the relative interface displacement on the basis of their decomposition into elastic and nonlinear, i.e. fracturing part.

$$\begin{aligned} \Delta u &= \Delta u_e + \Delta u_f \\ \Delta v_i &= \Delta v_i + \Delta v_{fi} \end{aligned} \tag{2.119}$$

This approach ensures that the degradation in shear affects also tensile strength and vice versa. For instance, when the interface is damaged in shear, the tensile strength is reduced as well. The typical behavior of the interface model with the softening evolution laws is shown in Fig. 2-39 by the dotted lines. The default behavior when no softening law is given is brittle with immediate drop to zero in tension and to the residual dry friction in shear. The behavior is shown in Fig. 2-39 by the solid black line.

When user softening laws are defined for the interface material, it is recommended that the softening law for cohesion is always more ductile then the one for tensile strength, i.e. the cohesion should be higher than the tensile strength at any time during the softening process.

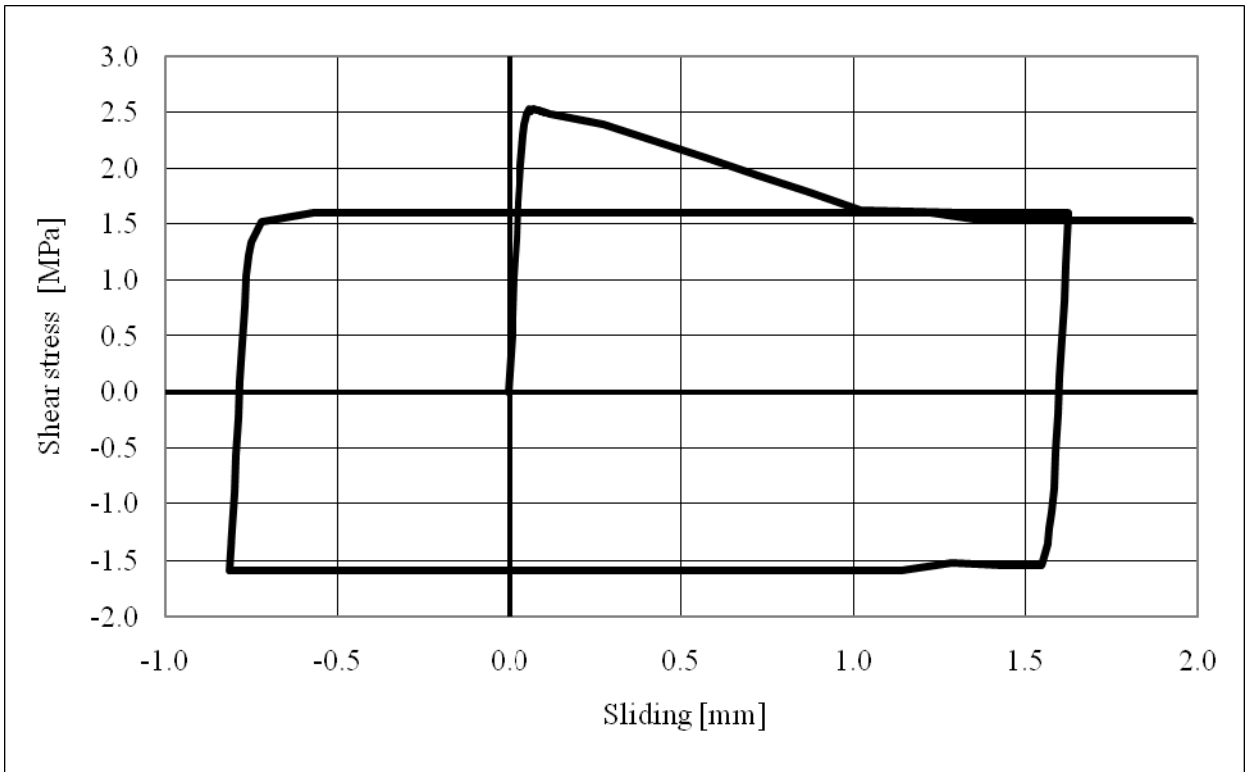


Fig. 2-41: Example of a cyclic response of the model in shear under constant normal pre-stress.

2.7 Reinforcement Stress-Strain Laws

2.7.1 Introduction

Reinforcement can be modeled in two distinct forms: discrete and smeared. Discrete reinforcement is in form of reinforcing bars and is modeled by truss elements. The smeared reinforcement is a component of composite material and can be considered either as a single (only one-constituent) material in the element under consideration or as one of the more such constituents. The former case can be a special mesh element (layer), while the later can be an element with concrete containing one or more reinforcements. In both cases the state of uniaxial stress is assumed, and the same formulation of stress-strain law is used in all types of reinforcement. More info about discrete reinforcement is available in Section 10.2.3 Discrete Reinforcement Embedded in Solid Elements, located near the end of this manual.

2.7.2 Bilinear Law

The bilinear law, elastic-perfectly plastic, is assumed as shown in Fig. 2-42.

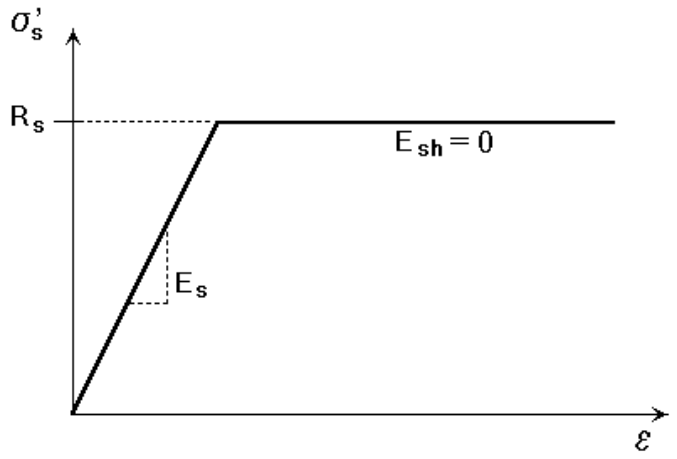


Fig. 2-42 The bilinear stress-strain law for reinforcement.

The initial elastic part has the elastic modulus of steel E_s. The second line represents the plasticity of the steel with hardening and its slope is the hardening modulus E_{sh}. In case of perfect plasticity E_{sh} =0. Limit strain ε_L represents limited ductility of steel.

2.7.3 Multi-line Law

The multi-linear law consists of four lines as shown in Fig. 2-43. This law allows to model all four stages of steel behavior: elastic state, yield plateau, hardening and fracture. The multi-line is defined by four points, which can be specified by input.

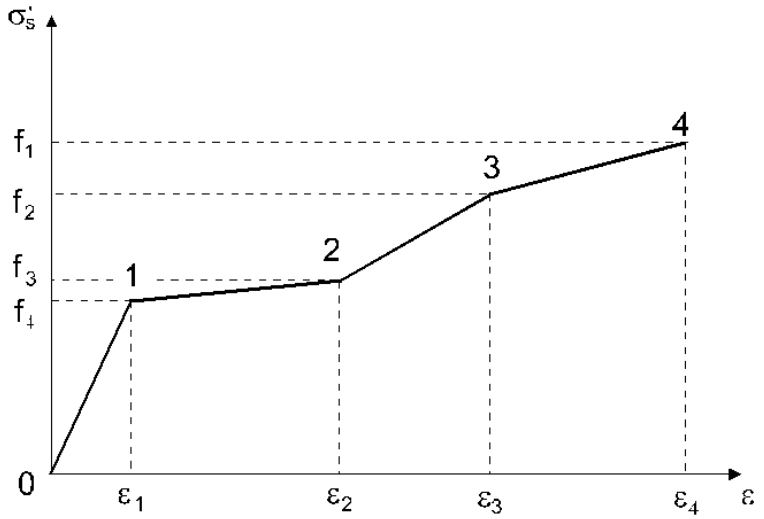


Fig. 2-43 The multi-linear stress-strain law for reinforcement.

The above-described stress-strain laws can be used for the discrete as well as the smeared reinforcement. The smeared reinforcement requires two additional parameters: the reinforcing ratio p (see Section 2.1.1.1) and the direction angle β as shown in Fig. 2-44.

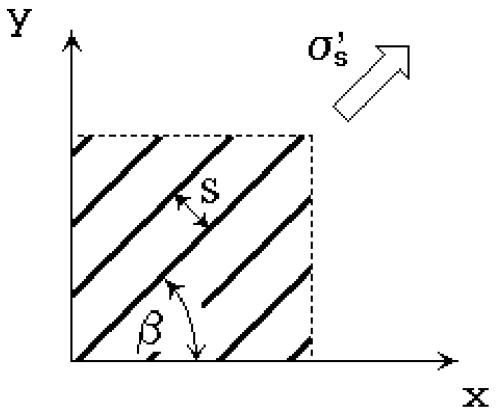


Fig. 2-44 Smeared reinforcement.

The spacing *s* of the smeared reinforcement is assumed infinitely small. The stress in the smeared reinforcement is evaluated in the cracks, therefore it should also include a part of stress due to tension stiffening (which is acting in concrete between the cracks, section 2.1.9).

$$\sigma_{scr}^{'} = \sigma_{s}^{'} + \sigma_{ts} \tag{2.120}$$

where $\sigma_{s}^{'}$ is the steel stress between the cracks (the steel stress in smeared reinforcement), $\sigma_{scr}^{'}$ is the steel stress in a crack. If no tension stiffening is specified σ_{ts}=0 and $\sigma_{scr}^{'} = \sigma_{s}^{'}$. In case of the discrete reinforcement the steel stress is always $\sigma_{s}^{'}$.

2.7.4 No Compression Reinforcement

Normally all reinforcement material models in ATENA exhibit the same behavior in tension as well as in compression. The material types `CCReinforcement` and `CCSmearedReinforcement` include the capability to deactivate the compressive response of the reinforcement. This is sometimes useful, if this material model is used to simulate the behavior of reinforcement elements that have a very low bending stiffness, so it can be assumed that when the reinforcement is loaded by compressive forces, buckling occurs and the strength of the elements in compression is negligible. This is controlled by the command COMPRESSION 0 or 1, which deactivates and activates the compressive response respectively (for more details see ATENA Input File Format).

2.7.5 Cyclic Reinforcement Model

The reinforcing steel stress-strain behavior can be described by the nonlinear model of Menegotto and Pinto (1973). In ATENA this model is extended to account of the isotropic hardening due to an arbitrary hardening law that can be specified for reinforcement (see Sections 2.7.2, 2.7.3). The stress in the cyclic model is calculated according to the following expression.

$$\sigma = \left(\sigma_0 - \sigma_r\right)\sigma^* + \sigma_r \tag{2.121}$$

where

$$\sigma^* = b\varepsilon^* + \frac{\left(1-b\right)\varepsilon^*}{\left(1+\varepsilon^{*R}\right)^{1/R}}, \quad \varepsilon^* = \frac{\varepsilon - \varepsilon_r}{\varepsilon_0 - \varepsilon_r}, \quad R = R_0 - \frac{c_1\xi}{c_2 + \xi} \tag{2.122}$$

where R_0, c_1 and c_2 are experimentally determined parameters, and b the current hardening modulus. The Fig. 2-45 shows the meaning of strain values ε_r, ε_0, ξ and stress values σ_r and σ_0. These values changes for each cycle. The values with the subscript r indicate the point where the cycle started, and the subscript 0 indicates the theoretical yield point that would be reached during the unloading if the response would not have been modified by the hysteretic behavior. During the calculation of this point the material stress-strain law is considered (see Sections 2.7.2, 2.7.3)

$$\sigma^* = f_R\left(\varepsilon_{eq}\right), \quad \varepsilon_{eq} = \sum_{i=1}^{N_{incr.}} \left|\Delta\varepsilon_{eq}^i\right| \tag{2.123}$$

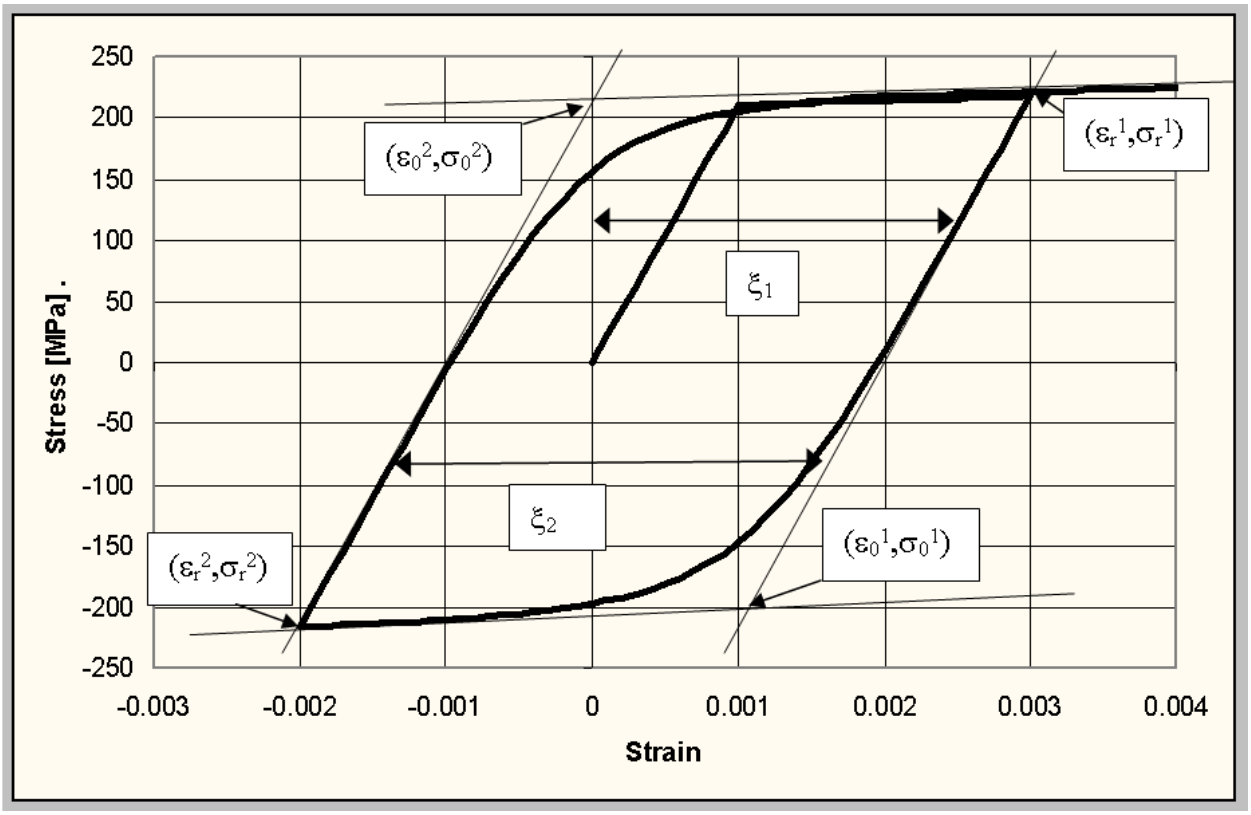


Fig. 2-45: Cyclic reinforcement model based on Menegotto and Pinto (1973).

2.7.6 Cyclic Reinforcement Model – Steel DRC

Another nonlinear constitutive model for reinforcement which captures cyclic behavior and is implemented in ATENA is described by Dodd and Restrepo (1995) and further improved by Se-Hyung Kim (2015).

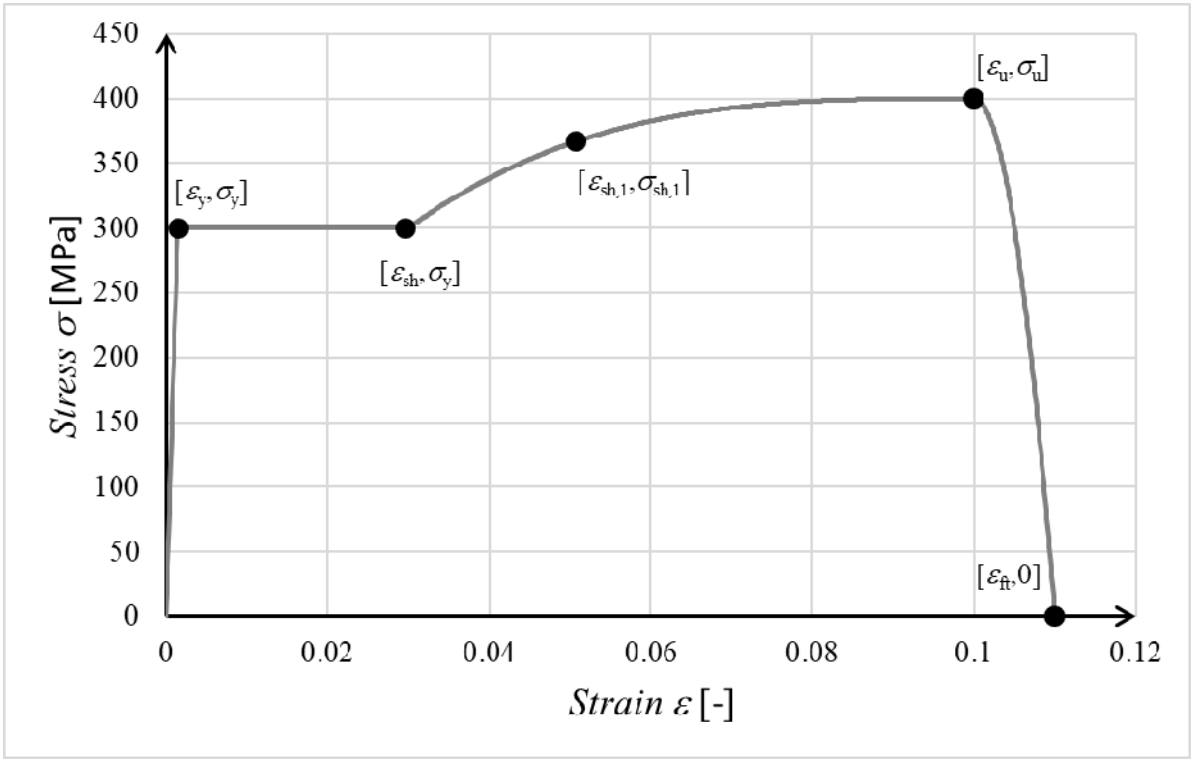


Fig. 2-46: Cyclic reinforcement model based on Dodd and Restrepo (1995) – backbone curve definition points.

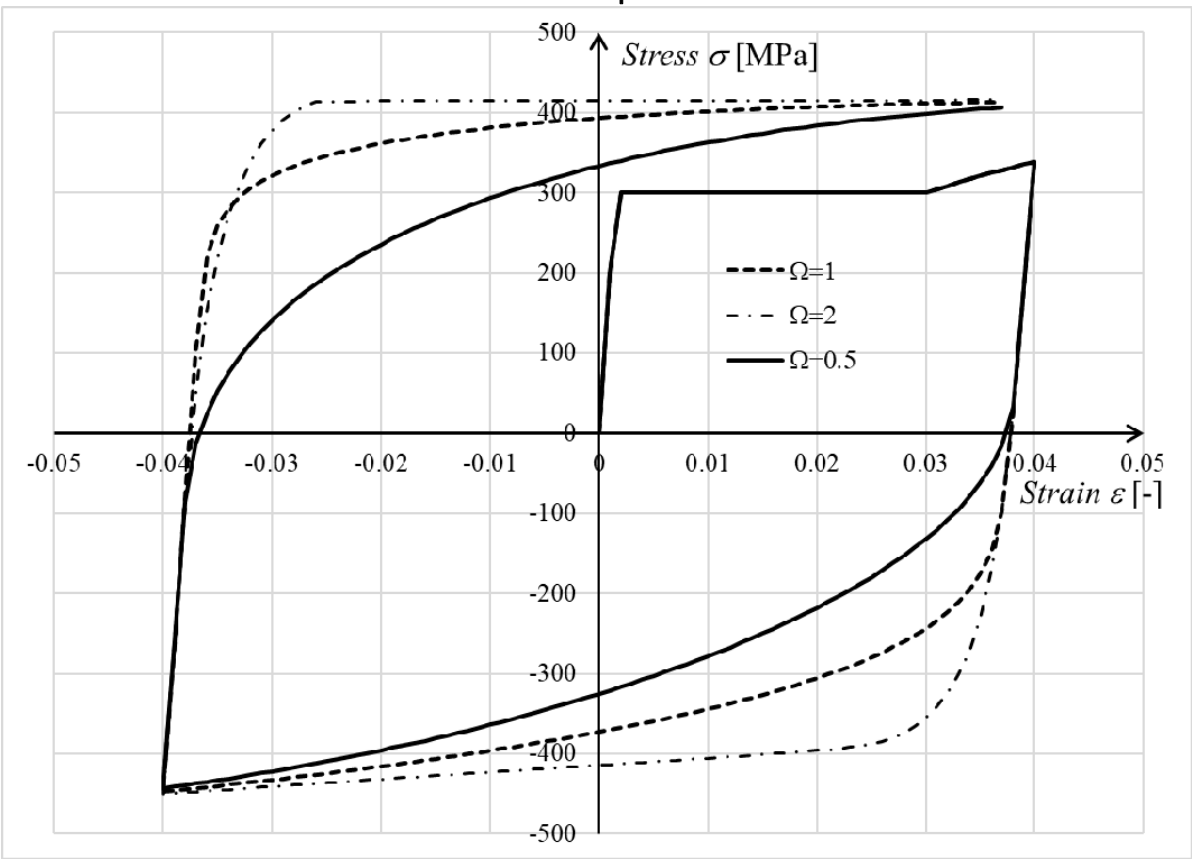


Fig. 2-47: Cyclic reinforcement model based on Dodd and Restrepo (1995) – effect of parameter Ω.

2.8 Reinforcement Bond Models

The basic property of the reinforcement bond model is the bond-slip relationship. This relationship defines the bond strength (cohesion) τ_b depending on the value of current slip between reinforcement and surrounding concrete. ATENA contains three bond-slip models: according to the CEB-FIB model code 1990, slip law by Bigaj and the user defined law. In the first two models, the laws are generated based on the concrete compressive strength, reinforcement diameter and reinforcement type. The important parameters are also the confinement conditions and the quality of concrete casting.

2.8.1 CEB-FIP 1990 Model Code

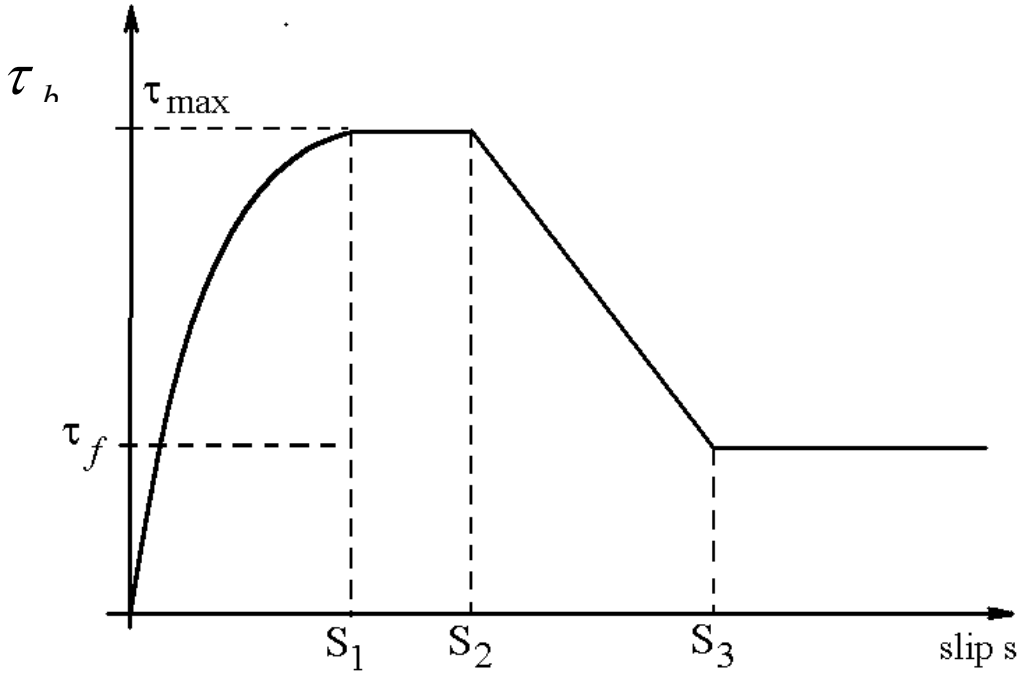


Fig. 2-48: Bond-slip law by CEB-FIP model code 1990.

$$\tau_b = \tau_{\max} \left(\frac{s}{s_1} \right)^{\alpha}, \quad 0 \le s \le s_1 \tag{2.124}$$

$$\tau_b = \tau_{\max}, \qquad s_1 < s \le s_2 \tag{2.125}$$

$$\tau_b = \tau_{\max} - \left(\tau_{\max} - \tau_f\right)\left(\frac{s - s_2}{s_3 - s_2}\right), \quad s_2 < s \leq s_3 \tag{2.126}$$

$$\tau_b = \tau_f, \quad s_3 < s \tag{2.127}$$

Table 2.8-1: Parameters for defining the mean bond strength-slip relationship for ribbed bars.

	2	3	4	5
Value	Unconfined concrete*		Confined concrete**	
	Bond conditions		Bond conditions	
	Good	All other cases	Good	All other cases
S_1	0.6 mm	0.6 mm	1.0 mm	
S_2	0.6 mm	0.6 mm	3.0 mm	
S_3	1.0 mm	2.5 mm	clear rib spacing	
α	0.4		0.4	
τ_{max}	$2.0\sqrt{f_c}$	$1.0\sqrt{f_c}$	$2.5\sqrt{f_c}$	$1.25\sqrt{f_c}$
τ_f	$0.15\ \tau_{\max}$		$0.40\ \tau_{\max}$	

* Failure by splitting of the concrete

**Failure by shearing of the concrete between the ribs

Table 2.8-2: Parameters for defining the bond strength-slip relationship for smooth bars.

Values	Cold drawn wire		Hot rolled bars	
	Bond conditions		Bond conditions	
	Good	All other cases	Good	All other cases
$s_1 = s_2 = s_3$	0.01 mm		0.1 mm	
α	0.5		0.5	

$\tau_{max} = \tau_f$	$0.1\sqrt{f_c}$	$0.05\sqrt{f_c}$	$0.3\sqrt{f_c}$	$0.15\sqrt{f_c}$

2.8.2 Bond Model by Bigaj

The second pre-defined bond model available in ATENA is based on the work by BIGAJ 1999. This model depends on the bond quality, concrete cubic compressive strength $f_{cu}^{'}$ and reinforcement bar radius D. The slip law for this model is shown in Fig. 2-49.

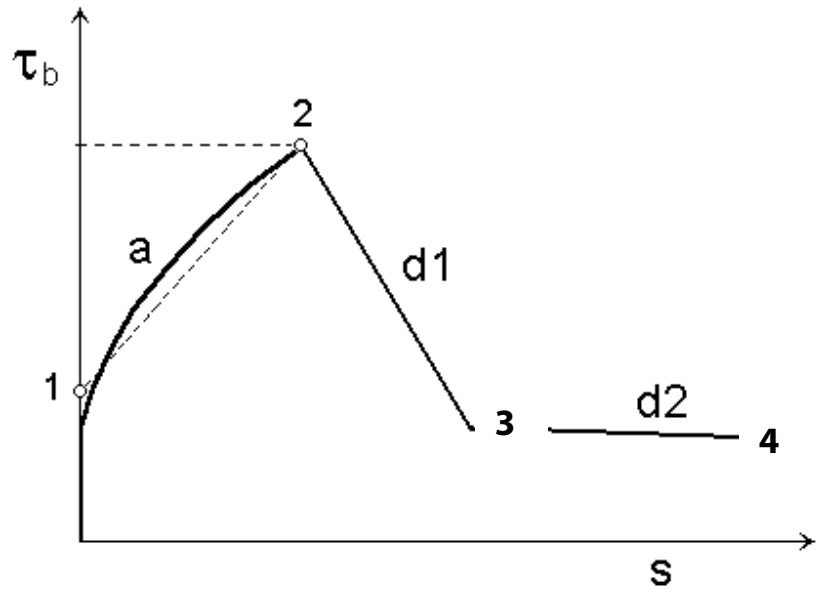


Fig. 2-49: Bond law by BIGAJ 1999

The ascending part of the stress-slip law i.e. part a is modeled by a bi-linear curve. The coordinates of the four points defining this stress-slip relationship are listed in the table below.

Table 2.8-3: Parameters for defining the bond strength-slip relationship for ribbed bars.

Concrete Type	Bond quality		Point 1	Point 2	Point 3	Point 4
$f_c^{'} < 60$	Excelent	s / D	0.000	0.020	0.044	0.480
		$\tau_b / \sqrt{0.8\, f_{cu}^{'}}$	0.500	3.000	0.700	0.000
	Good	s / D	0.000	0.030	0.047	0.480
		$\tau_b / \sqrt{0.8\, f_{cu}^{'}}$	0.500	2.000	0.700	0.000
	Bad	s / D	0.000	0.040	0.047	0.480
		$\tau_b / \sqrt{0.8\, f_{cu}^{'}}$	0.500	1.000	0.700	0.000
	Excelent	s / D	0.000	0.012	0.030	0.340

$f_c^{'} > 60$		$\tau_b / \sqrt{0.88\, f_{cu}^{'}}$	0.600	2.500	0.900	0.000
	Good	s / D	0.000	0.020	0.030	0.340
		$\tau_b / \sqrt{0.88\, f_{cu}^{'}}$	0.600	1.900	0.900	0.000
	Bad	s / D	0.000	0.025	0.030	0.340
		$\tau_b / \sqrt{0.88\, f_{cu}^{'}}$	0.600	1.100	0.900	0.000

2.8.3 Memory Bond Material

The Memory Bond material is an improvement to better capture the response during cyclic loading and unloading in general. It can be used with any of the above-mentioned bond strength – bond slip envelope functions. The response only differs after the bond stress sign changes. Instead of following the same envelope as during loading, the maximum bond stress is determined by the additional parameter τ_1, see Fig. 2-50. Admissible values are $\tau_{res} \le \tau_1 \le \tau_{\max}$, where τ_{res} is the residual bond stress (last value from the bond strength – bond slip function) and $\tau_{\max}$ the maximum bond stress (max. value from the bond strength – bond slip function).

In the figure, s is the current slip value, $s_{\max}$ the maximum of the absolute slip value ever reached (damage variable), $\tau = f(s)$ is the bond strength function.

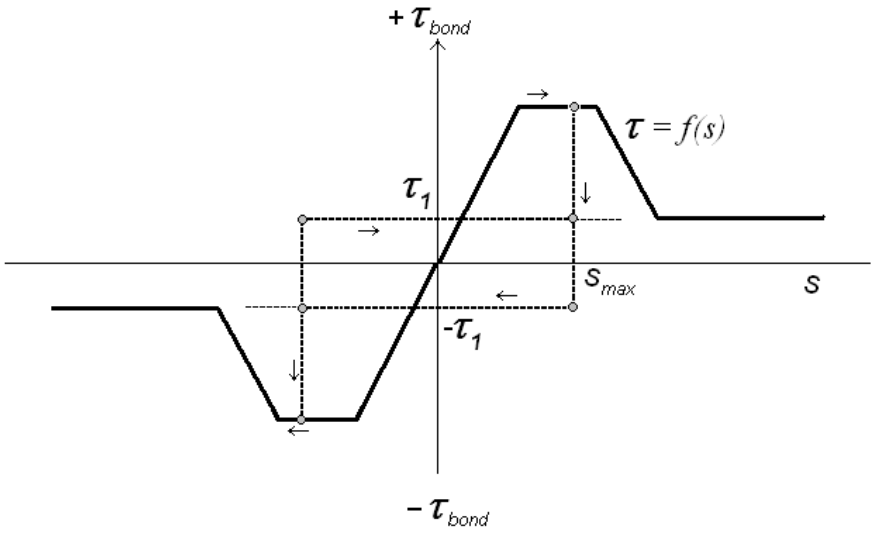


Fig. 2-50: Memory Bond working diagram

The response for a slip change $s_i = s_{i-1} + \Delta s_i$ is defined separately for 2 cases:

(1) Loading range $|s| \ge s_{\max}$

$\tau = f(s)$

(2) Unloading range $-s_{\max} < s < s_{\max}$

$$\Delta s \geq 0 \quad \tau = \tau_1$$

$$\Delta s \leq 0 \quad \tau = -\tau_1$$

2.9 Microplane Material Model (CCMicroplane4)

The basic idea of the microplane model is to abandon constitutive modelling in terms of tensors and their invariants and formulate the stress-strain relation in terms of stress and strain vectors on planes of various orientations in the material, now generally called the microplanes. This idea arose in G.I. Taylor's (TAYLOR 1938) pioneering study of hardening plasticity of polycrystalline metals. Proposing the first version of the microplane model, BAZANT 1984, in order to model strain softening, extended or modified Taylor's model in several ways (in detail see BAZANT et al. 2000), among which the main one was the kinematic constraint between the strain tensor and the microplane strain vectors. Since 1984, there have been numerous improvements and variations of the microplane approach. A detailed overview of the history of the microplane model is included in BAZANT et al 2000 and CANER and BAZANT 2000. In what follows, we briefly review the derivation of the microplane model that is used in this work.

In the microplane model, the constitutive equations are formulated on a plane, called microplane, having an arbitrary orientation characterized by its unit normal n_i. The kinematic constraint means that the normal strain ε_N and shear strains $\varepsilon_M, \varepsilon_L$ on the microplane are calculated as the projections of the macroscopic strain tensor ε_{ij}:

$$\varepsilon_N = n_i n_j \varepsilon_{ij}, \quad \varepsilon_M = \frac{1}{2}\left(m_i n_j + m_j n_i\right)\varepsilon_{ij}, \quad \varepsilon_L = \frac{1}{2}\left(l_i n_j + l_j n_i\right)\varepsilon_{ij} \tag{2.128}$$

where m_i and l_i are chosen orthogonal vectors lying in the microplane and defining the shear strain components. The constitutive relations for the microplane strains and stresses can be generally stated as:

$$\begin{aligned} \sigma_N(t) &= F^t_{\tau=0}\left[\varepsilon_N(\tau), \varepsilon_L(\tau), \varepsilon_M(\tau)\right] \\ \sigma_M(t) &= G^t_{\tau=0}\left[\varepsilon_N(\tau), \varepsilon_L(\tau), \varepsilon_M(\tau)\right] \\ \sigma_L(t) &= G^t_{\tau=0}\left[\varepsilon_N(\tau), \varepsilon_L(\tau), \varepsilon_M(\tau)\right] \end{aligned} \tag{2.129}$$

where F and G are functionals of the history of the microplane strains in time t. For a detailed derivation of these functionals a reader is referred to BAZANT et al 2000 and CANER and BAZANT 2000. The macroscopic stress tensor is obtained by the principle of virtual work that is applied to a unit hemisphere Ω. After the integration, the following expression for the macroscopic stress tensor is recovered (BAZANT 1984):

$$\sigma_{ij} = \frac{3}{2\pi}\int_\Omega s_{ij}\, d\Omega \approx 6\sum_{\mu=1}^{N_m} w_\mu s_{ij}^{(\mu)}, \quad \text{where } s_{ij} = \sigma_N n_i n_j + \frac{\sigma_M}{2}\left(m_i n_j + m_j n_i\right) + \frac{\sigma_L}{2}\left(l_i n_j + l_j n_i\right) \tag{2.130}$$

where the integral is approximated by an optimal Gaussian integration formula for a spherical surface; numbers μ label the points of the integration formula and w_μ are the corresponding optimal weights.

2.9.1 Equivalent Localization Element

The objective of the equivalent localization element is to achieve equivalence with the crack band model. This basic idea is that the material properties and parameters of the softening material model are not modified to account for the differences in the finite element size, but rather the softening crack band is coupled in series with an elastically behaving layer, to obtain equivalence. For brevity, this layer will henceforth be called the `spring'. For large finite elements, the effective length of this added elastic spring, representing the thickness of the added elastic layer having the elastic properties of the material, will be much larger than the size (or thickness) of the localization zone (crack band). Thus, after the crack initiation, the energy stored in the elastic spring can be readily transferred to the localization zone and dissipated in the softening (i.e., fracturing) process.

Inside each finite element at each integration point, an equivalent localization element is assumed. The localization element is a serial arrangement of the localization zone, which is loading, and an elastic zone (spring), which is unloading. The total length of the element is equivalent to the crack band size L (width), and can be determined using the same methods as described in Section 2.1.3 (see Fig. 2-12). The width of the localization zone is given either by the characteristic length of the material or by the size of the test specimen for which the adopted material model has been calibrated.

The three-dimensional equivalent element is constructed by three serial arrangements of the elastic zone (spring) and localization band. The spring-band systems are perpendicular to each other, and they are arranged parallel to the principal strain directions (Fig. 2-51). The simplified two-dimensional version is shown in Fig. 2-52. In this arrangement of spring-band systems it is possible to identify the following unknown stresses and strains:

$$\sigma_{ij}^{b},\ {}^{1}\sigma_{ij}^{u},\ {}^{2}\sigma_{ij}^{u},\ {}^{3}\sigma_{ij}^{u} \quad \text{and} \quad \varepsilon_{ij}^{b},\ {}^{1}\varepsilon_{ij}^{u},\ {}^{2}\varepsilon_{ij}^{u},\ {}^{3}\varepsilon_{ij}^{u}$$

where superscript b denotes the quantities in the localization band and the symbol ${}^{m}x^{u}$ with superscripts u and m defines the quantities in the elastic spring in the direction m.

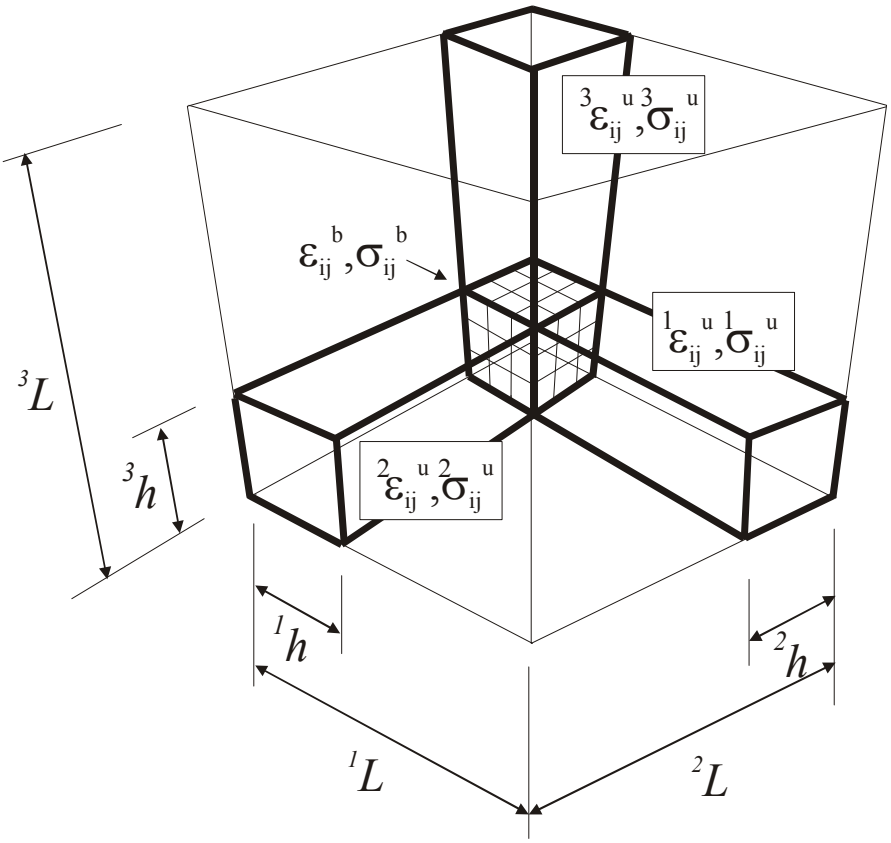


Fig. 2-51: The arrangement of the three-dimensional equivalent localization element.

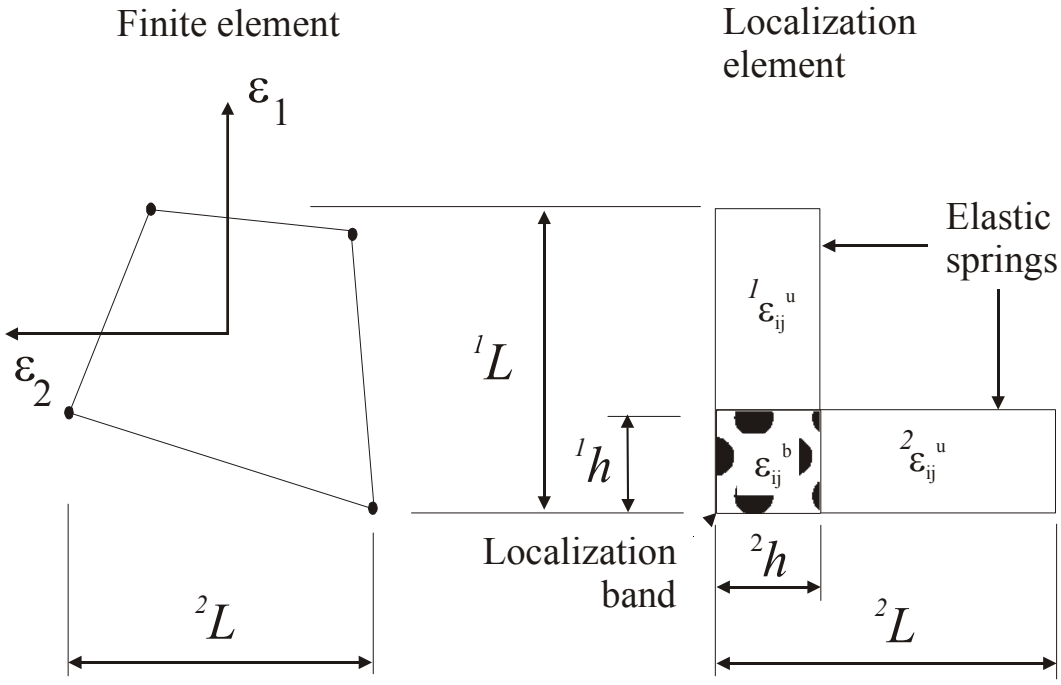


Fig. 2-52: The simplified two-dimensional view of the spring-band arrangement.

Ideally, the chosen directions should be perpendicular to the planes of failure propagation. In ATENA, it is assumed for them to be aligned with the principal axes of the total macroscopic strain tensor, which in most cases should approximately correspond to the above requirement.

Altogether there are 48 unknown variables. In the subsequent derivations, it is assumed that these stresses and strains are defined in the principal frame of the total macroscopic strain tensor. The set of equations available for determining these variables starts with the constitutive formulae for the band and the elastic springs:

$$\sigma_{ij}^{b} = F(\varepsilon_{ij}^{b}) \tag{2.131}$$

$$^{m}\sigma_{ij}^{u} = D_{ijlk}\ ^{m}\varepsilon_{kl}^{u} \quad \text{for } m = 1...3 \tag{2.132}$$

The first formula (2.131) represents the evaluation of the non-linear material model, which in our case is the microplane model for concrete. The second equation (2.132) is a set of three elastic constitutive formulations for the three linear zones (springs) that are involved in the arrangement at Fig. 2-51. This provides the first 24 equations, which can be used for the calculation of unknown strains and stresses.

The second set of equations is provided by the kinematic constrains on the strain tensors.

$$\begin{aligned}
\varepsilon_{11} &= \frac{1}{^{1}L}\left[\varepsilon_{11}^{b}\ ^{1}h + {}^{1}\varepsilon_{11}^{u}\left(^{1}L - {}^{1}h\right)\right] \\
\varepsilon_{22} &= \frac{1}{^{2}L}\left[\varepsilon_{22}^{b}\ ^{2}h + {}^{2}\varepsilon_{22}^{u}\left(^{2}L - {}^{2}h\right)\right] \\
\varepsilon_{33} &= \frac{1}{^{3}L}\left[\varepsilon_{33}^{b}\ ^{3}h + {}^{3}\varepsilon_{33}^{u}\left(^{3}L - {}^{3}h\right)\right] \\
\varepsilon_{12} &= \frac{1}{2}\left\{\frac{1}{^{1}L}\left[\varepsilon_{12}^{b}\ ^{1}h + {}^{1}\varepsilon_{12}^{u}\left(^{1}L - {}^{1}h\right)\right] + \frac{1}{^{2}L}\left[\varepsilon_{12}^{b}\ ^{2}h + {}^{2}\varepsilon_{12}^{u}\left(^{2}L - {}^{2}h\right)\right]\right\} \\
\varepsilon_{23} &= \frac{1}{2}\left\{\frac{1}{^{2}L}\left[\varepsilon_{23}^{b}\ ^{2}h + {}^{2}\varepsilon_{23}^{u}\left(^{2}L - {}^{2}h\right)\right] + \frac{1}{^{3}L}\left[\varepsilon_{23}^{b}\ ^{3}h + {}^{3}\varepsilon_{23}^{u}\left(^{3}L - {}^{3}h\right)\right]\right\} \\
\varepsilon_{13} &= \frac{1}{2}\left\{\frac{1}{^{1}L}\left[\varepsilon_{13}^{b}\ ^{1}h + {}^{1}\varepsilon_{13}^{u}\left(^{1}L - {}^{1}h\right)\right] + \frac{1}{^{3}L}\left[\varepsilon_{13}^{b}\ ^{3}h + {}^{3}\varepsilon_{13}^{u}\left(^{3}L - {}^{3}h\right)\right]\right\}
\end{aligned} \tag{2.133}$$

These 6 additional equations can be written symbolically as:

$$\varepsilon_{ij} = \frac{1}{2}\left\{\frac{1}{^{i}L}\left[\varepsilon_{ij}^{b}\ ^{i}h + {}^{i}\varepsilon_{ij}^{u}\left(^{i}L - {}^{i}h\right)\right] + \frac{1}{^{j}L}\left[\varepsilon_{ij}^{b}\ ^{j}h + {}^{j}\varepsilon_{ij}^{u}\left(^{j}L - {}^{j}h\right)\right]\right\} \tag{2.134}$$

The next set of equations is obtained by enforcing equilibrium in each direction between the corresponding stress components in the elastic zone and in the localization band. For each direction m, the following condition must be satisfied:

$$\sigma_{ij}^{b}\ ^{m}e_{j} = {}^{m}\sigma_{ij}^{u}\ ^{m}e_{j} \qquad \text{for} \quad m = 1...3 \tag{2.135}$$

where $^{m}e_{j}$ denotes coordinates of a unit direction vector for principal strain direction m. Since the principal frame of the total macroscopic strain tensor is used the unit vectors have the following coordinates:

$$^{1}e_{j} = (1,0,0),\ ^{2}e_{j} = (0,1,0),\ ^{3}e_{j} = (0,0,1) \tag{2.136}$$

The remaining equations are obtained by enforcing equilibrium between tractions on the other surfaces of the band and the elastic zone (layer) imagined as a spring:

$$\sigma_{ij}^{b}\ ^{m}e_{j} = {}^{n}\sigma_{ij}^{u}\ ^{m}e_{j} \qquad \text{where} \quad m = 1..3, n = 1...3, m \neq n \tag{2.137}$$

The equation (2.137) is equivalent to a static constraint on the remaining stress and strain components of the elastic springs. Formulas (2.135) and (2.137) together with the assumption of stress tensor symmetry represent the remaining 18 equations that are needed for the solution of the three-dimensional equivalent localization element. These 18 equations can be written as:

$$\sigma_{ij}^{b} = {}^{m}\sigma_{ij}^{u} \qquad \text{for} \quad m = 1...3 \tag{2.138}$$

This means that the macroscopic stress must be equal to σ_{ij}^{b}, i.e., the stress in the localization element, and that the stresses in all the three elastic zones must be equal and to the microplane stress σ_{ij}^{b}. This also implies the equivalence of all the three elastic strain tensors.

Based on the foregoing derivations, it is possible to formulate an algorithm for the calculation of unknown quantities in the three-dimensional equivalent localization element.

Input: $$\varepsilon_{ij}, \Delta\varepsilon_{ij}, \varepsilon_{ij}^{b}, \varepsilon_{ij}^{u} \tag{2.139}$$

Initialization: $$\Delta\varepsilon_{ij}^{b} = \Delta\varepsilon_{ij}^{u} = \Delta\varepsilon_{ij} \tag{2.140}$$

Step 1: $$d\varepsilon_{ij}^{u(i)} = \frac{{}^{i}L\ {}^{j}h + {}^{j}L\ {}^{i}h}{2\,{}^{i}L\ {}^{j}L} C_{ijkl} r_{kl}^{(i-1)} \tag{2.141}$$

Step 2: $$\Delta\varepsilon_{ij}^{u(i)} = \Delta\varepsilon_{ij}^{u(i-1)} + d\varepsilon_{ij}^{u(i)} \tag{2.142}$$

Step 3: $$\Delta\varepsilon_{ij}^{b(i)} = \frac{2\,{}^{i}L\ {}^{j}L}{{}^{i}L\ {}^{j}h + {}^{j}L\ {}^{i}h}\Delta\varepsilon_{ij} - \frac{2\,{}^{i}L\ {}^{j}L - {}^{i}L\ {}^{j}h - {}^{j}L\ {}^{i}h}{{}^{i}L\ {}^{j}h + {}^{j}L\ {}^{i}h}\Delta\varepsilon_{ij}^{u} \tag{2.143}$$

Step 4: $$r_{ij}^{(i)} = \sigma_{ij}^{b(i)} - \sigma_{ij}^{u(i)} \tag{2.144}$$

where C_{ijlk} is the compliance tensor. The above iterative process is controlled by the following convergence criteria;

$$\frac{\left\| d\varepsilon_{ij}^{u(i)} \right\|}{\left\| \Delta\varepsilon_{ij} \right\|} < e \quad , \quad \frac{\left\| r_{ij}^{(i)} \right\|}{\left\| \Delta\sigma_{ij}^{b} \right\|} < e \quad , \quad \frac{\left| r_{ij}^{(i)\mathrm{T}}\, d\varepsilon_{ij}^{u(i)} \right|}{\left\| \Delta\sigma_{ij}^{b}\, \Delta\varepsilon_{ij} \right\|} < e \tag{2.145}$$

The macroscopic stress is then equal to the stress in the localization band σ_{ij}^{b}. More details about the derivations of the above algorithm as well as various examples of application can be obtained from the original reference CERVENKA et al. 2004. It should be noted that the described equivalent localization element is used only if the calculated crack band size L (see Section 2.1.3) in each principal strain direction is larger than the prescribed localization band size h. For smaller element sizes the equivalent localization approach is not used and mesh-dependent results may be obtained.

2.10 References

BASQUIN, H.O. (1910), The exponential law of endurance tests, Proc. ASTM, 10 (II).

BAZANT, Z.P, OH, B.H (1983) - Crack Band Theory for Fracture of Concrete, Materials and Structures, RILEM, Vol. 16, 155-177.

BAŽANT, Z.P., (1984), 'Microplane model for strain controlled inelastic behavior', Chapter 3 in *Mechanics of Engineering Materials* (Proc., Conf. held at U. of Arizona, Tucson, Jan. 1984), C.S. Desai and R.H. Gallagher, eds., J. Willey, London, 45-59.

BAŽANT, Z.P., CANER, F.C., CAROL, I., ADLEY, M.D., AND AKERS, S.A., (2000), 'Microplane Model M4 for Concrete: I. Formulation with Work-Conjugate Deviatoric Stress', *J. of Engrg. Mechanics ASCE,* **126** (9), 944-961.

BIGAJ, A.J (1999) - Structural Dependence of Rotation Capacity of Plastic Hinges in RC Beams and Slabs, PhD Thesis, Delft University of Technogy, ISBN 90-407-1926-8.

BRUEHWILER, E., and WITTMAN, F.H. (1990), "The Wedge Splitting Test, A New Method of Performing Stable Fracture-Mechanics Tests", *Engineering Fracture Mechanics*, Vol. 35, No. 1-3, pp. 117-125.

CANER, F.C., AND BAŽANT, Z.P., (2000) 'Microplane Model M4 for Concrete: II. algorithm and calibration.", *J. of Engrg. Mechanics ASCE,* **129** (9), 954-961.

CEB-FIP Model Code 1990, First Draft, Comittee Euro-International du Beton, Bulletin d'information No. 195,196, Mars.

CEB 1988, Bulletin D'Information No 188, Fatigue of concrete structures, State of the art report.

CERVENKA, V., GERSTLE, K. (1972) - Inelastic Analysis of Reinforced Concrete Panels: (1) Theory, (2) Experimental Verification and application, Publications IABSE, Zürich, V.31-00, 1971, pp.32-45, and V.32-II,1972, pp.26-39.

CERVENKA, V. (1985) - Constitutive Model for Cracked Reinforced Concrete, Journal ACI, Proc. V.82, Nov-Dec., No.6,pp.877-882.

CERVENKA, V., PUKL, R., ELIGEHAUSEN, R. (1991) - Fracture Analysis of Concrete Plane Stress Pull-out Tests, Proceedings, Fracture process in Brittle Disordered Materials, Noordwijk, Holland, June 19-21.

CERVENKA, V., PUKL, R., OZBOLT, J., ELIGEHAUSEN, R. (1995), Mesh Sensitivity Effects in Smeared Finite Element Analysis of Concrete Structures, Proc. FRAMCOS 2, 1995, pp 1387-1396.

CERVENKA, V., PUKL, R. (1992) - Computer Models of Concrete Structures, Structural Engineering International, Vol.2, No.2, May 1992. IABSE Zürich, Switzerland, ISSN 1016-8664, pp.103-107.

CERVENKA, V., PUKL, R., OZBOLT, J., ELIGEHAUSEN, R. (1995) - Mesh Sensitivity Effects in Smeared Finite Element Analysis of Concrete Fracture, Proceedings of FRAMCOS2, Zurich, Aedificatio.

CERVENKA, V., CERVENKA, J. (1996) - Computer Simulation as a Design Tool for Concrete Structures, ICCE-96, proceedings of The second International Conference in Civil Engineering on Computer Applications Research and Practice, 6-8 April, Bahrain.

CERVENKA, J, CERVENKA, V., ELIGEHAUSEN, R. (1998), Fracture-Plastic Material Model for Concrete, Application to Analysis of Powder Actuated Anchors, Proc. FRAMCOS 3, 1998, pp 1107-1116.

ČERVENKA, J., BAŽANT Z.P., WIERER, M., (2004), `Equivalent Localization Element for Crack Band Approach to Mesh Sensitivity in Microplane Model', submitted for publication, *Int. J. for Num. Methods in Engineering.*

ČERVENKA, J., PRYL, D., (2007), `Fatigue Modelling of Crack Growth by Finite Element Method and Smeared Crack Approach', Internal Report 2007-08-03-2002-DP, *Cervenka Consulting.*

CRISFIELD, M.A., WILLS, J. (1989)- The Analysis of Reinforced Concrete Panels Using Different Concrete Models, Jour. of Engng. Mech., ASCE, Vol 115, No 3, March, pp.578-597.

CRISFIELD, M.A. (1983) - An Arc-Length Method Including Line Search and Accelerations, International Journal for Numerical Methods in Engineering, Vol.19,pp.1269-1289.

CHEN, W.F, SALEEB, A.F. (1982) - Constitutive Equations For Engineering Materials, John Willey \& Sons, ISBN 0-471-09149-9.

DARWIN, D., PECKNOLD, D.A.W. (1974) - Inelastic Model for Cyclic Biaxial Loading of Reinforced Concrete, Civil Engineering Studies, University of Illinois, July.

DE BORST, R. (1986), *Non-linear analysis of frictional materials*, Ph.D. Thesis**,** Delft University of Technology, 1986.

DRUCKER, D.C., PRAGER, W., Soil Mechanics and Plastic Analysis or Limit Design, Q. Appl. Math., 1952, 10(2), pp 157-165.

DYNGELAND, T. (1989) - Behavior of Reinforced Concrete Panels, Dissertation, Trondheim University, Norway, BK-report 1989:1

FEENSTRA, P.H., *Computational Aspects of Bi-axial Stress in Plain and Reinforced Concrete.* Ph.D. Thesis, Delft University of Technology, 1993.

FEENSTRA, P.H., ROTS, J.G., AMESEN, A., TEIGEN, J.G., HOISETH, K.V., A 3D Constitutive Model for Concrete Based on Co-rotational concept. *Proc. EURO-C 1998*, **1**, pp. 13-22.

Taerwe, L., and Matthys, S. (2013), *Fib model code for concrete structures 2010,* Wilhelm Ernst & Sohn, Berlin, Germany, ISBN 978-3-433-03061-5.

fib Model Code for Concrete Structures 2010, (2013), Wilhelm Ernst & Sohn, Berlin, Germany, ISBN 978-3-433-03061-5.

ETSE, G., *Theoretische und numerische Untersuchung zum diffusen und lokalisierten Versagen in Beton*, Ph.D. Thesis, University of Karlsruhe 1992.

FELIPPA, C. (1966) - Refined Finite Element Analysis of Linear and Nonlinear Two-Dimensional Structures, Ph.D. Dissertation, University of California, Engineering, pp.41-50.

GRASSL, P., LUNDGREN, K., and GYLLTOFT, K. (2002) "Concrete in compression: A plasticity theory with a novel hardening law", *International Journal of Solids and Structures*, **39**(20), 5205-5223.

VAN GYSEL, A., and TAERWE, L. (1996) "Analytical formulation of the complete stress-strain curve for high strength concrete", *Materials and Structures*, RILEM, **29**(193), 529-533.

HARTL, G. (1977) "Die Arbeitlinie Eingebetete Staehle bei erst und kurz=Belastung", Dissrtation, Univbersitaet Innsbruck

HORDIJK, D.A. (1991) - Local Approach to Fatigue of Concrete, Doctor dissertation, Delft University of Technology, The Netherlands, ISBN 90/9004519-8.

Juhász, K. P. (2013) "Modified fracture energy method for fibre reinforced concrete", Fibre Concrete 2013, Prague, Czech Republic, pp. 89-90, ISBN 978-80-01-05238-9.

KABELE, P. (2002) - Equivalent Continuum Model of Multiple Cracking, Engineering Mechanics 2002, 9 (1/2), pp.75-90, Assoc.for Engineering Mechanics, Czech Republic

KESSLER-KRAMER, CH., (2002) "Zugverhalten von Beton unter Ermüdungsbeanspruchung", Schriftenreihe des Instituts für Massivbau und Baustofftechnologie, Heft 49, Karlsruhe.

KLAUSEN, D. (1978), Festigkeit und Schadigung von Beton bei haufig wiederholter Beanschpruchung, PhD Thesis, University of Technology Darmstadt, 85 pp.

KOLLEGGER, J. - MEHLHORN, G. (1988) - Experimentelle und Analytische Untersuchungen zur Aufstellung eines Materialmodels für Gerissene Stahbetonscheiben, Nr.6 Forschungsbericht, Massivbau, Gesamthochschule Kassel.

KOLMAR, W. (1986) - Beschreibung der Kraftuebertragung über Risse in nichtlinearen Finite-Element-Berechnungen von Stahlbetontragwerken", Dissertation, T.H. Darmstadt, p. 94.

KUPFER, H., HILSDORF, H.K., RÜSCH, H. (1969) - Behavior of Concrete under Biaxial Stress, Journal ACI, Proc. V.66, No.8, Aug., pp.656-666.

MARGOLDOVA, J., CERVENKA V., PUKL R. (1998), Applied Brittle Analysis*, Concrete Eng. International*, November/December 1998.

MENETREY, P., WILLAM, K.J. (1995), Triaxial failure criterion for concrete and its generalization. *ACI, Structural Journal*, 1995, 92(3), pp 311-318.

MENETREY, Ph., WALTHER, R., ZIMMERMAN, Th., WILLAM, K.J., REGAN, P.E. Simulation of punching failure in reinforced concrete structures. *Journal of Structural Engineering*, 1997, 123(5)**,** pp 652-659.

MIER J.G.M van (1986) - Multiaxial Strain-softening of Concrete, Part I: fracture, *Materials and Structures*, RILEM, Vol. 19, No.111.

MINER M.A. (1945), Cumulative damage in fatigue. *Transactions of the American Society of Mechanical Engineering*, 67:A159-A164.

OLIVIER, J., A Consistent Characteristic Length for Smeared Cracking Models*, Int. J. Num. Meth. Eng*., 1989, **28**, pp 461-474.

OWEN, J.M., FIGUEIRAS, J.A., DAMJANIC, F., Finite Element Analysis of Reinforced and Pre-stressed concrete structures including thermal loading, *Comp. Meth. Appl. Mech. Eng.*, 1983, 41, pp 323-366.

PALMGREN, A. (1924), Die Lebensdauer von Kugellagern. *Zeitschrift Verein Deutscher Ingenieure*, 68(14):339-341.

PAPANIKOLAOU, V.K., and KAPPOS, A.J. (2007) "Confinement-sensitive plasticity constitutive model for concrete in triaxial compression", *International Journal of Solids and Structures,* **44**(21), 7021-7048.

PRAMONO, E, WILLAM, K.J., Fracture Energy-Based Plasticity Formulation of Plain Concrete, *ASCE-JEM*, 1989, 115, pp 1183-1204.

PRYL, D., CERVENKA, J., and PUKL, R. (2010) "Material model for finite element modelling of fatigue crack growth in concrete", *Procedia Engineering*, 2 (2010) 203–212.

PRYL, D., PUKL, R., and CERVENKA, J. (2013) "Modelling high-cycle fatigue of concrete specimens in three-point bending", Life-Cycle and Sustainability of Civil Infrastructure Systems (Eds. Strauss, Frangopol & Bergmeister) 1303–1306.

PRYL, D., MIKOLÁŠKOVÁ, J., PUKL, R. (2014) "Modeling Fatigue Damage of Concrete", *Key Engineering Materials*, 2014 Vols. 577-578, pp. 385-388, ISSN: 1662-9795

RAMM, E. (1981) - Strategies for Tracing Non- linear Responses Near Limit Points, Non- linear Finite Element Analysis in Structural Mechanics, (Eds. W.Wunderlich, E.Stein, K.J.Bathe)

RASHID, Y.R. (1968), Ultimate Strength Analysis of Pre-stressed Concrete Pressure Vessels*, Nuclear Engineering and Design*,1968, 7, pp 334-344.

ROTS, J.G., BLAAUWENDRAAD, J., Crack models for concrete: discrete or smeared? Fixed, multi-directional or rotating? *HERON* 1989, 34(1).

SAE, AE-4, Fatigue Design Handbook

SIMO, J.C., JU, J.W., Strain and Stress-based Continuum Damage Models-I. Formulations, II-Computational Aspects, *Int. J. Solids Structures*, 1987, 23(7)**,** pp 821-869.

SIMO, J.C., KENNEDY, J.G., GOVINDJEE, S., (1988), Non-smooth Multi-surface Plasticity and Visco-plasticity. Loading/unloading Conditions and Numerical Algorithms, *Int. J. Num. Meth. Eng*., **26**, pp 2161–2185.

TAYLOR, G.I., (1938), 'Plastic strain in metal', *J. Inst. Metals*, 62, 307-324.

UIJL, J. den, BIGAJ, A. J. (1996): A Bond Model for Ribbed Bars Based on Concrete Confinement. *Heron*, Vol.41, No.3.

VECCHIO, F.J., COLLINS, M.P (1986)- Modified Compression-Field Theory for Reinforced Concrete Beams Subjected to Shear, ACI Journal, Proc. V.83, No.2, Mar-Apr., pp 219-231.

VOS, E. (1983) - Influence of Loading Rate and Radial Pressure on Bond in Reinforced Concrete, Dissertation, Delft University, pp.219-220.

WILKINS, M.L., Calculation of Elastic-Plastic Flow, *Methods of Computational Physics*, **3**, Academic Press, New York, 1964.

DODD, L. L., and J. I. RESTREPO-POSADA. "Model for predicting cyclic behavior of reinforcing steel." Journal of structural engineering 121.3 (1995): 433-445.

KIM, SE-HYUNG. Cyclic uniaxial constitutive model for steel reinforcement. Diss. Virginia Tech, 2015.

3 FINITE ELEMENTS

3.1 Introduction

The preceding chapters dealt with the general formulation of the problem, geometric and constitutive equations. All expressions were derived independently of the structural shape, the finite elements used etc. Here, an information about finite elements currently implemented in ATENA is given.

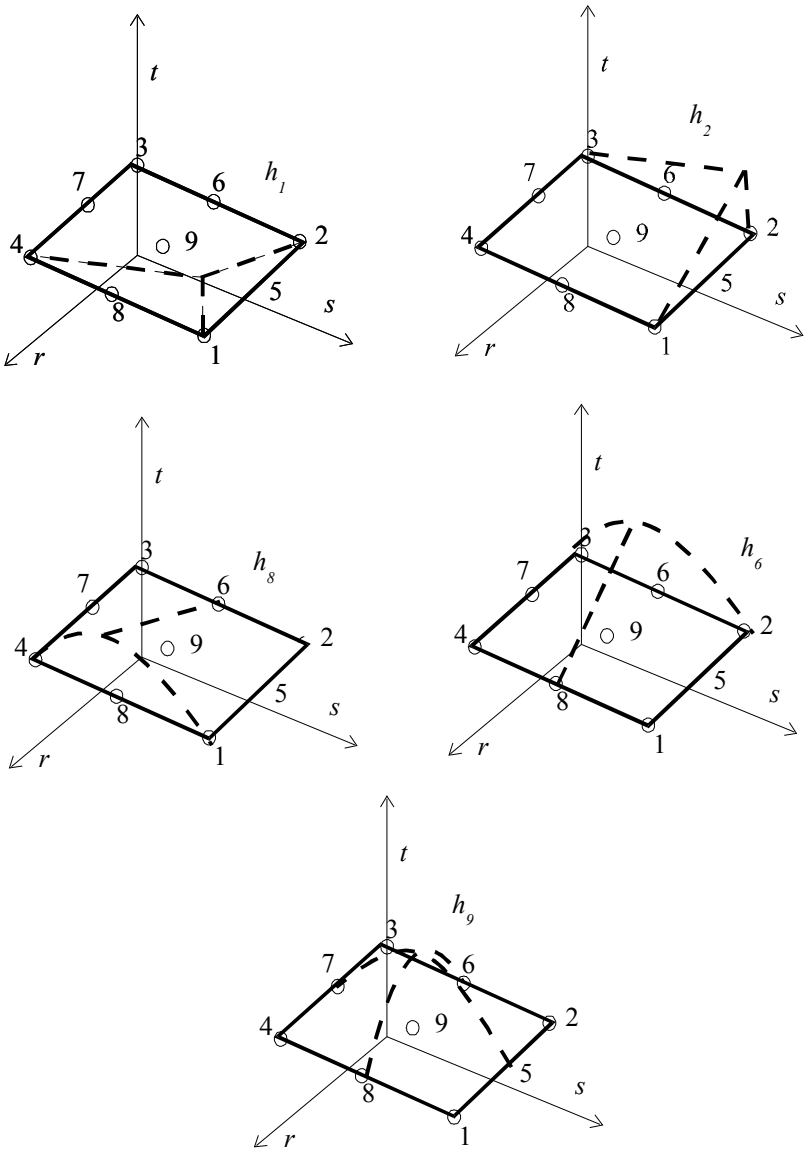


Fig. 3-1 Examples of interpolation function for plane quadrilateral elements.

The available elements can be divided into three groups: plane elements for 2D, 3D and axi-symmetric analysis, solid 3D elements and special elements, which comprises elements for modeling external cable, springs, gaps etc.

With few exceptions all elements implemented in ATENA are constructed using isoparametric formulation with linear and/or quadratic interpolation functions. The isoparametric formulation of one-, two- and three-dimensional elements belong to the "classic" element formulations. This

is not because of its superior properties, but since it is a versatile and general approach with no hidden difficulties and, also very important, these elements are easy to understand. This is very important particularly in nonlinear analysis. For example, it is highly undesirable to add element-related problems to problems related to e.g. material modeling.

Big advantage of ATENA isoparametric elements is that their interpolation functions $h_i(r,s,t)$ are constructed in hierarchical manner. Take an example of plane quadrilateral element. Some of its interpolation functions are depicted in Fig. 3-1. The 1st four functions, i.e. functions $h_1(r,s,t)$ to $h_4(r,s,t)$ has to be always present in the interpolation set, (to ensure bilinear approximation). Then, any additional function $h_6(r,s,t)$ through $h_9(r,s,t)$ can be added independently. This would involve adding the new function itself and amendments to the already present interpolation functions. This approach (and use of C++ templates) makes possible that one element formulation generates quadrilateral elements with nodes (1,2,3,4), (1,2,3,4,5), (1,2,3,4,6), ... (1,2,3,4,8), (1,2,3,4,9), (1,2,3,4,5,6), (1,2,3,4,5, 7), ... (1,2,3,8,9), ... (1,2,3,4,5,6,7,8,9). Additional mid-side points are particularly useful for changing mesh density, (i.e. element size), see Fig. 3-2, as they allow change of mesh density without need triangular elements.

Although the concept of hierarchical elements was described for plane quadrilateral elements, in ATENA it applies for plane triangular elements, 3D bricks, tetrahedral and wedge elements, too. Always there is a set of basic interpolation function that can be extended by any "higher" interpolation function.

Apart of interpolation functions finite element properties depend strongly on numerical integration scheme used to integrate element stiffness matrix, element nodal forces etc. In Atena, majority of elements are integrated by Gauss integration scheme that ensure $n(n-1)$ order accuracy, where n is degree of the polynomial used to approximate the integrated function.

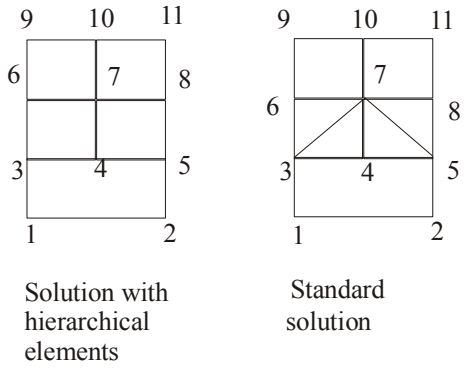


Fig. 3-2 Change of finite element mesh density.

3.2 Truss 2D and 3D Element

2D and 3D truss elements in ATENA are coded in group of elements CCIsoTruss<xx> ... CCIsoTruss<xxx>. The string in < > describes present element nodes, (see *Atena Input File Format* document for more information). These are isoparametric elements integrated by Gauss integration at 1 or 2 integration points for the case of linear or quadratic interpolation, i.e. for elements with 2 or 3 element nodes, respectively. They are suitable for plane 2D as well as 3D analysis problems. Geometry, interpolation functions and integration points of the elements are given in Fig. 3-3, Table 3.2-1 to Table 3.2-3.

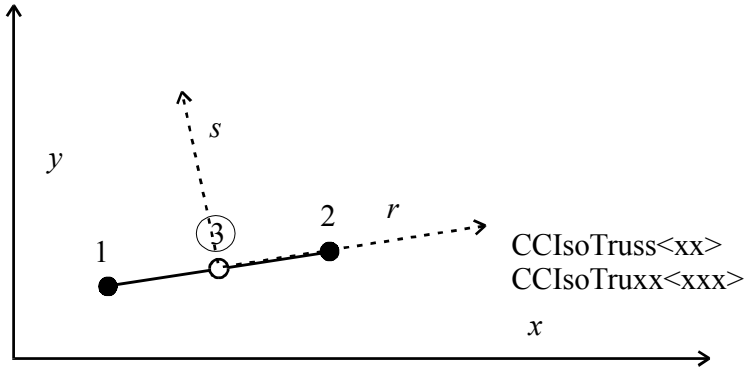


Fig. 3-3 Geometry of CCIsoTruss<...> elements.

Table 3.2-1 Interpolation functions of CCIsoTruss<...> elements.

Node i	Function h_i	Include only if node *3* is defined
1	$\frac{1}{2}(1-r)$	$-\frac{1}{2}h_3$
2	$\frac{1}{2}(1+r)$	$-\frac{1}{2}h_3$
3	$(1-r^2)$	

Table 3.2-2 Sample points for Gauss integration of 1 node CCIsoTruss<xx> element.

Integration point	Coordinate r	Weight
1	0.	2.

Table 3.2-3 Sample points for Gauss integration of 2 and 3 nodes CCIsotruss<xxx> elements.

Integration point	Coordinate *r*	Weight
1	0.577350269189626	1.
2	-0.577350269189626	1.

The element vectors and matrices for *Total Lagrangian formulation*, configuration at time t and iteration $^{(i)}$ are as follows. Note that they are equally applicable for *Updated Lagrangian formulation* upon applying changes related to the element reference coordinate system (undeformed vs. deformed element axis.). The formulation is present for 3-nodes element option. The 2-nodes variant is obtained by simply neglecting the terms for the element mid-point.

An arbitrary point on the truss element has at reference time t coordinates ${}^{t}\underline{X} = [{}^{t}x_1, {}^{t}x_1, {}^{t}x_1]$:

$$
\begin{aligned}
{}^{t}x_1 &= {}^{t}x_1^1 h_1 + {}^{t}x_1^2 h_2 + {}^{t}x_1^3 h_3 \\
{}^{t}x_2 &= {}^{t}x_2^1 h_1 + {}^{t}x_2^2 h_2 + {}^{t}x_2^3 h_3 \\
{}^{t}x_3 &= {}^{t}x_3^1 h_1 + {}^{t}x_3^2 h_2 + {}^{t}x_3^3 h_3
\end{aligned}
\tag{3.1}
$$

At time $t + \Delta t^{(i-1)}$ the same point has coordinates ${}^{t+\Delta t}X^{(i-1)}$:

$$
\begin{aligned}
{}^{t+\Delta t}x_1^{(i-1)} &= ({}^{t}x_1^1 + {}^{t}u_1^{1(i-1)}) h_1 + ({}^{t}x_1^2 + {}^{t}u_1^{2(i-1)}) h_2 + ({}^{t}x_1^3 + {}^{t}u_1^{3(i-1)}) h_3 \\
{}^{t+\Delta t}x_2^{(i-1)} &= ({}^{t}x_2^1 + {}^{t}u_2^{1(i-1)}) h_1 + ({}^{t}x_2^2 + {}^{t}u_2^{2(i-1)}) h_2 + ({}^{t}x_2^3 + {}^{t}u_2^{3(i-1)}) h_3 \\
{}^{t+\Delta t}x_3^{(i-1)} &= ({}^{t}x_3^1 + {}^{t}u_3^{1(i-1)}) h_1 + ({}^{t}x_3^2 + {}^{t}u_3^{2(i-1)}) h_2 + ({}^{t}x_3^3 + {}^{t}u_3^{3(i-1)}) h_3
\end{aligned}
\tag{3.2}
$$

and at time $t + \Delta t^{(i)}$ coordinates ${}^{t+\Delta t}X^{(i)}$

$$
\begin{aligned}
{}^{t+\Delta t}x_1^{(i)} &= ({}^{t}x_1^1 + {}^{t}u_1^{1(i)}) h_1 + ({}^{t}x_1^2 + {}^{t}u_1^{2(i)}) h_2 + ({}^{t}x_1^3 + {}^{t}u_1^{3(i)}) h_3 \\
{}^{t+\Delta t}x_2^{(i)} &= ({}^{t}x_2^1 + {}^{t}u_2^{1(i)}) h_1 + ({}^{t}x_2^2 + {}^{t}u_2^{2(i)}) h_2 + ({}^{t}x_2^3 + {}^{t}u_2^{3(i)}) h_3 \\
{}^{t+\Delta t}x_3^{(i)} &= ({}^{t}x_3^1 + {}^{t}u_3^{1(i)}) h_1 + ({}^{t}x_3^2 + {}^{t}u_3^{2(i)}) h_2 + ({}^{t}x_3^3 + {}^{t}u_3^{3(i)}) h_3
\end{aligned}
\tag{3.3}
$$

Increment of Green Lagrange strain ${}_{t}\varepsilon_{11}^{(i)} = {}_{t}^{t+\Delta t}\varepsilon_{11}^{(i)t+\Delta t} - {}_{t}^{t+\Delta t}\varepsilon_{11}^{(i-1)}$ (at time $t + \Delta t$, iteration $^{(i)}$ with to configuration at time t) is calculated:

$$ {}_{t}\varepsilon_{11}^{(i)} = \frac{1}{2}\left(\frac{\left(\frac{\partial^{t+\Delta t} l^{(i)}}{\partial r} \right)^2 - \left(\frac{\partial^{t+\Delta t} l^{(i-1)}}{\partial r} \right)^2}{\left(\frac{\partial^{t} l}{\partial r} \right)^2} \right) \tag{3.4} $$

where truss length differentials are

$$ \left(\frac{\partial^{t} l}{\partial r} \right)^2 = \left(\frac{\partial^{t} x_1}{\partial r} \right)^2 + \left(\frac{\partial^{t} x_2}{\partial r} \right)^2 + \left(\frac{\partial^{t} x_3}{\partial r} \right)^2 $$

$$ \left(\frac{\partial^{t+\Delta t} l^{(i-1)}}{\partial r} \right)^2 = \left(\frac{\partial^{t+\Delta t} x_1^{(i-1)}}{\partial r} \right)^2 + \left(\frac{\partial^{t+\Delta t} x_2^{(i-1)}}{\partial r} \right)^2 + \left(\frac{\partial^{t+\Delta t} x_3^{(i-1)}}{\partial r} \right)^2 \tag{3.5} $$

$$ \left(\frac{\partial^{t+\Delta t} l^{(i)}}{\partial r} \right)^2 = \left(\frac{\partial^{t+\Delta t} x_1^{(i)}}{\partial r} \right)^2 + \left(\frac{\partial^{t+\Delta t} x_2^{(i)}}{\partial r} \right)^2 + \left(\frac{\partial^{t+\Delta t} x_3^{(i)}}{\partial r} \right)^2 $$

Substituting (3.5), (3.3) into (3.4) after some math manipulation it can be derived:

$$ {}_{t}^{t+\Delta t}\boldsymbol{B}_{L0} = \frac{1}{\left(\frac{\partial^{t} l}{\partial r} \right)^2} \begin{bmatrix} \frac{\partial h_1}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_1^1 + \frac{\partial h_1}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_1^2 + \frac{\partial h_1}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_1^3 \\ \frac{\partial h_2}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_1^1 + \frac{\partial h_2}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_1^2 + \frac{\partial h_2}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_1^3 \\ \frac{\partial h_3}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_1^1 + \frac{\partial h_3}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_1^2 + \frac{\partial h_3}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_1^3 \\ \frac{\partial h_1}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_2^1 + \frac{\partial h_1}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_2^2 + \frac{\partial h_1}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_2^3 \\ \frac{\partial h_2}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_2^1 + \frac{\partial h_2}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_2^2 + \frac{\partial h_2}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_2^3 \\ \frac{\partial h_3}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_2^1 + \frac{\partial h_3}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_2^2 + \frac{\partial h_3}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_2^3 \\ \frac{\partial h_1}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_3^1 + \frac{\partial h_1}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_3^2 + \frac{\partial h_1}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_3^3 \\ \frac{\partial h_2}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_3^1 + \frac{\partial h_2}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_3^2 + \frac{\partial h_2}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_3^3 \\ \frac{\partial h_3}{\partial r}\frac{\partial h_1}{\partial r}{}^{t}x_3^1 + \frac{\partial h_3}{\partial r}\frac{\partial h_2}{\partial r}{}^{t}x_3^2 + \frac{\partial h_3}{\partial r}\frac{\partial h_3}{\partial r}{}^{t}x_3^3 \end{bmatrix} \tag{3.6} $$

$$
{}^{t+\Delta t}_{t}\boldsymbol{B}^{(i-1)}_{L1} = \frac{1}{\left(\frac{\partial^t l}{\partial r}\right)^2}
\begin{bmatrix}
\frac{\partial h_1}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_1^{1(i-1)} + \frac{\partial h_1}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_1^{2(i-1)} + \frac{\partial h_1}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_1^{3(i-1)} \\
\frac{\partial h_2}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_1^{1(i-1)} + \frac{\partial h_2}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_1^{2(i-1)} + \frac{\partial h_2}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_1^{3(i-1)} \\
\frac{\partial h_3}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_1^{1(i-1)} + \frac{\partial h_3}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_1^{2(i-1)} + \frac{\partial h_3}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_1^{3(i-1)} \\
\frac{\partial h_1}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_2^{1(i-1)} + \frac{\partial h_1}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_2^{2(i-1)} + \frac{\partial h_1}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_2^{3(i-1)} \\
\frac{\partial h_2}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_2^{1(i-1)} + \frac{\partial h_2}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_2^{2(i-1)} + \frac{\partial h_2}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_2^{3(i-1)} \\
\frac{\partial h_3}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_2^{1(i-1)} + \frac{\partial h_3}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_2^{2(i-1)} + \frac{\partial h_3}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_2^{3(i-1)} \\
\frac{\partial h_1}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_3^{1(i-1)} + \frac{\partial h_1}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_3^{2(i-1)} + \frac{\partial h_1}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_3^{3(i-1)} \\
\frac{\partial h_2}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_3^{1(i-1)} + \frac{\partial h_2}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_3^{2(i-1)} + \frac{\partial h_2}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_3^{3(i-1)} \\
\frac{\partial h_3}{\partial r}\frac{\partial h_1}{\partial r}\,{}^{t+\Delta t}u_3^{1(i-1)} + \frac{\partial h_3}{\partial r}\frac{\partial h_2}{\partial r}\,{}^{t+\Delta t}u_3^{2(i-1)} + \frac{\partial h_3}{\partial r}\frac{\partial h_3}{\partial r}\,{}^{t+\Delta t}u_3^{3(i-1)}
\end{bmatrix}
\tag{3.7}
$$

and

$$
{}^{t+\Delta t}_{t}\boldsymbol{B}^{(n-1)}_{NL} = \frac{1}{\left(\frac{\partial^t l}{\partial r}\right)}
\begin{bmatrix}
\frac{\partial h_1}{\partial r} & 0 & 0 & \frac{\partial h_2}{\partial r} & 0 & 0 & \frac{\partial h_3}{\partial r} & 0 & 0 \\
0 & \frac{\partial h_1}{\partial r} & 0 & 0 & \frac{\partial h_2}{\partial r} & 0 & 0 & \frac{\partial h_3}{\partial r} & 0 \\
0 & 0 & \frac{\partial h_1}{\partial r} & 0 & 0 & \frac{\partial h_2}{\partial r} & 0 & 0 & \frac{\partial h_3}{\partial r}
\end{bmatrix}
\tag{3.8}
$$

The 2nd Piola-Kirchhoff stress matrix and tensor are:

$$
{}^{t+\Delta t}_{t}S^{(i-1)} = \begin{bmatrix} {}^{t+\Delta t}_{t}S^{(i-1)}_{11} & 0 & 0 \\ 0 & {}^{t+\Delta t}_{t}S^{(i-1)}_{11} & 0 \\ 0 & 0 & {}^{t+\Delta t}_{t}S^{(i-1)}_{11} \end{bmatrix}, \quad {}^{t+\Delta t}_{t}\underline{S}^{(i-1)} = \left[{}^{t+\Delta t}_{t}S^{(i-1)}_{11}\right] \tag{3.9}
$$

The formulation is completed by relationship for element deformation gradient ${}^{t+\Delta t}_{t}X^{(i)}_{1,1}$, which yields:

$$
{}^{t+\Delta t}_{t}X^{(i)}_{1,1} = \frac{\left(\frac{\partial^{t+\Delta t} l^{(i)}}{\partial r}\right)}{\left(\frac{\partial^t l}{\partial r}\right)} \tag{3.10}
$$

Note that 2-nodes truss element has constant strains along its length and thus the increment of Green Lagrange strain can be calculated directly, (i.e. not using differentials truss length as it was the case of (3.4)):

$$ {}_t\varepsilon_{11}^{(i)} = \frac{1}{2}\left(\frac{\left({}^{t+\Delta t}l^{(i)}\right)^2 - \left({}^{t+\Delta t}l^{(i-1)}\right)^2}{{}^{t}l^2} \right) \tag{3.11} $$

This yields a bit simpler element formulation (with the same results). However, for the sake of preserving unified approach to all truss elements, ATENA uses even in this case the equation (3.4).

3.3 Plane Quadrilateral Elements

Plane quadrilateral elements in ATENA are coded in group of elements CCIsoQuad<xxxx> ... CCIsoQuad<xxxxxxxxx>. The string in < > describes present element nodes (see Atena Input File Format document for more information). These are isoparametric elements integrated by Gauss integration at 4 or 9 integration points for the case of bilinear or bi-quadratic interpolation, i.e. for elements with 4 or 5 and more element nodes, respectively. They are suitable for plane 2D, axisymmetric and 3D problems.

CCIsoQuad2_5<...> elements present a simplified 3D formulation of the CCIsoQuad<...> elements. Their higher execution performance is achieved at cost of omitting some nonlinear terms, see below.

Geometry, interpolation functions and integration points of the elements are given in Fig. 3-4 and in the subsequent tables.

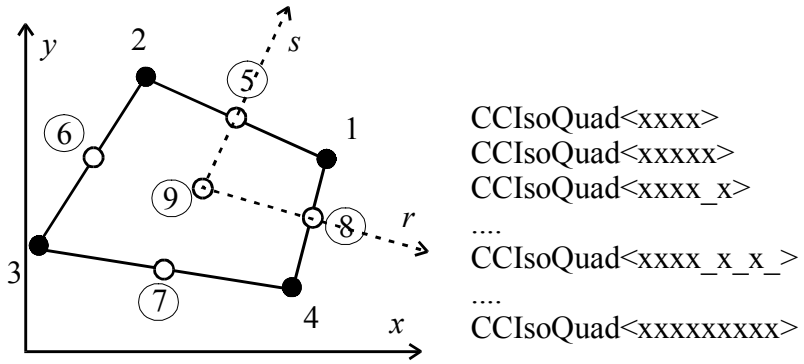


Fig. 3-4 Geometry of CCIsoQuad<...> elements.

Table 3.3-1: Interpolation functions of CCIsoQuad<...> elements.

Node i	Function h_i	Include only if node i is defined				
		$i = 5$	$I = 6$	$i = 7$	$i = 8$	$i = 9$
1	$\frac{1}{4}(1+r)(1+s)$	$-\frac{1}{2}h_5$			$-\frac{1}{2}h_8$	$\frac{1}{4}h_9$
2	$\frac{1}{4}(1-r)(1+s)$	$-\frac{1}{2}h_5$	$-\frac{1}{2}h_6$			$\frac{1}{4}h_9$
3	$\frac{1}{4}(1-r)(1-s)$		$-\frac{1}{2}h_6$	$-\frac{1}{2}h_7$		$\frac{1}{4}h_9$
4	$\frac{1}{4}(1+r)(1-s)$			$-\frac{1}{2}h_7$	$-\frac{1}{2}h_8$	$\frac{1}{4}h_9$
5	$\frac{1}{2}(1-r^2)(1+s)$					$-\frac{1}{2}h_9$
6	$\frac{1}{2}(1-s^2)(1-r)$					$-\frac{1}{2}h_9$
7	$\frac{1}{2}(1-r^2)(1-s)$					$-\frac{1}{2}h_9$
8	$\frac{1}{2}(1+r)(1-s^2)$					$-\frac{1}{2}h_9$
9	$\frac{1}{2}(1-r^2)(1-s^2)$					

Table 3.3-2: Sample points for Gauss integration of 4 nodes CCIsoQuad<...> element.

Integration point	Coordinate r	Coordinate s	Weight
1	0.577350269189626	0.577350269189626	1.
2	0.577350269189626	-0.577350269189626	1.
3	-0.577350269189626	0.577350269189626	1.
4	-0.577350269189626	-0.577350269189626	1.

Table 3.3-3: Sample points for Gauss integration 5 to 9 nodes CCIsoQuad<...> elements.

Integration point	Coordinate *r*	Coordinate *s*	Weight
1	0.774596669241483	0.774596669241483	0.3086419753
2	0.774596669241483	0.	0.4938271605
3	0.774596669241483	-0.774596669241483	0.3086419753
4	0.	0.774596669241483	0.4938271605
5	0.	0.	0.7901234568
6	0.	-0.774596669241483	0.4938271605
7	-0.774596669241483	0.774596669241483	0.3086419753
8	-0.774596669241483	0.	0.4938271605
9	-0.774596669241483	-0.774596669241483	0.3086419753

Equations (3.12) through (3.21) present CCIsoQuad<...> axisymmetric element formulation. 2D element formulation is simply obtained by removing terms associated with circumferential strains and stresses ${}^{t+\Delta t}_{t}\varepsilon^{(i)}_{33}$, ${}^{t+\Delta t}_{t}S^{(i)}_{33}$.

Incremental strains:

$$ {}_{t}\varepsilon^{(i)}_{11} = {}_{t}u^{(i)}_{1,1} + {}^{t+\Delta t}_{t}u^{(i-1)}_{1,1}\ {}_{t}u^{(i)}_{1,1} + {}^{t+\Delta t}_{t}u^{(i-1)}_{2,1}\ {}_{t}u^{(i)}_{2,1} + \frac{1}{2}\left(\left({}_{t}u^{(i)}_{1,1}\right)^2 + \left({}_{t}u^{(i)}_{2,1}\right)^2\right) $$

$$ {}_{t}\varepsilon^{(i)}_{22} = {}_{t}u^{(i)}_{2,2} + {}^{t+\Delta t}_{t}u^{(i-1)}_{1,2}\ {}_{t}u^{(i)}_{1,2} + {}^{t+\Delta t}_{t}u^{(i-1)}_{2,2}\ {}_{t}u^{(i)}_{2,2} + \frac{1}{2}\left(\left({}_{t}u^{(i)}_{1,2}\right)^2 + \left({}_{t}u^{(i)}_{2,2}\right)^2\right) $$

$$ \begin{aligned} {}_{t}\varepsilon^{(i)}_{12} = {} & \frac{1}{2}\left({}_{t}u^{(i)}_{1,2} + {}_{t}u^{(i)}_{2,1}\right) + \\ & \frac{1}{2}\left({}^{t+\Delta t}_{t}u^{(i-1)}_{1,1}\ {}_{t}u^{(i)}_{1,2} + {}^{t+\Delta t}_{t}u^{(i-1)}_{2,1}\ {}_{t}u^{(i)}_{2,2} + {}^{t+\Delta t}_{t}u^{(i-1)}_{1,2}\ {}_{t}u^{(i)}_{1,1} + {}^{t+\Delta t}_{t}u^{(i-1)}_{2,2}\ {}_{t}u^{(i)}_{2,1}\right) + \\ & \frac{1}{2}\left({}_{t}u^{(i)}_{1,1}\ {}_{t}u^{(i)}_{1,2} + {}_{t}u^{(i)}_{2,1}\ {}_{t}u^{(i)}_{2,2}\right) \end{aligned} $$

$$ {}_{t}\varepsilon^{(i)}_{33} = \frac{u^{(i)}_{1}}{{}^{t}x_{1}} + \frac{{}^{t+\Delta t}u^{(i)}_{1}\,u^{(i)}_{1}}{\left({}^{t}x_{1}\right)^2} + \frac{1}{2}\left(\frac{u^{(i)}_{1}}{{}^{t}x_{1}}\right)^2 \qquad (3.12) $$

Displacement derivatives:

$$ {}_{t}u_{i,j}^{(i)} = \frac{\partial\left({}^{t+\Delta t}u_i^{(i)} - {}^{t+\Delta t}u_i^{(i-1)}\right)}{\partial {}^{t}x_j} $$

$$ {}_{t}^{t+\Delta t}u_{i,j}^{(i-1)} = \frac{\partial {}^{t+\Delta t}u_i^{(i-1)}}{\partial {}^{t}x_j} \tag{3.13} $$

Strains and matrices to calculate them:

$$ {}_{t}\underline{\varepsilon}^{(i)} = {}_{t}^{t+\Delta t}\boldsymbol{B}_L^{(i-1)}\,\Delta\underline{U}^{(i)} $$

$$ {}_{t}\underline{\varepsilon}^{(i)} = \left[{}_{t}\varepsilon_{11}^{(i)},\ {}_{t}\varepsilon_{22}^{(i)},\ 2\,{}_{t}\varepsilon_{12}^{(i)},\ {}_{t}\varepsilon_{33}^{(i)} \right] \tag{3.14} $$

$$ \Delta\underline{U}^{(i)} = {}^{t+\Delta t}\underline{U}^{(i)} - {}^{t+\Delta t}\underline{U}^{(i-1)} = \left[u_1^{1(i)}, u_2^{1(i)}, u_1^{2(i)}, u_2^{2(i)}, ...u_1^{n(i)}, u_2^{n(i)} \right] $$

Linear strain-displacement matrix:

$$ {}_{t}^{t+\Delta t}\boldsymbol{B}_L^{(i-1)} = {}_{t}^{t+\Delta t}\boldsymbol{B}_{L0} + {}_{t}^{t+\Delta t}\boldsymbol{B}_{L1}^{(i-1)} \tag{3.15} $$

Linear strain-displacement matrix – constant part:

$$ {}_{t}^{t+\Delta t}\boldsymbol{B}_{L0} = \begin{bmatrix} {}_{t}h_{1,1} & 0 & {}_{t}h_{2,1} & 0 & \dots & 0 & 0 \\ 0 & {}_{t}h_{1,2} & 0 & {}_{t}h_{2,2} & \dots & 0 & {}_{t}h_{n,2} \\ {}_{t}h_{1,2} & {}_{t}h_{1,1} & {}_{t}h_{2,2} & {}_{t}h_{2,1} & \dots & {}_{t}h_{n,2} & {}_{t}h_{n,1} \\ \dfrac{h_1}{{}_{t}\widetilde{x}_1} & 0 & \dfrac{h_2}{{}_{t}\widetilde{x}_1} & 0 & \dots & \dfrac{h_n}{{}_{t}\widetilde{x}_1} & 0 \end{bmatrix} \tag{3.16} $$

where

$$ {}_{t}h_{i,j} = \frac{\partial h_i}{\partial {}^{t}x_j} $$

$$ u_i^{(i)} = {}^{t+\Delta t}u_i^{(i)} - {}^{t+\Delta t}u_i^{(i-1)} \tag{3.17} $$

$$ {}_{t}\widetilde{x}_1 = \sum_{k=1}^{n} h_k\ {}_{t}x_1^k $$

Linear strain-displacement matrix – non-constant part:

$$
{}^{t+\Delta t}_{t}\boldsymbol{B}_{L1}^{(i-1)} = \begin{bmatrix}
{}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{1,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{1,1} & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{2,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{2,1} \\
{}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{1,2} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{1,2} & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{2,2} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{2,2} \\
{}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{1,2} + {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{1,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{1,2} + {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{1,1} & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{2,2} + {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{2,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{2,2} + {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{2,1} \\
{}^{t+\Delta t}l_{33}^{(i-1)}\,\frac{h_1}{{}_{t}\widetilde{x}_1} & 0 & {}^{t+\Delta t}l_{33}^{(i-1)}\,\frac{h_2}{{}_{t}\widetilde{x}_1} & 0
\end{bmatrix}
\begin{bmatrix}
\cdots & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{n,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{n,1} \\
\cdots & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{n,2} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{n,2} \\
\cdots & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{n,2} + {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_{t}h_{n,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{n,2} + {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_{t}h_{n,1} \\
\cdots & {}^{t+\Delta t}l_{33}^{(i-1)}\,\frac{h_n}{{}_{t}\widetilde{x}_1} & 0
\end{bmatrix}
\tag{3.18}
$$

where

$$
{}^{t+\Delta t}l_{11}^{(i-1)} = \sum_{k=1}^{n} {}_{t}h_{k,1}\ {}^{t+\Delta t}_{t}u_1^{k(i-1)}
$$

$$
{}^{t+\Delta t}l_{12}^{(i-1)} = \sum_{k=1}^{n} {}_{t}h_{k,2}\ {}^{t+\Delta t}_{t}u_1^{k(i-1)}
$$

$$
{}^{t+\Delta t}l_{21}^{(i-1)} = \sum_{k=1}^{n} {}_{t}h_{k,1}\ {}^{t+\Delta t}_{t}u_2^{k(i-1)} \tag{3.19}
$$

$$
{}^{t+\Delta t}l_{22}^{(i-1)} = \sum_{k=1}^{n} {}_{t}h_{k,2}\ {}^{t+\Delta t}_{t}u_2^{k(i-1)}
$$

$$
{}^{t+\Delta t}l_{33}^{(i-1)} = \frac{1}{{}_{t}\widetilde{x}_1}\sum_{k=1}^{n} h_k\ {}^{t+\Delta t}_{t}u_1^{k(i-1)}
$$

Nonlinear strain-displacement matrix

$$
{}^{t+\Delta t}_{t}\boldsymbol{B}_{NL}^{(i-1)} = \begin{bmatrix}
{}_{t}h_{1,1} & 0 & {}_{t}h_{2,1} & 0 & \cdots & {}_{t}h_{n,1} & 0 \\
{}_{t}h_{1,2} & 0 & {}_{t}h_{2,2} & 0 & \cdots & {}_{t}h_{n,2} & 0 \\
0 & {}_{t}h_{1,1} & 0 & {}_{t}h_{2,1} & \cdots & 0 & {}_{t}h_{n,1} \\
0 & {}_{t}h_{1,2} & 0 & {}_{t}h_{2,2} & \cdots & 0 & {}_{t}h_{n,2} \\
\frac{h_1}{{}_{t}\widetilde{x}_1} & 0 & \frac{h_2}{{}_{t}\widetilde{x}_1} & 0 & \cdots & \frac{h_n}{{}_{t}\widetilde{x}_1} & 0
\end{bmatrix}
\tag{3.20}
$$

2nd Piola-Kirchhoff stress tensor and vector

$$ {}_t\boldsymbol{S}^{(i-1)} = \begin{bmatrix} {}^{t+\Delta t}_{t}S_{11}^{(i-1)} & {}^{t+\Delta t}_{t}S_{12}^{(i-1)} & 0 & 0 & 0 \\ {}^{t+\Delta t}_{t}S_{21}^{(i-1)} & {}^{t+\Delta t}_{t}S_{22}^{(i-1)} & 0 & 0 & 0 \\ 0 & 0 & {}^{t+\Delta t}_{t}S_{11}^{(i-1)} & {}^{t+\Delta t}_{t}S_{12}^{(i-1)} & 0 \\ 0 & 0 & {}^{t+\Delta t}_{t}S_{21}^{(i-1)} & {}^{t+\Delta t}_{t}S_{22}^{(i-1)} & 0 \\ 0 & 0 & 0 & 0 & {}^{t+\Delta t}_{t}S_{33}^{(i-1)} \end{bmatrix} $$

$$ {}_t\underline{S}^{(i)} = \begin{bmatrix} {}^{t+\Delta t}_{t}S_{11}^{(i-1)} & {}^{t+\Delta t}_{t}S_{22}^{(i-1)} & {}^{t+\Delta t}_{t}S_{21}^{(i-1)} & {}^{t+\Delta t}_{t}S_{33}^{(i-1)} \end{bmatrix} \tag{3.21} $$

In case of the simplified 3D analysis, i.e. elements CCIsoQuad2_5<...>, the equations are further extended as follows:

All element matrices and vectors are computed with respect to element local coordinate system $x_{local,1}, x_{local,2}$ using equations in (3.12) through (3.21). They are transformed into 3D global coordinate system by means of simple transformation:

$$ \boldsymbol{M}_{global} = \boldsymbol{T}\ \boldsymbol{M}_{local}\ \boldsymbol{T}^{T},\ \underline{v}_{global} = \boldsymbol{T}\ \underline{v}_{local} \tag{3.22} $$

where

$\boldsymbol{M}_{global}, \boldsymbol{M}_{local}, \underline{v}_{global}, \underline{v}_{local}$ are global and local finite element matrices and vectors,

$\boldsymbol{T}$ is transformation matrix from local to global coordinate system:

$$ \boldsymbol{T} = \begin{bmatrix} \cos(x_{local,1}, x_{global,1}), \cos(x_{local,2}, x_{global,1}) \\ \cos(x_{local,1}, x_{global,2}), \cos(x_{local,2}, x_{global,2}) \\ \cos(x_{local.1}, x_{global,3}), \cos(x_{local,2}, x_{global,3}) \end{bmatrix} \tag{3.23} $$

where:

$x_{local,i}, x_{global,i}$ are local and global coordinates (in 2D and 3D space).

The local element coordinate system (see Fig. 3-5) is defined by local $\underline{x}_{local,1}, \underline{x}_{local,2}, \underline{x}_{local,3}$ coordinates. All of them pass through origin of the global (reference) coordinate system. The axes $\underline{x}_{local,1}$ and $\underline{x}_{local,2}$ constitute a local coordinates element plane that is parallel to the element. The axis $\underline{x}_{local,3}$ is perpendicular to the element and the axis $\underline{x}_{local,1}$ is defined as a projection of global x_1 axis to the local coordinate element plane. An exception to that is, when the element is normal to the global x_1. In this case the local $\underline{x}_{local,1}$ coincides with the global x_2 axis.

The present definition of local element coordinate system depends on plane of the finite element, but it does not depend on its shape itself. This is very important property, as ATENA supports use of local (instead of global) nodal degrees of freedom and, (of course) these degrees of freedom must refer to a coordinate system common to all elements of the plane, in which they lie.

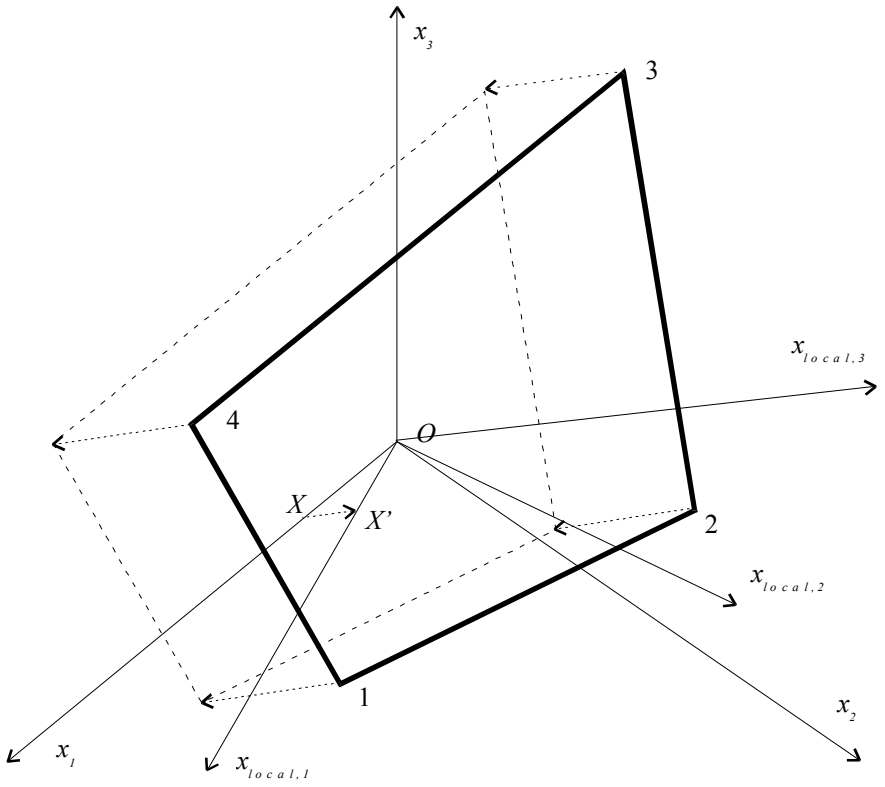


Fig. 3-5 Local plane element coordinate system.

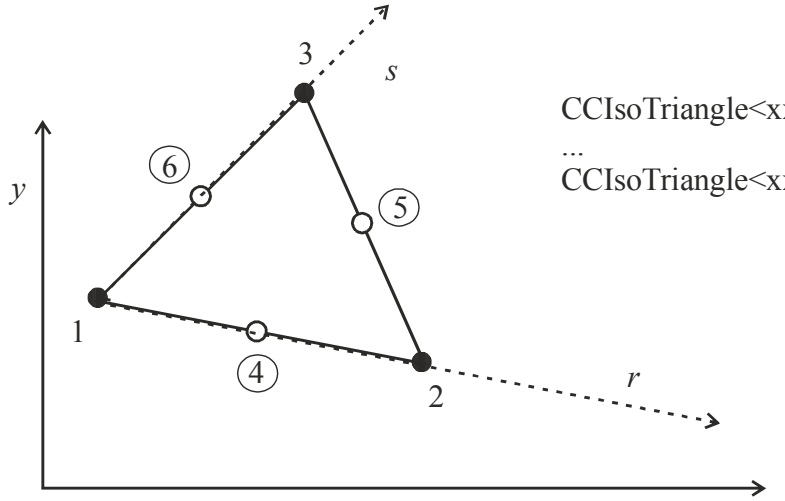
Full 3D formulation of the CCIsoQuad<...> elements is much the same as that for simplified 3D elements CCIsoQuad2_5<...>. The only difference is that the matrix ${}^{t}_{0}\boldsymbol{B}_{NL}$ will include also terms related to the „out-of-element-plane“ direction:

$$ {}^{t+\Delta t}_{t}\boldsymbol{B}^{(i-1)}_{NL} = \begin{bmatrix} {}_t h_{1,1} & 0 & 0 & {}_t h_{2,1} & 0 & 0 & {}_t h_{3,1} & 0 & 0 & \cdots & {}_t h_{N,1} & 0 & 0 \\ {}_t h_{1,2} & 0 & 0 & {}_t h_{2,2} & 0 & 0 & {}_t h_{3,2} & 0 & 0 & & {}_t h_{N,2} & 0 & 0 \\ 0 & {}_t h_{1,1} & 0 & 0 & {}_t h_{2,1} & 0 & 0 & {}_t h_{3,1} & 0 & & 0 & {}_t h_{N,1} & 0 \\ 0 & {}_t h_{1,2} & 0 & 0 & {}_t h_{2,2} & 0 & 0 & {}_t h_{3,2} & 0 & & 0 & {}_t h_{N,2} & 0 \\ 0 & 0 & {}_t h_{1,1} & 0 & 0 & {}_t h_{2,1} & 0 & 0 & {}_t h_{3,1} & & 0 & 0 & {}_t h_{N,1} \\ 0 & 0 & {}_t h_{1,2} & 0 & 0 & {}_t h_{2,2} & 0 & 0 & {}_t h_{3,2} & & 0 & 0 & {}_t h_{N,2} \end{bmatrix} \tag{3.24} $$

3.4 Plane Triangular Elements

Plane triangular elements in ATENA are coded in group of elements CCIsoTriangle<xxx> ... CCIsoTriangle<xxxxxx>. The string in < > describes present element nodes (see Atena Input File Format document for more information). These are isoparametric elements integrated by Gauss integration at 1 or 3 integration points for the case of bilinear or bi-quadratic interpolation, i.e. for elements with 3 or 4 and more element nodes, respectively. They are suitable for plane

2D, axisymmetric and 3D problems. Geometry, interpolation functions and integration points of



the elements are given in Fig. 3-6, Table 3-1, Table 3-2, and Table 3-3.

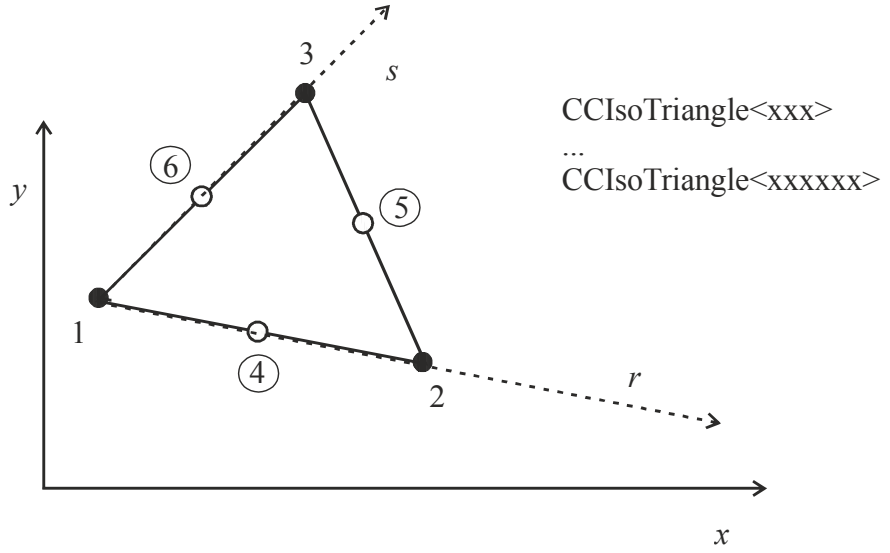


Fig. 3-6: Geometry of CCIsoTriangle<...> elements.

Table 3-1: Interpolation functions of CCIsoTriangle<...> elements.

Node i	Function h_i	Include only if node i is defined		
		$i = 4$	$i = 5$	$i = 6$
1	$1 - r - s$	$-\frac{1}{2}h_4$		$-\frac{1}{2}h_6$
2	r	$-\frac{1}{2}h_4$	$-\frac{1}{2}h_5$	
3	s		$-\frac{1}{2}h_5$	$-\frac{1}{2}h_6$

4	$4r(1-r-s)$			
5	$4rs$			
6	$4s(1-r-s)$			

Table 3-2: Sample point for Gauss integration of 3 nodes CCIsoTriangle<...> elements.

Integration point	Coordinate r	Coordinate s	Weight
1	1/3	1/3	1/2

Table 3-3: Sample points for Gauss integration of 3 to 6 nodes CCIsoTriangle<...> elements.

Integration point	Coordinate r	Coordinate s	Weight
1	1/6	1/6	1/6
2	2/3	1/6	1/6
3	1/6	2/3	1/6

All the above expressions for the formulation for plane quadrilateral elements remain valid also for the triangular elements, including the extension from 2D to simplified and full 3D analysis. The expressions only use different approximation functions $h_i(r,s,t)$ and different integration points $[r,s,t]$, see Table 3-1, Table 3-2, and Table 3-3.

3.5 3D Solid Elements

ATENA finite element library includes the following group of 3D solid elements:

tetrahedral elements CCIsoTetra<xxxx> ... CCIsoTetra<xxxxxxxxxx> with 4 to 10 nodes, see Fig. 3-7,

brick elements CCIsoBrick<xxxxxxxx> ... CCIsoBrick<xxxxxxxxxxxxxxxxxxxx> with 8 up to 20 nodes see Fig. 3-8 and

wedge elements CCIsoWedge<xxxxxx> ... CCIsoWedge<xxxxxxxxxxxxxx> with 6 to 15 nodes, see Fig. 3-9.

The string in < > describes present element nodes (see Atena Input File Format document for more information). These are isoparametric elements integrated by Gauss integration at integration points given in the following tables. Interpolation functions for all variants of the elements are also given in the tables below.

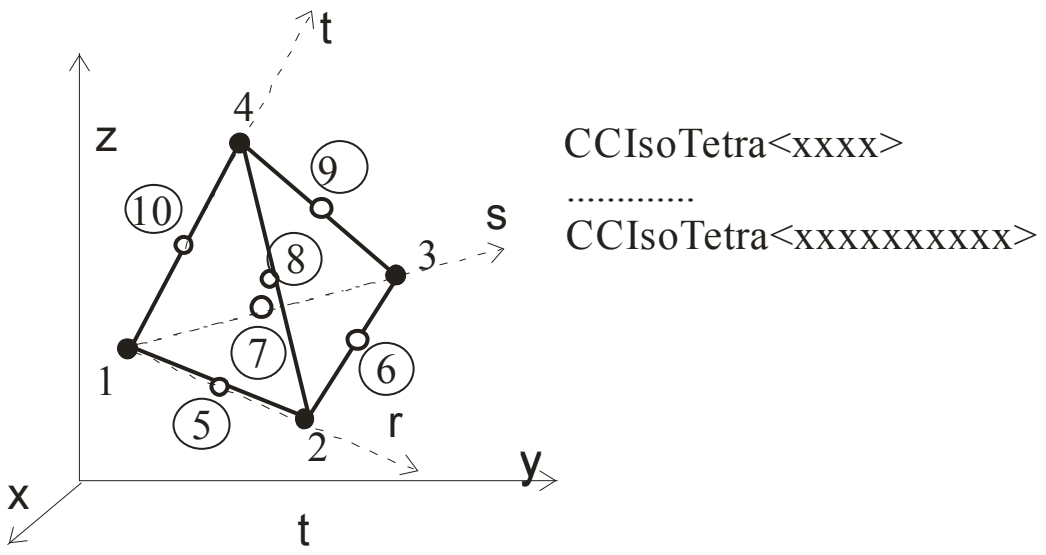


Fig. 3-7 Geometry of CCIsoTetra<...> elements.

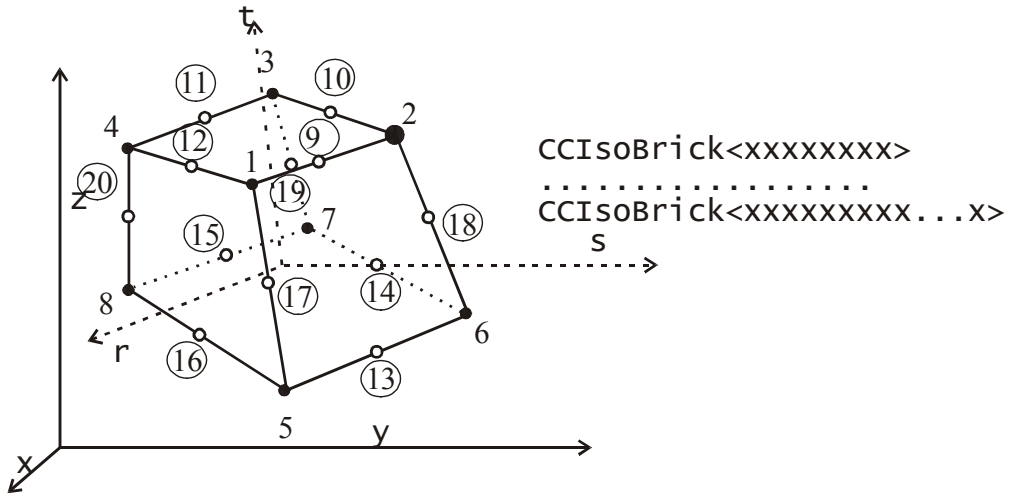


Fig. 3-8 Geometry of CCIsoBrick<...> elements.

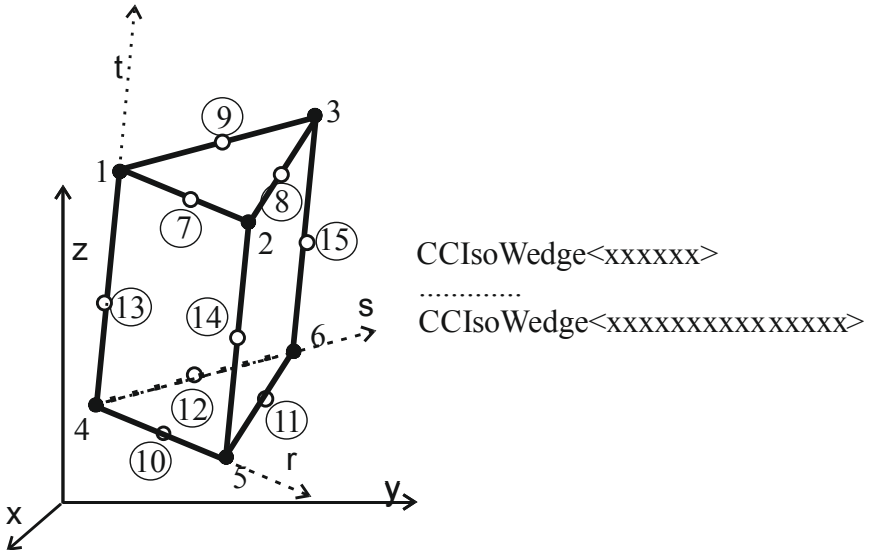


Fig. 3-9 Geometry of CCIsoWedge<...> elements.

Table 3.5-1 Interpolation functions of CCIsoTetra<...> elements.

Node i	Function h_i	Include only if node i is defined					
		$i = 5$	$i = 6$	$i = 7$	$i = 8$	$i = 9$	$i = 10$
1	$1 - r - s - t$	$-\frac{1}{2}h_5$		$-\frac{1}{2}h_7$			$-\frac{1}{2}h_{10}$
2	r	$-\frac{1}{2}h_5$	$-\frac{1}{2}h_6$		$-\frac{1}{2}h_8$		
3	s		$-\frac{1}{2}h_6$	$-\frac{1}{2}h_7$		$-\frac{1}{2}h_9$	
4	t				$-\frac{1}{2}h_8$	$-\frac{1}{2}h_9$	$-\frac{1}{2}h_{10}$
5	$4r(1 - r - s - t)$						
6	$4rs(1 - t)$						
7	$4s(1 - r - s - t)$						
8	$4rt(1 - s)$						
9	$4st(1 - r)$						
10	$4t(1 - r - s - t)$						

Table 3.5-2 Sample point for Gauss integration of 4 nodes CCIsoTetra<...> element.

Integration point	Coordinate r	Coordinate s	Coordinate t	Weight
1	1/4	1/4	1/4	1/6

Table 3.5-3 Sample points for Gauss integration of 5 to 10 nodes CCIsoTetra<...> elements.

Integration point	Coordinate r	Coordinate s	Coordinate t	Weight
1	0.13819660	0.13819660	0.13819660	1/24
2	0.13819660	0.13819660	0.58541020	1/24
3	0.58541020	0.13819660	0.13819660	1/24
4	0.13819660	0.58541020	0.13819660	1/24

Table 3-4 Interpolation functions of CCIsoBrick<...> elements.

Node i	Function h_i	Include only if node i is defined											
		$i=9$	$i=10$	$i=11$	$i=12$	$i=13$	$i=14$	$i=15$	$i=16$	$i=17$	$i=18$	$i=19$	$i=20$
1	$\frac{1}{8}(1+r)(1+s)(1+t)$	$-\frac{1}{2}h_9$			$-\frac{1}{2}h_{12}$					$-\frac{1}{2}h_{17}$			
2	$\frac{1}{8}(1-r)(1+s)(1+t)$	$-\frac{1}{2}h_9$	$-\frac{1}{2}h_{10}$								$-\frac{1}{2}h_{18}$		
3	$\frac{1}{8}(1-r)(1-s)(1+t)$		$-\frac{1}{2}h_{10}$	$-\frac{1}{2}h_{11}$								$-\frac{1}{2}h_{19}$	
4	$\frac{1}{8}(1+r)(1-s)(1+t)$			$-\frac{1}{2}h_{11}$	$-\frac{1}{2}h_{12}$								$-\frac{1}{2}h_{20}$
5	$\frac{1}{8}(1+r)(1+s)(1-t)$					$-\frac{1}{2}h_{13}$			$-\frac{1}{2}h_{16}$	$-\frac{1}{2}h_{17}$			
6	$\frac{1}{8}(1-r)(1+s)(1-t)$					$-\frac{1}{2}h_{13}$	$-\frac{1}{2}h_{14}$				$-\frac{1}{2}h_{18}$		
7	$\frac{1}{8}(1-r)(1-s)(1-t)$						$-\frac{1}{2}h_{14}$	$-\frac{1}{2}h_{15}$				$-\frac{1}{2}h_{19}$	

8	$\frac{1}{8}(1+r)(1-s)(1-t)$							$-\frac{1}{2}h_{15}$	$-\frac{1}{2}h_{16}$				$-\frac{1}{2}h_{20}$
9	$\frac{1}{4}(1-r^2)(1+s)(1+t)$												
10	$\frac{1}{4}(1-r)(1-s^2)(1+t)$												
11	$\frac{1}{4}(1-r^2)(1-s)(1+t)$												
12	$\frac{1}{4}(1+r)(1-s^2)(1+t)$												
13	$\frac{1}{4}(1-r^2)(1+s)(1-t)$												
14	$\frac{1}{4}(1-r)(1-s^2)(1-t)$												
15	$\frac{1}{4}(1-r^2)(1-s)(1-t)$												
16	$\frac{1}{4}(1+r)(1-s^2)(1-t)$												
17	$\frac{1}{4}(1+r)(1+s)(1-t^2)$												
18	$\frac{1}{8}(1-r)(1+s)(1-t^2)$												
19	$\frac{1}{4}(1-r)(1-s)(1-t^2)$												
20	$\frac{1}{4}(1+r)(1-s)(1-t^2)$												

Table 3.5-4 Sample points for Gauss integration of 8 nodes CCIsoBrick<...> element.

Inte-gration point	Coordinate *r*	Coordinate *s*	Coordinate *t*	Weight
1	0.577350269189626	0.577350269189626	0.577350269189626	1.
2	0.577350269189626	0.577350269189626	-0.577350269189626	1.
3	0.577350269189626	-0.577350269189626	0.577350269189626	1.

4	0.577350269189626	-0.577350269189626	-0.577350269189626	1.
5	-0.577350269189626	0.577350269189626	0.577350269189626	1.
6	-0.577350269189626	0.577350269189626	-0.577350269189626	1.
7	-0.577350269189626	-0.577350269189626	0.577350269189626	1.
8	-0.577350269189626	-0.577350269189626	-0.577350269189626	1.

Table 3.5-5 Sample points for Gauss integration of 9 to 20 nodes CCIsoBrick<...> element.

Inte-gration point	Coordinate *r*	Coordinate *s*	Coordinate *t*	Weight
1	0.774596669241483	0.774596669241483	0.774596669241483	0.1714677641
2	0.774596669241483	0.774596669241483	0.	0.2743484225
3	0.774596669241483	0.774596669241483	-0.774596669241483	0.1714677641
4	0.774596669241483	0.	0.774596669241483	0.2743484225
5	0.774596669241483	0.	0.	0.4389574760
6	0.774596669241483	0.	-0.774596669241483	0.2743484225
7	0.774596669241483	-0.774596669241483	0.774596669241483	0.1714677641

8	0.774596669241483	-0.774596669241483	0.	0.2743484225
10	0.	0.774596669241483	0.774596669241483	0.2743484225
11	0.	0.774596669241483	0.	0.4389574760
12	0.	0.774596669241483	-0.774596669241483	0.2743484225
13	0.	0.	0.774596669241483	0.4389574760
14	0.	0.	0.	0.7023319616
15	0.	0.	-0.774596669241483	0.4389574760
16	0.	-0.774596669241483	0.774596669241483	0.2743484225
17	0.	-0.774596669241483	0.	0.4389574760
18	0.	-0.774596669241483	-0.774596669241483	0.2743484225
19	-0.774596669241483	0.774596669241483	0.774596669241483	0.1714677641
20	-0.774596669241483	0.774596669241483	0.	0.2743484225
21	-0.774596669241483	0.774596669241483	-0.774596669241483	0.1714677641
22	-0.774596669241483	0.	0.774596669241483	0.2743484225
23	-0.774596669241483	0.	0.	0.4389574760

24	-0.774596669241483	0.	-0.774596669241483	0.2743484225
25	-0.774596669241483	-0.774596669241483	0.774596669241483	0.1714677641
26	-0.774596669241483	-0.774596669241483	0.	0.2743484225
27	-0.774596669241483	-0.774596669241483	-0.774596669241483	0.1714677641

Table 3.5-6 Interpolation functions of CCIsoWedge<...> elements.

$$hh_1 = (1 - r - s)$$
$$hh_2 = r$$
$$hh_3 = s$$
$$hh_4 = 4r(1 - r - s)$$
$$hh_5 = 4rs$$
$$hh_6 = 4s(1 - r - s)$$
$$hv_1 = \frac{1 + t}{2}$$
$$hv_2 = \frac{1 - t}{2}$$
$$hv_3 = (1 - t^2)$$

Node *I*	Function h_i	Include only if node *i* is defined								
		i = 7	*i* = 8	*i* = 9	*i* = 10	*i* = 11	*i* = 12	*i* = 13	*i* = 14	*i* = 15
1	$hh_1\, hv_1$	$-\frac{1}{2}h_7$		$-\frac{1}{2}h_9$				$-\frac{1}{2}h_{13}$		
2	$hh_2\, hv_1$	$-\frac{1}{2}h_7$	$-\frac{1}{2}h_8$						$-\frac{1}{2}h_{14}$	
3	$hh_3\, hv_1$		$-\frac{1}{2}h_8$	$-\frac{1}{2}h_9$						$-\frac{1}{2}h_{15}$
4	$hh_1\, hv_2$				$-\frac{1}{2}h_{10}$		$-\frac{1}{2}h_{12}$	$-\frac{1}{2}h_{13}$		
5	$hh_2\, hv_2$				$-\frac{1}{2}h_{10}$	$-\frac{1}{2}h_{11}$			$-\frac{1}{2}h_{14}$	
6	$hh_3\, hv_2$					$-\frac{1}{2}h_{11}$	$-\frac{1}{2}h_{12}$			$-\frac{1}{2}h_{15}$
7	$hh_4\, hv_1$									
8	$hh_5\, hv_1$									
9	$hh_6\, hv_1$									
10	$hh_4\, hv_2$									
11	$hh_5\, hv_2$									
12	$hh_6\, hv_2$									
13	$hh_1\, hv_3$									
14	$hh_2\, hv_3$									
15	$hh_3\, hv_3$									

Table 3.5-7 Sample points for Gauss integration of 6 nodes CCIsoWedge<...> element.

Integration point	Coordinate *r*	Coordinate *s*	Coordinate *t*	Weight
1	1/6	1/6	0.577350269189626	1/6
2	2/3	1/6	0.577350269189626	1/6
3	1/6	2/3	0.577350269189626	1/6

4	1/6	1/6	-0.577350269189626	1/6
5	2/3	1/6	-0.577350269189626	1/6
6	1/6	2/3	-0.577350269189626	1/6

Table 3.5-8 Sample points for Gauss integration of 7 to 15 nodes CCIsoWedge<...> element.

Integration point	Coordinate *r*	Coordinate *s*	Coordinate *t*	Weight
1	1/6	1/6	0.774596669241483	0.0925925926
2	2/3	1/6	0.774596669241483	0.0925925926
3	1/6	2/3	0.774596669241483	0.0925925926
4	1/6	1/6	0.	0.1481448148
5	2/3	1/6	0.	0.1481448148
6	1/6	2/3	0.	0.1481448148
7	1/6	1/6	-0.774596669241483	0.0925925926
8	2/3	1/6	-0.774596669241483	0.0925925926
9	1/6	2/3	-0.774596669241483	0.0925925926

Formulation of 3D solid elements is given in the following equations:

Incremental strains:

$$ {}_t\varepsilon_{ij}^{(i)} = \frac{1}{2}\left({}_t u_{i,j}^{(i)} + {}_t u_{j,i}^{(i)} \right) + \frac{1}{2}\left({}^{t+\Delta t}_{\ t}u_{k,i}^{(i-1)}\ {}_t u_{k,j}^{(i)} + {}^{t+\Delta t}_{\ t}u_{k,j}^{(i-1)}\ {}_t u_{k,i}^{(i)} \right) + \frac{1}{2}\left({}_t u_{k,i}^{(i)}\ {}_t u_{k,j}^{(i)} \right) \tag{3.25} $$

where indices $i, j, k \in <1...3>$

Displacement derivatives:

$$ {}_t u_{i,j}^{(i)} = \frac{\partial\left({}^{t+\Delta t}u_i^{(i)} - {}^{t+\Delta t}u_i^{(i-1)} \right)}{\partial {}^t x_j} $$

$$ {}^{t+\Delta t}_{\ t}u_{i,j}^{(i-1)} = \frac{\partial {}^{t+\Delta t}u_i^{(i-1)}}{\partial {}^t x_j} \tag{3.26} $$

Strains and matrices to calculate them:

$$
{}_t\underline{\varepsilon}^{(i)} = {}^{t+\Delta t}_{t}\boldsymbol{B}_L^{(i-1)}\,\Delta\underline{U}^{(i)}
$$

$$
{}_t\underline{\varepsilon}^{(i)} = \left[\, {}_t\varepsilon_{11}^{(i)} \quad {}_t\varepsilon_{22}^{(i)} \quad {}_t\varepsilon_{33}^{(i)} \quad 2\,{}_t\varepsilon_{12}^{(i)} \quad 2\,{}_t\varepsilon_{23}^{(i)} \quad 2\,{}_t\varepsilon_{13}^{(i)} \,\right] \tag{3.27}
$$

$$
\Delta\underline{U}^{(i)} = {}^{t+\Delta t}\underline{U}^{(i)} - {}^{t+\Delta t}\underline{U}^{(i-1)} =
$$
$$
\left[\, u_1^{1(i)} \quad u_2^{1(i)} \quad u_3^{1(i)} \quad u_1^{2(i)} \quad u_2^{2(i)} \quad u_3^{2(i)} \quad \ldots \quad u_1^{n(i)} \quad u_2^{n(i)} \quad u_3^{n(i)} \,\right]
$$

Linear strain-displacement matrix:

$$
{}^{t+\Delta t}_{t}\boldsymbol{B}_L^{(i-1)} = {}^{t+\Delta t}_{t}\boldsymbol{B}_{L0} + {}^{t+\Delta t}_{t}\boldsymbol{B}_{L1}^{(i-1)} \tag{3.28}
$$

Linear strain-displacement matrix – constant part:

$$
{}^{t+\Delta t}_{t}\boldsymbol{B}_{L0} = \begin{bmatrix}
{}_th_{1,1} & 0 & 0 & {}_th_{2,1} & 0 & 0 & \ldots & {}_th_{n,1} & 0 & 0 \\
0 & {}_th_{1,2} & 0 & 0 & {}_th_{2,2} & 0 & \ldots & 0 & {}_th_{n,2} & 0 \\
0 & 0 & {}_th_{1,3} & 0 & 0 & {}_th_{2,3} & \ldots & 0 & 0 & {}_th_{n,3} \\
{}_th_{1,2} & {}_th_{1,1} & 0 & {}_th_{2,2} & {}_th_{2,1} & 0 & \ldots & {}_th_{n,2} & {}_th_{n,1} & 0 \\
0 & {}_th_{1,3} & {}_th_{1,2} & 0 & {}_th_{2,3} & {}_th_{2,2} & \ldots & 0 & {}_th_{n,3} & {}_th_{n,2} \\
{}_th_{1,3} & 0 & {}_th_{1,1} & {}_th_{2,3} & 0 & {}_th_{2,1} & \ldots & {}_th_{n,3} & 0 & {}_th_{n,1}
\end{bmatrix} \tag{3.29}
$$

where

$$
{}_th_{i,j} = \frac{\partial h_i}{\partial {}^tx_j} \tag{3.30}
$$

$$
u_i^{(i)} = {}^{t+\Delta t}u_i^{(i)} - {}^{t+\Delta t}u_i^{(i-1)}
$$

Linear strain-displacement matrix – non-constant part:

$$
{}^{t+\Delta t}_{t}\boldsymbol{B}_{L1}^{(i-1)} = \begin{bmatrix}
{}^{t+\Delta t}l_{11}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{31}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_th_{2,1} \\
{}^{t+\Delta t}l_{12}^{(i-1)}\,{}_th_{1,2} & {}^{t+\Delta t}l_{22}^{(i-1)}\,{}_th_{1,2} & {}^{t+\Delta t}l_{32}^{(i-1)}\,{}_th_{1,2} & {}^{t+\Delta t}l_{12}^{(i-1)}\,{}_th_{2,2} \\
{}^{t+\Delta t}l_{13}^{(i-1)}\,{}_th_{1,3} & {}^{t+\Delta t}l_{23}^{(i-1)}\,{}_th_{1,3} & {}^{t+\Delta t}l_{33}^{(i-1)}\,{}_th_{1,3} & {}^{t+\Delta t}l_{13}^{(i-1)}\,{}_th_{2,3} \\
{}^{t+\Delta t}l_{11}^{(i-1)}\,{}_th_{1,2} + {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_th_{1,2} + {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{31}^{(i-1)}\,{}_th_{1,2} + {}^{t+\Delta t}l_{31}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_th_{2,2} + {}^{t+\Delta t}l_{12}^{(i-1)}\,{}_th_{2,1} \\
{}^{t+\Delta t}l_{12}^{(i-1)}\,{}_th_{1,3} + {}^{t+\Delta t}l_{13}^{(i-1)}\,{}_th_{1,2} & {}^{t+\Delta t}l_{22}^{(i-1)}\,{}_th_{1,3} + {}^{t+\Delta t}l_{23}^{(i-1)}\,{}_th_{1,2} & {}^{t+\Delta t}l_{32}^{(i-1)}\,{}_th_{1,3} + {}^{t+\Delta t}l_{33}^{(i-1)}\,{}_th_{1,2} & {}^{t+\Delta t}l_{12}^{(i-1)}\,{}_th_{2,3} + {}^{t+\Delta t}l_{13}^{(i-1)}\,{}_th_{2,2} \\
{}^{t+\Delta t}l_{11}^{(i-1)}\,{}_th_{1,3} + {}^{t+\Delta t}l_{13}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{21}^{(i-1)}\,{}_th_{1,3} + {}^{t+\Delta t}l_{23}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{31}^{(i-1)}\,{}_th_{1,3} + {}^{t+\Delta t}l_{33}^{(i-1)}\,{}_th_{1,1} & {}^{t+\Delta t}l_{11}^{(i-1)}\,{}_th_{2,3} + {}^{t+\Delta t}l_{13}^{(i-1)}\,{}_th_{2,1}
\end{bmatrix}
$$

$$
\begin{bmatrix}
\ldots & {}^{t+\Delta t}l_{31}^{(i-1)}\,{}_th_{n,1} \\
\ldots & {}^{t+\Delta t}l_{31}^{(i-1)}\,{}_th_{n,2} \\
\ldots & {}^{t+\Delta t}l_{33}^{(i-1)}\,{}_th_{n,3} \\
\ldots & {}^{t+\Delta t}l_{31}^{(i-1)}\,{}_th_{n,2} + {}^{t+\Delta t}l_{32}^{(i-1)}\,{}_th_{n,1} \\
\ldots & {}^{t+\Delta t}l_{32}^{(i-1)}\,{}_th_{n,3} + {}^{t+\Delta t}l_{33}^{(i-1)}\,{}_th_{n,2} \\
\ldots & {}^{t+\Delta t}l_{31}^{(i-1)}\,{}_th_{n,3} + {}^{t+\Delta t}l_{33}^{(i-1)}\,{}_th_{n,1}
\end{bmatrix} \tag{3.31}
$$

where

$$
{}^{t+\Delta t}l_{ij}^{(i-1)} = \sum_{k=1}^{n} {}_th_{k,j}\;{}^{t+\Delta t}_{t}u_i^{k(i-1)} \tag{3.32}
$$

Nonlinear strain-displacement matrix

$$
{}^{t+\Delta t}_{t}\boldsymbol{B}_{NL}^{(i-1)} = \begin{bmatrix}
{}_{t}h_{1,1} & 0 & 0 & {}_{t}h_{2,1} & \dots & {}_{t}h_{n,1} & 0 & 0 \\
{}_{t}h_{1,2} & 0 & 0 & {}_{t}h_{2,2} & \dots & {}_{t}h_{n,2} & 0 & 0 \\
{}_{t}h_{1,3} & 0 & 0 & {}_{t}h_{2,3} & \dots & {}_{t}h_{n,3} & 0 & 0 \\
0 & {}_{t}h_{1,1} & 0 & 0 & \dots & 0 & {}_{t}h_{n,1} & 0 \\
0 & {}_{t}h_{1,2} & 0 & 0 & \dots & 0 & {}_{t}h_{n,2} & 0 \\
0 & {}_{t}h_{1,3} & 0 & 0 & \dots & 0 & {}_{t}h_{n,3} & 0 \\
0 & 0 & {}_{t}h_{1,1} & 0 & \dots & 0 & 0 & {}_{t}h_{n,1} \\
0 & 0 & {}_{t}h_{1,2} & 0 & \dots & 0 & 0 & {}_{t}h_{n,2} \\
0 & 0 & {}_{t}h_{1,3} & 0 & \dots & 0 & 0 & {}_{t}h_{n,3}
\end{bmatrix} \tag{3.33}
$$

2nd Piola-Kirchhoff stress tensor and vector

$$
{}_{t}\boldsymbol{S}^{(i-1)} = \begin{bmatrix}
{}^{t+\Delta t}_{t}S_{11}^{(i-1)} & {}^{t+\Delta t}_{t}S_{12}^{(i-1)} & {}^{t+\Delta t}_{t}S_{13}^{(i-1)} & 0 & 0 & 0 & 0 & 0 & 0 \\
{}^{t+\Delta t}_{t}S_{21}^{(i-1)} & {}^{t+\Delta t}_{t}S_{22}^{(i-1)} & {}^{t+\Delta t}_{t}S_{23}^{(i-1)} & 0 & 0 & 0 & 0 & 0 & 0 \\
{}^{t+\Delta t}_{t}S_{31}^{(i-1)} & {}^{t+\Delta t}_{t}S_{32}^{(i-1)} & {}^{t+\Delta t}_{t}S_{33}^{(i-1)} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & {}^{t+\Delta t}_{t}S_{11}^{(i-1)} & {}^{t+\Delta t}_{t}S_{12}^{(i-1)} & {}^{t+\Delta t}_{t}S_{13}^{(i-1)} & 0 & 0 & 0 \\
0 & 0 & 0 & {}^{t+\Delta t}_{t}S_{21}^{(i-1)} & {}^{t+\Delta t}_{t}S_{22}^{(i-1)} & {}^{t+\Delta t}_{t}S_{23}^{(i-1)} & 0 & 0 & 0 \\
0 & 0 & 0 & {}^{t+\Delta t}_{t}S_{31}^{(i-1)} & {}^{t+\Delta t}_{t}S_{32}^{(i-1)} & {}^{t+\Delta t}_{t}S_{33}^{(i-1)} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & {}^{t+\Delta t}_{t}S_{11}^{(i-1)} & {}^{t+\Delta t}_{t}S_{12}^{(i-1)} & {}^{t+\Delta t}_{t}S_{13}^{(i-1)} \\
0 & 0 & 0 & 0 & 0 & 0 & {}^{t+\Delta t}_{t}S_{21}^{(i-1)} & {}^{t+\Delta t}_{t}S_{22}^{(i-1)} & {}^{t+\Delta t}_{t}S_{23}^{(i-1)} \\
0 & 0 & 0 & 0 & 0 & 0 & {}^{t+\Delta t}_{t}S_{31}^{(i-1)} & {}^{t+\Delta t}_{t}S_{32}^{(i-1)} & {}^{t+\Delta t}_{t}S_{33}^{(i-1)}
\end{bmatrix}
$$

$$
\underline{S}^{(i)} = \begin{bmatrix} {}^{t+\Delta t}_{t}S_{11}^{(i-1)} & {}^{t+\Delta t}_{t}S_{22}^{(i-1)} & {}^{t+\Delta t}_{t}S_{33}^{(i-1)} & {}^{t+\Delta t}_{t}S_{12}^{(i-1)} & {}^{t+\Delta t}_{t}S_{23}^{(i-1)} & {}^{t+\Delta t}_{t}S_{13}^{(i-1)} \end{bmatrix} \tag{3.34}
$$

3.6 Spring Element

Spring elements in ATENA are used to model spring-like boundary conditions, i.e. situation where external forces acting on boundary of the structure are linearly proportional to the associated displacements. Three elements of this type are available, see also Fig. 3-10, Fig. 3-11:

CCSpring – 2D and 3D element to model spring-like boundary conditions at a point,

CCLineSpring – 2D element to model spring-like boundary conditions along a line

CCPlaneSpring – 3D element to model spring-like boundary conditions along a triangular area.

All these elements are derived from 2D or 3D formulation of the CCIsoTruss<xx> element described earlier in this chapter. For example, CCSpring element consists of one CCIsoTruss<xx> element. The 1st node of each CCIsoTruss<xx> coincides with one node of the CCSpring element, whereas the 2nd node of the CCIsoTruss<xx> is set by *direction* vector, see Fig. 4-4. Note that as the analysis is nonlinear, length of the *direction* does matter. This vector is specified in ATENA &SPRING_GEOMETRY_SPEC command and is common for all spring elements that use this geometry.

CCLineSpring and CCPlaneSpring elements were created to enable convenient definition of „uniform“ spring-like conditions along the boundaries. The boundary force at a node i of the spring element is calculated:

$$R_i = \frac{u_i k A}{n \left\| direction \right\|} \tag{3.35}$$

where

k is spring material stiffness parameter set by &MATERIAL SPRING command, (parameter k has character of multi-linear Young modulus),

u_i is displacement at spring element node i ,

A is the area of CCPlaneSpring element or length of CCLineSpring multiplied by thickness (which defaults to 1 if not specified in element geometry) or the area defined in element geometry for CCSpring (similarly, with a default of 1 if not specified) for the respective element,

n is number element nodes, i.e. 1, 2 or 3 for CCSpring, CCLineSpring or CCPlaneSpring element respectively,

$\left\| direction \right\|$ is Euclidean norm (i.e., length) of the *direction* vector, see above.

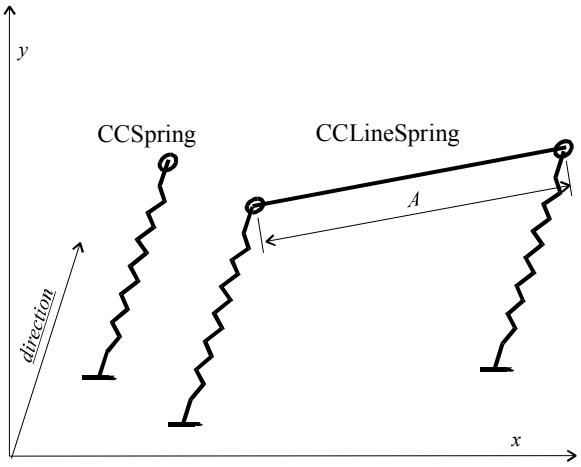


Fig. 3-10 Geometry of 2D CCSpring and CCIsoLineSpring.

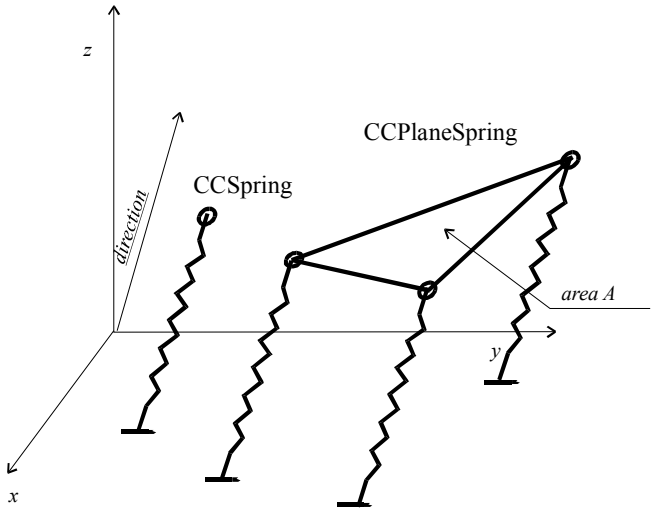


Fig. 3-11 Geometry of 3D CCSpring and CCPlaneSpring.

3.7 Quadrilateral Element Q10

3.7.1 Element Stiffness Matrix

The quadrilateral finite element Q-10 is derived from a six-node triangle (CCQ10<xxxx>, CCQ10Sbeta<xxxx>). The derivation of the stiffness matrix is taken from FELIPPA 1966. The position of any internal point *P* in the element is defined by the triangular coordinates □□ (called also natural coordinates). These coordinates are expressed by means of areas within the triangle as shown in Fig. 3-12. Sub-areas A_i are subtended by the point *P* and two corners. *A* is the area of triangular element.

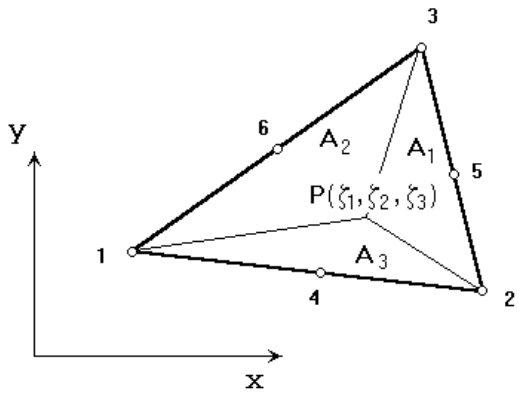


Fig. 3-12 Coordinate systems of the six-node triangular element.

$$\zeta_1 = \frac{A_1}{A}, \zeta_2 = \frac{A_2}{A}, \zeta_3 = \frac{A_3}{A} \tag{3.36}$$

$$\zeta_1 + \zeta_2 + \zeta_3 = 1$$

Using the quadratic interpolation function, the displacement components $u(\zeta_i)$, $v(\zeta_i)$ is written in the terms of triangular coordinates ζ_i and nodal displacement vectors :

$$u(\zeta_i) = \mathbf{F}(\zeta_i)^T \mathbf{u}, v(\zeta_i) = \mathbf{F}(\zeta_i)^T \mathbf{v} \tag{3.37}$$

The displacement vectors **u, v** contain six components of the nodal displacements and the vector $\mathbf{F}(\zeta_i)$ contains the quadratic interpolation functions in triangular coordinates:

$$\mathbf{u} = \left\{ \begin{array}{cccccc} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \end{array} \right\}^T, \mathbf{v} = \left\{ \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \right\}^T \tag{3.38}$$

$$\mathbf{F}(\zeta_i) = \left\{ \zeta_1(2\zeta_1 - 1) \quad \zeta_2(2\zeta_2 - 1) \quad \zeta_3(2\zeta_3 - 1) \quad 4\zeta_1\zeta_2 \quad 4\zeta_2\zeta_3 \quad 4\zeta_3\zeta_1 \right\}^T \tag{3.39}$$

A general procedure to construct the element stiffness matrix is described by the set of following equations:

(a) The constitutive equation:

$$\mathbf{s} = \mathbf{D} \quad \mathbf{e} \tag{3.40}$$

(b) The strain-displacement equations in the Cartesian coordinates:

$$\varepsilon_x = \frac{\partial u(x,y)}{\partial x}, \quad \varepsilon_y = \frac{\partial v(x,y)}{\partial y}, \quad \gamma = \frac{\partial u(x,y)}{\partial y} + \frac{\partial v(x,y)}{\partial x} \tag{3.41}$$

which is written in terms of the natural coordinates ζ_i and the nodal displacements vectors **u, v**:

$$\varepsilon \ (\zeta_i) = \mathbf{F}_\partial{}^T \left\{ \begin{array}{c} \mathbf{u} \\ \mathbf{v} \end{array} \right\} \tag{3.42}$$

The stiffness matrix:

$$\mathbf{K} = \int_V \mathbf{F}_\partial{}^T \mathbf{D} \quad \mathbf{F}_\partial dV \tag{3.43}$$

The matrix $\mathbf{F}_\partial$ contains partial derivatives of the interpolation function **F** and the integral in the last equation is made over the element volume *V*. The details of the derivation can be found in FELIPPA 1966 and here only the final matrix equations are presented.

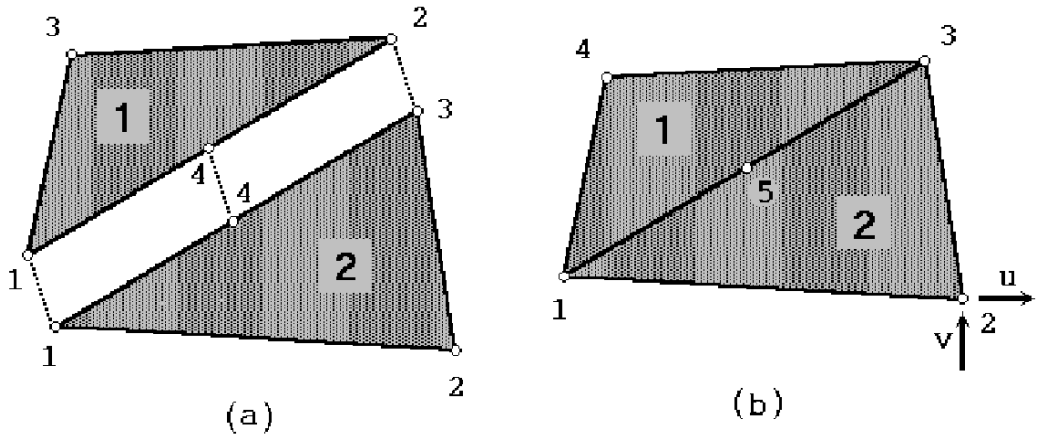


Fig. 3-13 Quadrilateral element (b) composed from two triangular elements (a).

The quadrilateral finite element is composed from two 4-node triangular elements, as shown in Fig. 3-13. Two degrees of freedom in a node are the horizontal and vertical displacements. The triangular element is derived from the 6-node triangle by imposing kinematic constraints on two mid-side nodes. The resulting strain-displacement matrix relation for the 4-node triangle is:

$$\mathbf{e} = \mathbf{B}\mathbf{d} \qquad \begin{Bmatrix} \mathbf{e}_x \\ \mathbf{e}_y \\ \mathbf{g} \end{Bmatrix} = \begin{bmatrix} \mathbf{U} & \mathbf{O} \\ \mathbf{O} & \mathbf{V} \\ \mathbf{V} & \mathbf{U} \end{bmatrix} \begin{Bmatrix} \mathbf{u} \\ \mathbf{v} \end{Bmatrix} \tag{3.44}$$

where $\mathbf{e}_x$, $\mathbf{e}_y$ are the normal strain vectors, $\mathbf{g}$ is the shear strain vector (engineering type) and $\mathbf{O}$ is the null matrix. The strain and displacement vectors contain nodal components:

$$\mathbf{e}_x = \begin{Bmatrix} \varepsilon_{x1} & \varepsilon_{x2} & \varepsilon_{x3} \end{Bmatrix}^T, \mathbf{e}_y = \begin{Bmatrix} \varepsilon_{y1} & \varepsilon_{y2} & \varepsilon_{y3} \end{Bmatrix}^T, g = \begin{Bmatrix} \gamma_{x1} & \gamma_{x2} & \gamma_{x3} \end{Bmatrix}^T \tag{3.45}$$

$$\mathbf{u} = \begin{Bmatrix} u_1 & u_2 & u_3 & u_4 \end{Bmatrix}^T, \mathbf{v} = \begin{Bmatrix} v_1 & v_2 & v_3 & v_4 \end{Bmatrix}^T \tag{3.46}$$

The strain interpolation function in the element is linear and is uniquely specified by three nodal values in the corners of the triangular element, while the displacement interpolation function is quadratic and is specified by three corners and one mid-side nodal displacement. The components u_i, v_i are the horizontal and vertical displacements, respectively, in the node i. The indexes 1, 2 and 3 denote the corner nodes of a sub-triangle and the index 4 is for the mid-side node, see Fig. 3-13 (a). The strain-displacement sub-matrices in (3.44) are

$$\mathbf{U} = \frac{1}{2S}\begin{bmatrix} 3b_1 + 2b_3 & -b_2 & b_3 & 4b_2 \\ -b_1 & 3b_2 + b_3 & b_3 & 4b_1 \\ b_1 & b_2 & b_3 & . \end{bmatrix}$$

$$\mathbf{V} = \frac{1}{2S}\begin{bmatrix} 3a_1 + 2a_3 & -a_2 & a_3 & 4a_2 \\ -a_1 & 3a_2 + a_3 & a_3 & 4a_1 \\ a_1 & a_2 & a_3 & . \end{bmatrix} \tag{3.47}$$

$$a_1 = x_3 - x_2 \; b_1 = y_2 - y_3$$
$$a_2 = x_1 - x_3 \; b_2 = y_3 - y_1$$
$$a_3 = x_2 - x_1 \; b_3 = y_1 - y_2$$
$$2S = a_3 b_2 - a_2 b_3$$

where x_i, y_i are the global Cartesian coordinates of the node i in a sub-triangle, S is the area of the sub-triangle.

The element stiffness matrix for the 4-node sub-triangle is

$$\mathbf{K} = \begin{bmatrix} K_{uu} & K_{uv} \\ K_{vu} & K_{vv} \end{bmatrix} \tag{3.48}$$

The stiffness matrix $\mathbf{K}$ has an order 8 and is so partitioned that the upper four rows correspond to the horizontal displacement components (index u) and the lower four rows correspond to the vertical displacement components (index v). The integration of the stiffness coefficients is made exactly, and the resulting sub-matrices are:

$$\mathbf{K}_{uu} = St\left[d_{11}\mathbf{A} + d_{13}(\mathbf{H} + \mathbf{H}^T) + d_{33}\mathbf{C} \right]$$

$$\mathbf{K}_{vv} = St\left[d_{22}\mathbf{C} + d_{23}(\mathbf{H} + \mathbf{H}^T) + d_{33}\mathbf{A} \right]$$

$$\mathbf{K}_{uv} = St\left[d_{12}\mathbf{H} + d_{13}\mathbf{A} + d_{23}\mathbf{C} + d_{33}\mathbf{H}^T \right] \tag{3.49}$$

where t is the thickness of the element, d_{ij} are the coefficients of the material stiffness matrix **D**, (3.40). The integration in (3.43) is done explicitly by the following matrix multiplication:

$$\mathbf{A} = \mathbf{U}^T\mathbf{Q}\mathbf{U}, \mathbf{H} = \mathbf{U}^T\mathbf{Q}\mathbf{V}, \mathbf{C} = \mathbf{V}^T\mathbf{Q}\mathbf{V} \tag{3.50}$$

Where the area integration matrix **Q** is:

$$\mathbf{Q} = \frac{1}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \tag{3.51}$$

The element stiffness matrix of the 5-node quadrilateral, Fig. 3-13(b), is composed of the two 4-node sub-triangles by summing the stiffness coefficients of the appropriate nodes. The resulting matrix of the 5-node quadrilateral $\mathbf{K}_{10}$ has the order 10. The coefficients of the matrix can be rearranged according to the external (index e) and internal (index i) degrees of freedom:

$$\mathbf{K}_{10} = \begin{bmatrix} \mathbf{K}_{ee} & \mathbf{K}_{ei} \\ \mathbf{K}_{ie} & \mathbf{K}_{ii} \end{bmatrix} \tag{3.52}$$

The sub-matrices corresponding to two internal degrees of freedom are eliminated by condensation procedure and the final element stiffness matrix **K** of the order 8 is obtained:

$$\mathbf{K} = \mathbf{K}_{ee} - \mathbf{K}_{ei}\mathbf{K}_{ii}^{-1}\mathbf{K}_{ie} \tag{3.53}$$

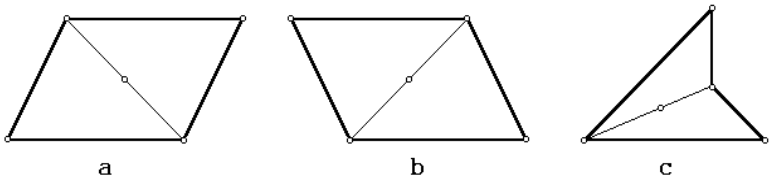


Fig. 3-14 Subdivision of quadrilateral element.

The subdivision of the quadrilateral element into the triangular elements must be done in an optimal way and it is preformed automatically by the program. The examples of the subdivisions are illustrated by Fig. 3-14. Due to this method of the subdivision, a concave form of the quadrilateral element is acceptable. This element form could not be achieved by an isoparametric element.

3.7.2 Evaluation of Stresses and Resisting Forces

For the given displacement field, the strains and stresses are evaluated in the center of the quadrilateral element. The stresses at this point are obtained from material laws as functions of strains according to Section 2.1.12. Also, the constitutive law for the element and the matrix **D** are calculated from the stresses and strains at the center of the element. These stresses and strains are written in the output file as a part of the results.

The calculation of resisting nodal forces of the sub-triangle for a current displacement field and a constitutive law is done by the following equation:

$$\mathbf{R} = t\mathbf{B}^T\mathbf{Q}_9\mathbf{s}_9 \tag{3.54}$$

where **R** is the vector of nodal forces (same arrangement and numbering as in the vector **d** in (3.44)). The matrix $\mathbf{Q}_9$ contains three integration matrices **Q** in the diagonal. The stress vector $\mathbf{s}_9$

(same numbering as the vector **e**, (3.40), is calculated from the current strains and secant material matrix, Section 2.1.12.

There are two variations of this element in program ATENA: CCQ10<xxxx> and CCQ10Sbeta<xxxx>. The main difference between these two elements lies in the way how the resisting forces are calculated. In case CCQ10<xxxx>, they are computed as described by Equation (3.54). In the second case, however, the material law is evaluated only at the element centroid. Based on the current state of damage a secant constitutive matrix is calculated and it is used to determine the integration point stresses and resulting resisting forces. This element type is almost identical to the element that was implemented in the program SBETA, i.e. the former version of this program. Due to this approach, there are some limitations for usage of this element with respect to some material models. It can be only used with material models that are able to calculate and exact secant constitutive matrix. This means that only the following material models can be used with the element CCQ10Sbeta<xxxx>: CCElastIsotropic and CCSbetaMaterial.

3.8 External Cable

External pre-stressing cables are reinforcing bars, which are not connected with the most of the concrete body, except of limited number of points, so called deviators, as shown in Fig. 3-15. This element type is denoted in ATENA as CCExternalCable.

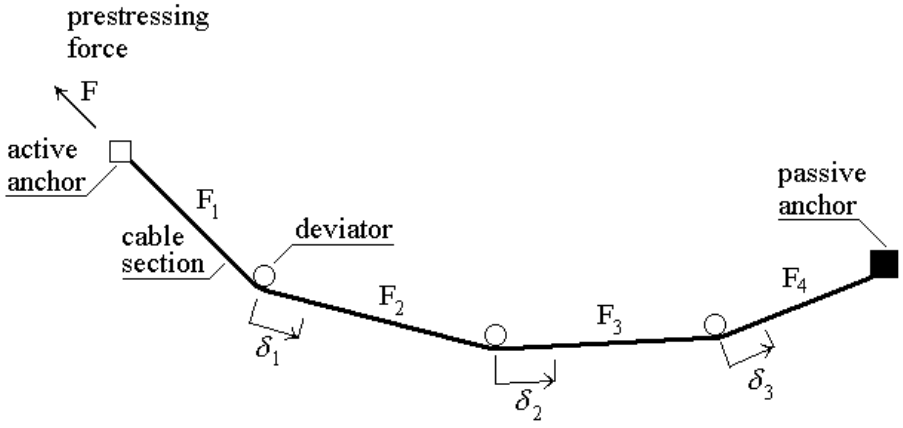


Fig. 3-15 External cable model.

Each cable has two ends provided with anchors. The anchor, where the pre-stressing force is applied is denoted as the *active anchor*, the anchor on the other side is the *passive anchor*. The points between the anchors are called *deviators* (or links). After applying pre-stressing the cable is fixed at anchors. In the deviators, cable can slide while its movements and the forces are governed by the law of dry friction. The slips of the cable in the deviators (the relative displacement of the cable ends with respect to the deviators) are denoted as δ_1, δ_2 ... They are introduced as variables to be determined by the analysis.

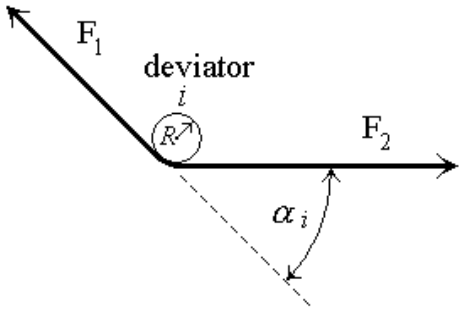


Fig. 3-16 Forces at the deviator.

The forces, F_1 and F_2 acting on a deviator i are the cable forces at the adjacent cable sections, Fig. 3-16. Their difference $P_i = F_1 - F_2$, ($F_1 > F_2$) is the loss of the pre-stressing force due to friction in the deviator i. The relation between these forces according to the law of friction is expressed as:

$$F_2 = \left(F_1 e^{-\varphi \alpha_i p} - Q\right) f_\delta(\delta)\, f_r(r) \tag{3.55}$$

In this expression α_i is the angular change of the cable direction at the deviator *i*, *R* is the radius of the deviator, (i.e. the product $R\alpha_i$ is the length the contact between the cable and the deviator.). φ is the friction coefficient. The constant part of the friction is $Q_i = p\,c_f\,R\,\alpha_i$, where c_f is the cohesion (a constant part of the friction) of the cable per unit length and unit perimeter. *p* stands for reinforcement bar perimeter. If the constant part of friction is neglected, the term *Q* is zero. $f_\delta(\delta)$, $f_r(r)$ are user defined function that enable change of deviator's properties depending on value of slip δ and deviator position coordinate *r* (measured from its starting point). By default, these functions are set to one.

Introducing $d_i^a = e^{-\varphi\alpha_i\,p}\,f_\delta(\delta)\,f_r(r)$ and $d_i^b = p\,c_f R\,\alpha_i\,f_\delta(\delta)\,f_r(r)$ we can simplify (3.55) to

$$F_2 = F_1\,d_1^a - d_1^b \tag{3.56}$$

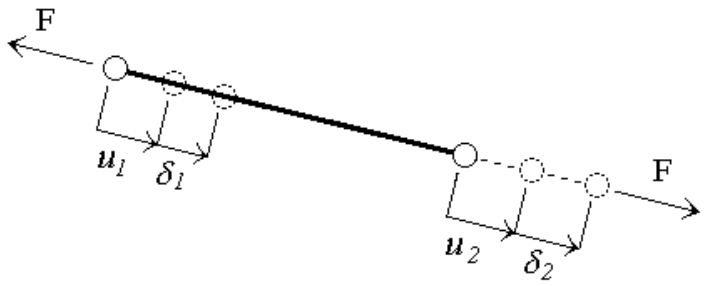


Fig. 3-17 Forces and displacements in the cable element (cable section).

A section of the cable between the deviators is considered as the uniaxial bar element, Fig. 3-17. The force *F* in the cable element depends on the pre-stressing force *P*, the displacements of ends u_1, u_2 due to structural deformation and the cable slips δ_1, δ_2 in the deviators. The slips δ are introduced as an additional variable for the external cables. The equilibrium equation of the cable section is:

$$F = P + K(u_2 - u_1 + \delta_2 - \delta_1) \tag{3.57}$$

The element stiffness $K = E_s\,A/L$, where *A, L* are the cable's cross section and length, respectively, and E_s is the actual secant or tangent modulus derived in the same way as in case of other reinforcement using bilinear or multi-linear law.

The cable forces F_1, F_2, … are determined by applying the above equations for all cable deviators, i.e. an iterative solution is executed for displacements u, (outer iterations loop), and on slips δ_i, (inner iteration loop).

Introduction of pre-stressing is accomplished by applying an initial slip (cable pull-out) at the anchor end until a prescribed pre-stressing force is reached. This procedure reflects a real process of pre-stressing and considers the loss of pre-stressing due to friction deviators and deformation of the structure.

3.9 Reinforcement Bars with Prescribed Bond

Reinforcement bars with prescribed bonds are an extension of the external cables described in the previous section. The main difference is that they can also account for a bond between the

bar and the surrounding concrete body. This connection need not be perfect, because the cohesion strength has a limited value. It is inputted in form of a "bond" cohesion stress.

This type of element is denoted as CCBarWithBond in ATENA. A typical reinforcement bar of this type is depicted in the figure below. The detail shows undeformed and deformed shape of a segment of the bar. The original length l_0 will change to l due to displacement u of the surrounding body and bar slips δ.

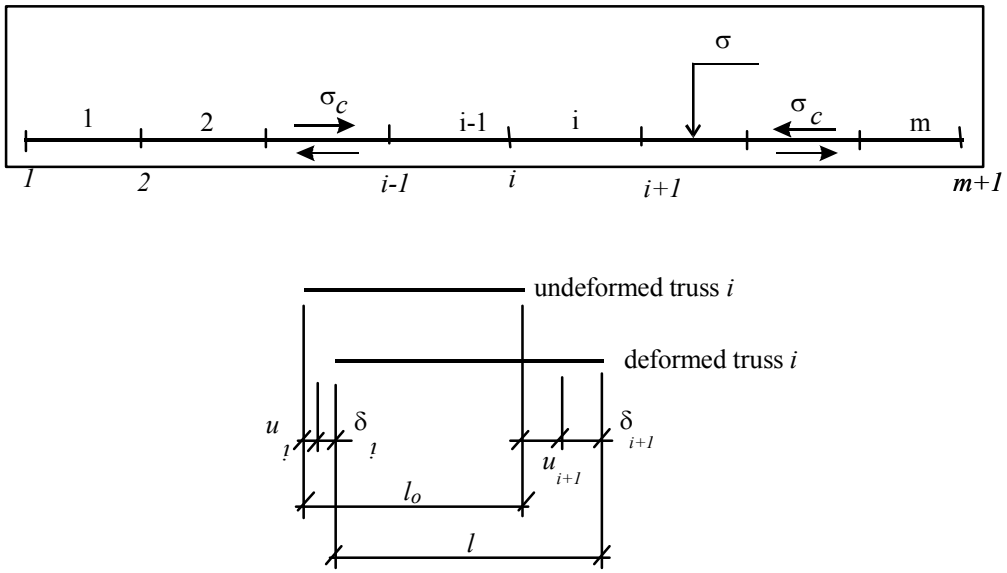


Fig. 3-18 Reinforcement bar with slips.

Normal stress at element i is calculated by:

$$\sigma_i = \frac{(u_{i+i} + \delta_{i+1} - u_i - \delta_i)}{l_i} E \tag{3.58}$$

Its derivative is compared with the total cohesion stress σ_c, i.e. $\frac{\partial \sigma_x}{\partial x} \le \sigma_c$. If the cohesion stress between the bar and the surrounding concrete is to be exceeded, the bar will slip to reduce this stress. Otherwise, the slips δ will remain unchanged (or initially equal to zero), which correspond to the case of perfect bond.

The total cohesion stress consists of two parts: base cohesion stress and so-called wobble cohesion stress, i.e. an extra cohesion dependent on axial stress in the bar, (see the term $\sigma_x f_w$ below). The wobble cohesion is derived as follows: Prestress losses are calculated by:

$$\begin{aligned}
&\Delta\sigma_r = \sigma_p (1 - e^{-\mu\kappa r}) \\
&\sigma_r = \sigma_p - \Delta\sigma = \sigma_p e^{-\mu\kappa r} \\
&\frac{\partial \sigma_r}{\partial r} = -\mu\kappa\, \sigma_p e^{-\mu\kappa r} = -\mu\kappa\, \sigma_r \\
&\frac{\partial \sigma_r}{\partial r} = -\mu\kappa\, \sigma_p e^{-\mu\kappa r} = f_w\, \sigma_r
\end{aligned} \tag{3.59}$$

The wobble related cohesion stress is thus $-\mu\kappa\, \sigma_r = f_w \sigma_r$.

Realizing that the cohesion stress can be constant, or it can be defined as a function of δ and r, we can calculate the total cohesion stress σ_c as follows:

$$\sigma_c = f_r(r) f_T(T) f_{c_{corr}}(c_{corr}) \left(\sigma_{c0} f_\delta(\delta) + \sigma_x f_w \right) \tag{3.60}$$

$f_\delta(\delta)$, $f_r(r)$ are the same as those described for external cables near (3.55), σ_{c0} is reference base cohesion stress due to slipping (to be inputted), σ_c is total cohesion stress due to slipping and wobble cohesion, p is perimeter of the reinforcement bar, r is location at the bar. σ_x is normal stress in the bar and f_w states for wobble coefficient. The remaining parameters are: σ_r is axial normal stress in the bar in direction of local coordinate axis *r* (in direction of the bar) and p, A means perimeter and cross-sectional area of the bar, (again similar to *r* in the case of external cable). Function $f_{c_{corr}}(c_{corr})$ and $f_T(T)$ expresses, how the cable's cohesion depends on current temperature and corrosion ratio A_{curr} / A_{orig} at a point of the cable. A_{curr} , A_{orig} is current and original (i.e. before corrosion started) area of cross section of the cable.

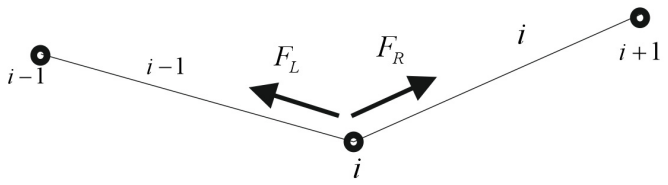


Fig. 3-19 Forces at node *i*.

The discretized solution equation for node *i*, (considering elements $i-1$, i), reads (the bars are of constant strain type):

BAR WITH PRESCRIBED BOND:

$$
\begin{aligned}
&F_i^L = A_{i-1}\ \sigma_{i-1} \\
&F_i^R = A_i\ \sigma_i \\
&\text{for } (F_i^R > F_i^L): \qquad F_i^R - F_i^L \le \frac{l_i + l_{i-1}}{2} p \left(\sigma_c + \frac{\partial \sigma_c}{\partial \delta_i} \Delta \delta_i \right) \\
&\text{for } (F_i^R < F_i^L): \qquad F_i^R - F_i^L \ge -\left(\frac{l_i + l_{i-1}}{2} p \left(\sigma_c + \frac{\partial \sigma_c}{\partial \delta_i} \Delta \delta_i \right) \right)
\end{aligned}
\tag{3.61}
$$

If this element acts as the external cable, see the previous section, then

EXTERNAL CABLE WITH DEVIATORS:

$$
\begin{aligned}
&\text{for } (F_i^R > F_i^L): \qquad F_i^L = F_i^R d_i^a - d_i^b \\
&\qquad\qquad\qquad\qquad\ \ F_i^R - F_i^L \le F_i^R - F_i^R d_i^a + d_i^b \\
&\qquad\qquad\qquad\qquad\ \ F_i^R - F_i^L \le \left(F_i^R \left(1 - d_i^a \right) + d_i^b \right) \\
&\text{for } (F_i^R < F_i^L): \qquad F_i^R = F_i^L\, d_i^a - d_i^b \\
&\qquad\qquad\qquad\qquad\ \ F_i^R - F_i^L \ge -F_i^L + F_i^L\, d_i^a - d_i^b \\
&\qquad\qquad\qquad\qquad\ \ F_i^R - F_i^L \ge -\left(F_i^L \left(1 - d_i^a \right) + d_i^b \right)
\end{aligned}
\tag{3.62}
$$

Assembling (3.61) and (3.62) yields final (in)equations for force difference at node *i*:

EXTERNAL CABLE WITH DEVIATORS AND PRESCRIBED BOND:

$$
\text{for } (F_i^R > F_i^L): \qquad F_i^R - F_i^L \le F_i^R \left(1 - d_i^a \right) + d_i^b + \frac{l_i + l_{i-1}}{2} p\ \sigma_c + p \frac{l_i + l_{i-1}}{2} \frac{\partial \sigma_c}{\partial \delta_i} \Delta \delta_i^k \tag{3.63}
$$

$$
\text{for } (F_i^R < F_i^L): \qquad F_i^R - F_i^L \ge -\left(F_i^L \left(1 - d_i^a \right) + d_i^b + \frac{l_i + l_{i-1}}{2} p\ \sigma_c + p \frac{l_i + l_{i-1}}{2} \frac{\partial \sigma_c}{\partial \delta_i} \Delta \delta_i^k \right)
$$

Note that at this stage we solve for slips δ, (while keeping constant cable displacements u). As the reference cohesion stress is a function of δ, i.e. $\sigma_{c0} = \sigma_{co}(\delta...)$, in the above equations we use its Taylor approximation $\sigma_c + \frac{\partial \sigma_c}{\partial \delta_i} \Delta \delta_i$

The above set of (in)equations is calculated in iterative manner. Assume we know the forces at iteration $(k-1)$, then the forces at iteration *k* are:

$$F_i^{R,k-1} = \frac{EA_i}{l_i}(u_{i+1} - u_i + \delta_{i+1}^{k-1} - \delta_i^{k-1})$$
$$F_i^{L,k-1} = \frac{EA_{i-1}}{l_{i-1}}(u_i - u_{i-1} + \delta_i^{k-1} - \delta_{i-1}^{k-1})$$

$$F_i^{R,k} = F_i^{R,k-1} + \frac{EA_i}{l_i}(\Delta\delta_{i+1}^k - \Delta\delta_i^k)$$
$$F_i^{L,k} = F_i^{L,k-1} + \frac{EA_{i-1}}{l_{i-1}}(\Delta\delta_i^k - \Delta\delta_{i-1}^k) \qquad (3.64)$$

$$\delta_{i-1}^k = \delta_{i-1}^{k-1} + \Delta\delta_{i-1}^k$$
$$\delta_i^k = \delta_i^{k-1} + \Delta\delta_i^k$$
$$\delta_{i+1}^k = \delta_{i+1}^{k-1} + \Delta\delta_{i+1}^k$$

and

for $(F_i^R > F_i^L)$:

$$F_i^R + \frac{EA_i}{l_i}(\Delta\delta_{i+1}^k - \Delta\delta_i^k) - F_i^L - \frac{EA_{i-1}}{l_{i-1}}(\Delta\delta_i^k - \Delta\delta_{i-1}^k) - \frac{l_i + l_{i-1}}{2} p \frac{\partial\sigma_c}{\partial\delta_i}\Delta\delta_i^k \leq$$

$$F_i^R\left(1 - d_i^a\right) + d_i^b + \frac{l_i + l_{i-1}}{2} p\ \sigma_c$$

$$\left(-\frac{EA_{i-1}}{l_{i-1}}\right)\Delta\delta_{i-1}^k + \left(\frac{EA_{i-1}}{l_{i-1}} + \frac{EA_i}{l_i}\right)\Delta\delta_i^k + \left(-\frac{EA_i}{l_i}\right)\Delta\delta_{i+1}^k + \frac{l_i + l_{i-1}}{2} p \frac{\partial\sigma_c}{\partial\delta_i}\Delta\delta_i^k \geq$$

$$\left(F_i^R - F_i^L\right) - \left(F_i^R\left(1 - d_i^a\right) + d_i^a + \frac{l_i + l_{i-1}}{2} p\ \sigma_c\right)$$

for $(F_i^R < F_i^L)$:

$$F_i^R + \frac{EA_i}{l_i}(\Delta\delta_{i+1}^k - \Delta\delta_i^k) - F_i^L - \frac{EA_{i-1}}{l_{i-1}}(\Delta\delta_i^k - \Delta\delta_{i-1}^k) + \frac{l_i + l_{i-1}}{2} p \frac{\partial\sigma_c}{\partial\delta_i}\Delta\delta_i^k \geq$$

$$-\left(F_i^L\left(1 - d_i^a\right) + d_i^b + \frac{l_i + l_{i-1}}{2} p\ \sigma_c\right)$$

$$\left(-\frac{EA_{i-1}}{l_{i-1}}\right)\Delta\delta_{i-1}^k + \left(\frac{EA_i}{l_i} + \frac{EA_{i-1}}{l_{i-1}}\right)\Delta\delta_i^k + \left(-\frac{EA_i}{l_i}\Delta\delta_{i+1}^k\right) + \frac{l_i + l_{i-1}}{2} p \frac{\partial\sigma_c}{\partial\delta_i}\Delta\delta_i^k \leq$$

$$\left(F_i^R - F_i^L\right) + \left(F_i^L\left(1 - d_i^a\right) + d_i^b + \frac{l_i + l_{i-1}}{2} p\ \sigma_c\right)$$

(3.65)

If the above equation is written for all nodes on the bar, we obtain a set of inequalities. It has to be solved in iterative manner (within each iteration of the main solution loop).

Atena also support so called CCBarWithMemoryBond 2D and 3D elements. They differ from their original formulation, (i.e. elements CCBarWithBond), in that they have different function $f_\delta(\delta)$ for "loading“ and $f_{\delta,unload}(\delta)$ for "unloading" regime. This means $\delta \notin (\delta_{\min}, \delta_{\max})$ in the former and $\delta \in (\delta_{\min}, \delta_{\max})$ in the latter case.

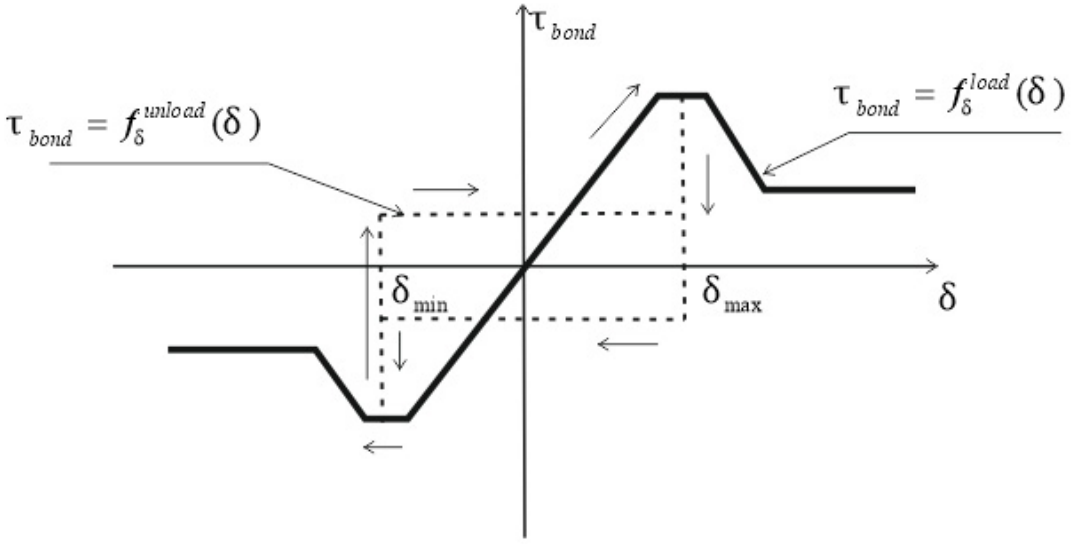


Fig. 3-20 Bond function for CCBarWithMemoryBond element

To obtain more realistic shape, the resulting cohesion stresses are prior their output smoothed. The smoothing operation for node *i* is expressed as follows:

$$\tilde{\sigma}_{right} = \frac{\sigma_{i+1} l_{i+1} + \sigma_i l_i}{l_{i+1} + l_i}$$

$$\tilde{\sigma}_{left} = \frac{\sigma_i l_i + \sigma_{i-1} l_{i-1}}{l_i + l_{i-1}} \tag{3.66}$$

$$\sigma_c = \frac{(\tilde{\sigma}_{right} - \tilde{\sigma}_{left}) A}{p l_i}$$

The equation (3.58) together with (3.61) completes the element description. The element can be used to realistically model cohesion between reinforcement bar and concrete. Such a model is needed for analysis of pullout tests etc. Although the adopted solution is simple, it provides reasonable results accuracy at low computation cost. A more elaborate model of cohesion between reinforcement bar and surrounding concrete can be achieved by using special interface elements that is described in the next section.

3.10 Interface Element

The interface elements are used to model a contact between two surfaces. Currently, the following element types are available: CCIsoCCIsoGap<xxxx> and CCIsoGap<xxxxxx>, CCIsoGap<xxxxxxxx> for 2D and 3D analysis, respectively. These elements use linear approximation of geometry. For the case of nonlinear geometry, use element type

CCIsoGap<xxxxxx> for 2D and CCIsoGap<xxxxxxxxxxxx> or CCIsoGap<xxxxxxxxxxxxxxxx> for 3D. The string in < > describes present element nodes, (see *Atena Input File Format* document for more information). The elements are derived from the corresponding isoparametric elements (described in sections 3.3 and 3.4), i.e. they use the same geometry and nodal ids etc. Geometry of the supported gap elements is depicted in Fig. 3-21.

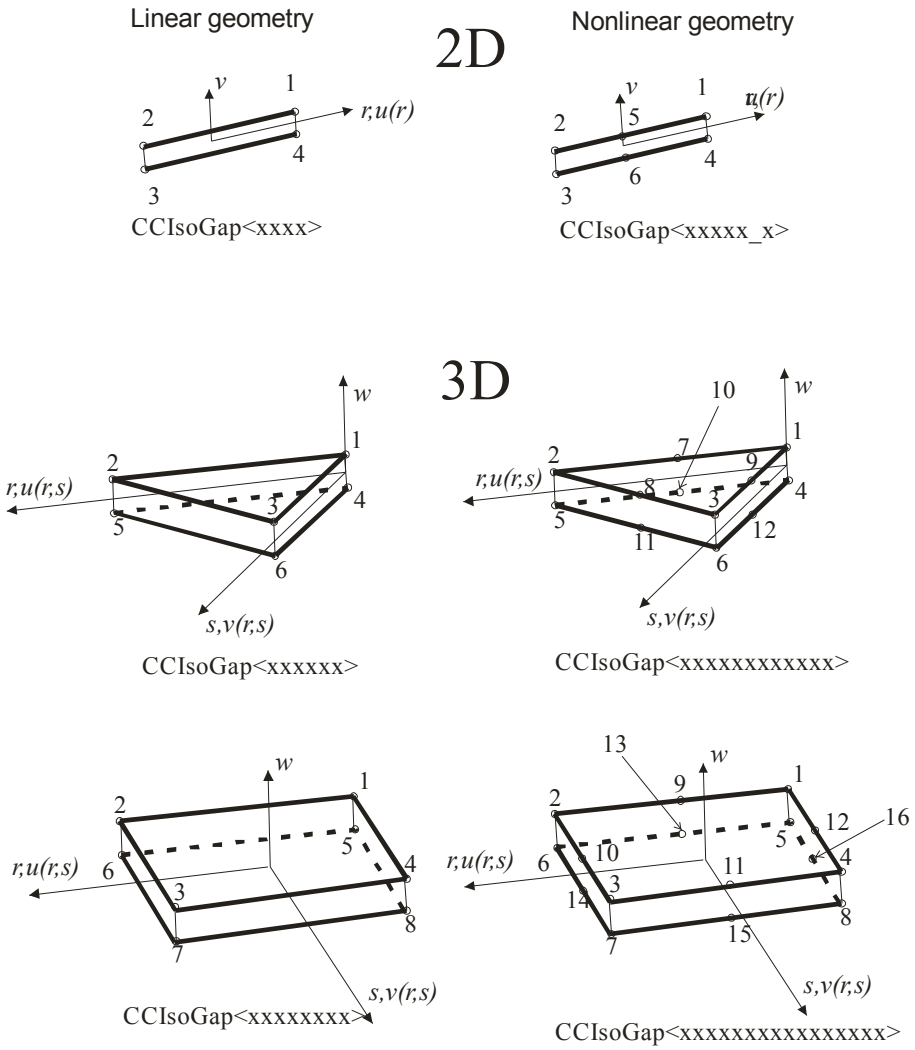


Fig. 3-21 CCIsoGap elements

The interface is defined by a pair of lines, (or surfaces in 3D) each located on the opposite side of interface. In the original (i.e. undeformed) geometry, the interface lines/surfaces can share the same position, or they can be separated by a small distance. In this case we speak about the interface with nonzero thickness.

In the following, the interface behavior is explained on a simple 2-dimensional case, see section 2.6 for a full description of the interface material.

The interface element has two states:

- Open state: There is no interaction of the contact sides.
- Closed state: There is full interaction of the contact sides. In addition, friction sliding of the interface is possible in case of interface element with a friction model.

Penalty method is employed to model the above behavior of the interface. For this purpose, we define a constitutive matrix of the interface in the form:

$$\underline{F} = \begin{Bmatrix} F_\tau \\ F_\sigma \end{Bmatrix} = \begin{bmatrix} K_{tt} & 0 \\ 0 & K_{nn} \end{bmatrix} \begin{Bmatrix} \Delta u \\ \Delta v \end{Bmatrix} = \mathbf{D}\underline{u} \tag{3.67}$$

in which $\Delta u, \Delta v$ are the relative displacements of the interface sides (sliding and opening displacements of the interface) in the local coordinate system r, s and K_{tt}, K_{nn} are the shear and normal stiffness, respectively. This coefficient can be regarded as stiffness of one material layer (real, or fictious) having a finite thickness. The layer is only a numerical tool to handle the gap opening and closing. F_τ, F_σ are forces at the interface, (again at the local coordinate system).

The actual derivation of gap elements is now demonstrated for the case of linear 2D gap element CCIsoGap<xxxx>, see Fig. 3-21. The other elements are constructed in a similar way.

The element has two degrees of freedom defined in the local coordinate system, which is aligned with the gap direction. They are relative displacements $\Delta v, \Delta u$ and are defined as follows:

$$h_1 = \frac{1}{2}(1+r), \qquad h_2 = \frac{1}{2}(1-r)$$

$$\Delta\underline{u} = \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} h_1 \Delta u_{1,4} + h_2 \Delta u_{2,3} \\ h_1 \Delta v_{1,4} + h_2 \Delta v_{2,3} \end{bmatrix}$$

$$\Delta\underline{u} = \begin{bmatrix} h_1 & 0 & h_2 & 0 & -h_2 & 0 & -h_1 & 0 \\ 0 & h_1 & 0 & h_2 & 0 & -h_2 & 0 & -h_1 \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{bmatrix} = \mathbf{B}\underline{u} \tag{3.68}$$

The rest of the element derivation is the same as in case of any other elements, i.e. the stiffness matrix $\mathbf{K} = \int \mathbf{B}^T \mathbf{D} \mathbf{B} dV$, vector of internal forces $\underline{Q} = \int \mathbf{B}^T \underline{F} dV$ etc. A numerical integration in two Gauss points is used to integrate the interface element stiffness matrix. The matrix **K** and the vector Q are in local coordinate system and therefore before they are assembled in the problem governing equations, they must be transformer in global coordinates.

The stiffness coefficients depend on the gap state. The interface is considered open, if the normal force $F_\sigma > R_{ti}$ (R_{ti} is the interface tensile strength force) and the corresponding constitutive law is (stress free interface):

$$\begin{Bmatrix} F_\tau \\ F_\sigma \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \tag{3.69}$$

The stiffness coefficients are set to small, but nonzero values K_{tt}^{op}, K_{nn}^{op}.

The interface element is considered closed if $F_\sigma \leq R_{ti}$. The stiffness coefficients are set to large values K_{tt}^{cl}, K_{nn}^{cl}. It should be noted that the stiffness coefficients are defined only for the purpose of the numerical iterative solution. (Hint: The values of coefficients in the closed state (the large values) are based on thickness comparable to the size of neighbor quadrilateral elements. The minimum values in the open state can be about 1000 times smaller.)

The interface thickness in the out-of plane direction is normally provided as an input parameter. In the case of axi-symmetric analysis it is however calculated using the formula:

$$t = 2\,\pi\,x \tag{3.70}$$

where x is the distance from the axis of symmetry.

There are two special options for processing the gap elements:

Initial gap opening

It is possible to "open" gap at a particular load step, typically the first step of the analysis, i.e. we can introduce to the gaps something like initial element strains in case of ordinary finite elements. This is achieved by LOAD INITIAL GAP ... INIT_STEP_ID *step_id* command. Upon that, during calculation of the (gap) element at the step *step_id* an artificial opening of the interface is introduced. Its value is the distance between upper and lower element surfaces/lines (with reference to undeformed structural shape).

The GAP element load is typically used as follows: we have a structure with a base and upper block. The upper block falls towards the base block that is typically fixed. The structure is solved by introducing a layer of gap elements between the base and upper blocks and applying the GAP element load (for these gaps elements) in the 1st step. As a result, in the first steps the gaps will open to the distance between the blocks. It involves some tensional forces, but as the interface material usually sustains only compression forces, they can be neglected. In next steps the upper block gradually is falling to the base block until it hits it. At this moment interface gaps get fully closed, they change their regime form tension to compression and the upper block gets fully supported by the base block.

Moving gaps[3]

Suppose we have a structure has a base block and an upper block sitting on the base block. The base block is fixed, the upper block is dragged on the upper surface of the base block. The blocks are not mutually interconnected, only some friction and cohesion forces exist between them.

Such problems can be modelled by the RESET_DISPLS *n* flag for the CC2DInterface / CC3DInterface. If this flag is input, then the upper and bottom surface/lines for all corresponding elements are realigned at the end of each step as shown for 2D elements in the following picture. The 3D gaps element is realigned in the same way.

Of course, the boundary surface/lines projection of the gap interface (and thus its "moving" can be used in more complex situation, but the essence of the described technique remains the same. The layer of interface elements is typically connected to the bottom/ upper block of structure by MASTER SLAVE NODAL LISTS boundary conditions, where we must not forget to use the flag PROCESS_FLAG USE_CURRENT_COORDS. It will assure that after realigning the interface gets properly connected to the rest of the (deformed) structure.

[3] Available starting from ATENA version 4.3.1.

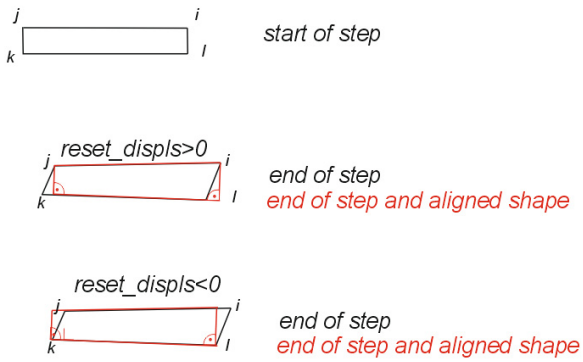


Fig. 3-22 Moving gap 2D element

Note that the option of the gap's initial opening and the reset displacements flag can be combined. Both these special processing options are possible, because the ATENA software uses incremental approach to solve the structure. Thus, changing shape of the gap (at the end of the steps) will not harm governing equilibrium equations.

3.11 Truss Axi-Symmetric Elements.

In the following a circumferential truss element for axisymmetric analysis are described. The elements call CCCircumferentialTruss and CCCircumferentialTruss2 and they are aimed mainly for modeling structural circumferential reinforcement. For radial reinforcement refer to CCIsoTruss<xx> and CCIsoTruss<xxx> elements.

The CCCircumferentialTruss has one node only, whereas the CCCircumferentialTruss2 has nodes two. They behave much the same, the difference being only in calculation of their "cross-sectional area". In case of the CCCircumferentialTruss element the area is entered directly from input data. The CCCircumferentialTruss2 element calculate the area as its thickness (defined in its geometry data) multiplied by its length. Unlike isoparametric elements thses elements are derived and computed analytically.

Geometry, interpolation functions and integration points of the elements are given in

Fig. 3-23.

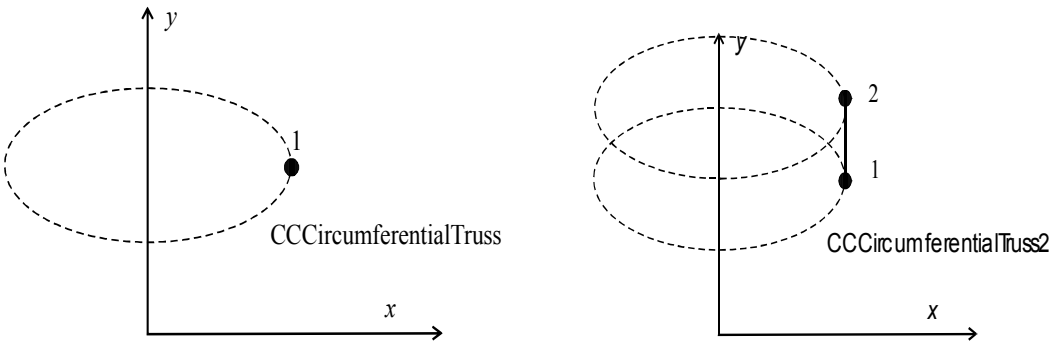


Fig. 3-23 Geometry of CCCircumferentialTruss and CCCircumferentialTruss2 elements.

In the following structural vectors and matrices for the CCIsoTruss element are derived. Development of the CCIsoTruss2 is much the same. In fact, it is CCIsoTruss acting at the centre-point of the CCIsoTruss2 element with its cross-sectional area calculated as explained above.

The element vectors and matrices for Total Lagrangian formulation (TL), configuration at time t and iteration $^{(i)}$ are as follows. Note that they are equally applicable for Updated Lagrangian formulation (UL) upon applying changes related to the element reference co-ordinate system (undeformed vs. deformed element axis.).

The truss element center has at reference time t and $t+\Delta t^{(i-1)}$ co-ordinates ${}^{t}\underline{X} = [{}^{t}x_1, {}^{t}x_1]$ and ${}^{t+\Delta t^{(i-1)}}\underline{X} = [{}^{t+\Delta t^{(i-1)}}x_1, {}^{t+\Delta t^{(i-1)}}x_1]$, respectively. The element length (at respective time) is its length is ${}^{t}l = 2\pi {}^{t}x_1$ and ${}^{t+\Delta t}l^{(i-1)} = 2\pi ({}^{t}x_1^1 + {}^{t}u_1^{1(i-1)})$.

Increment of Green Lagrange strain ${}_{t}\varepsilon_{11}^{(i)} = {}_{t}^{t+\Delta t}\varepsilon_{11}^{(i)t+\Delta t} - {}_{t}^{t+\Delta t}\varepsilon_{11}^{(i-1)}$ (at time $t+\Delta t$, iteration $^{(i)}$ with to configuration at time t) is calculated:

$$ {}_{t}\varepsilon_{11}^{(i)} = \frac{1}{2}\left(\frac{\left({}^{t+\Delta t}l^{(i)}\right)^2 - \left({}^{t+\Delta t}l^{(i-1)}\right)^2}{\left({}^{t}l\right)^2} \right) \tag{3.71} $$

where truss length ${}^{t+\Delta t}l^{(i)} = 2\pi ({}^{t}x_1^1 + {}^{t}u_1^{1(i-1)} + {}_{t}u_1^{1(i)})$. Note that ${}_{t}u_1^{1(i)}$ is co-ordinate increment $({}^{t+\Delta t^{(i)}}x_1 - {}^{t+\Delta t^{(i-1)}}x_1)$. Substituting expressions for element length into (3.71) yields:

$$ {}_{t}\varepsilon_{11}^{(i)} = \frac{4\pi^2\left(\left({}^{t}x_1^1 + {}^{t}u_1^{1(i-1)} + {}_{t}u_1^{1(i)}\right)^2 - \left({}^{t}x_1^1 + {}^{t}u_1^{1(i-1)}\right)^2\right)}{\left({}^{t}x_1^1\right)^2} = \frac{{}_{t}u_1^{1(i)}}{{}^{t}x_1^1} + \frac{{}^{t}u_1^{1(i-1)}\,{}_{t}u_1^{1(i)}}{\left({}^{t}x_1^1\right)^2} + \frac{1}{2}\left(\frac{{}_{t}u_1^{1(i)}}{{}^{t}x_1^1}\right)^2 \tag{3.72} $$

Separating ${}_{t}u_1^{1(i)}$ from (3.72) and rearranging in matrix form we obtain:

$$ {}_{t}^{t+\Delta t}\boldsymbol{B}_{L0} = \frac{1}{{}^{t}x_1^1} \tag{3.73} $$

$$ {}_{t}^{t+\Delta t}\boldsymbol{B}_{L1}^{(i-1)} = \frac{{}^{t}u_1^{1(i-1)}}{\left({}^{t}x_1^1\right)^2} \tag{3.74} $$

and

$$ {}_{t}^{t+\Delta t}\boldsymbol{B}_{NL}^{(n-1)} = \frac{1}{\left({}^{t}x_1^1\right)^2} \tag{3.75} $$

The 2nd Piola-Kirchhoff stress matrix and tensor are:

$$ {}_{t}^{t+\Delta t}S^{(i-1)} = {}_{t}^{t+\Delta t}\underline{S}^{(i-1)} = [{}_{t}^{t+\Delta t}S_{11}^{(i-1)}] \tag{3.76} $$

The formulation is completed by relationship for element deformation gradient ${}^{t+\Delta t}_{t}X^{(i)}_{1,1}$, which yields:

$${}^{t+\Delta t}_{t}X^{(i)}_{1,1} = 1 + {}_{t}e^{(i)}_{11} = \sqrt{\frac{\left({}^{t}x^{1}_{1} + {}^{t}u^{1(i)}_{1}\right)^{2}}{{}^{t}x^{1}_{1}}} \tag{3.77}$$

where engineering strain ${}_{t}e^{(i)}_{11}$ is calculated by

$${}_{t}e^{(i)}_{11} = \sqrt{\frac{{}^{t+\Delta t}l^{(i)}}{{}^{t}l}} = \frac{\left(\sqrt{4\pi^{2}\left({}^{t}x^{1}_{1} + {}^{t}u^{1(i)}_{1}\right)^{2}} - \sqrt{4\pi^{2}\left({}^{t}x^{1}_{1}\right)^{2}}\right)}{\sqrt{4\pi^{2}\left({}^{t}x^{1}_{1}\right)^{2}}} = \frac{\left(\sqrt{\left({}^{t}x^{1}_{1} + {}^{t}u^{1(i)}_{1}\right)^{2}} - {}^{t}x^{1}_{1}\right)}{{}^{t}x^{1}_{1}} \tag{3.78}$$

3.12 Ahmad Shell Element

This section describes Ahmad shell element implemented in ATENA, see Fig. 3-27. It can be used to model thin as well as thick shell or plate structures. It accounts for both plane and bending structural stiffness. The element features quadratic geometry and displacement approximation and therefore, the element's shape can be non-planar. It is possible to account for structural curvatures. Big advantage of this element is that it is seamlessly connectible to true 3D ATENA elements.

Three modifications of this element are supported, and these are characterized by Lagrangian, Serendipity and Heterosis variant of geometry and displacement field approximation. To avoid or minimize membrane and shear locking of shell element it is further possible to use full integration scheme, as well as reduced and/or selective integration. The problems concerned with combination of displacement approximation and integration scheme with respect to locking phenomena are discussed.

The element is derived in a way similar as the other finite elements, which are described in this manual. Hence, in the present description will concentrate mainly on features that are specific for this element. Following Total Lagrangian formulation of the problem, the principle of virtual displacement is used to assemble incremental form of governing equations of structure.

The present Ahmad element belongs to group of shell element formulation that is based on 3D elements' concept. Nevertheless, it uses some assumptions and restrictions, so that the originally 3D element is transformed into 2D space only. It saves computational time and it also avoids some formulation difficulties pertaining to 3D elements.

The element's in-plane integration is carried out in usual way by Gauss integration scheme, whilst in the 3rd dimension (i.e. perpendicular to mid surface of element) the integration can be done in closed (analytical) form. However, in order to enable accounting for nonlinearity of constitutive equations, the so-called layer concept is used instead. Hence, in the 3rd dimension simple quadrilateral integration is employed.

The present degenerate continuum element was originally proposed by Ahmad et al. (Ahmad, Irons et al. 1970). Following general shell element theory concept, every node of element has five degree of freedom, e.g. three displacements and two rotations in planes normal to mid-surface of element. In order to facilitate a simple connection of this element with other true 3D elements, the (original) five degrees of freedom are transformed into *x,y,z* displacement of a top node and *x,y* displacement of a bottom node degrees of freedom. The two nodes are located on the normal to mid-surface passing thru the original mid-surface element's node, see Fig. 3-28.

The essential point in the element's derivation is that displacements and rotations fields are approximated "independently", (see e.g. (Jendele 1981), where similar approach is used for plates). This means that they are handled separately. Unlike in true Mindlin theory our formulation matches geometric equations automatically. However, a special technique is used to improve the element's shear behavior (Hinton and Owen 1984).

The first formulation of this element proposed by Ahmad was linear but since that time many improvements have been achieved. The most important is the application of reduced or selective integration scheme that reduces or totally removes locking of the element. Also, many authors extended the original formulation to geometrically and later also materially nonlinear analysis. One such an advanced form of the element is the formulation implemented in ATENA.

On input, the Ahmad element uses the same geometry as 20 nodes isoparametric brick element, i.e. CCIsoBrick<xxxxxxxxxxxxxxxxxxxx>, see Fig. 3-27. This is needed, in order to be able to use the same pre- and postprocessors' support for the shell and native 3D brick (i.e. hexahedron) elements. After the 1st step of the analysis, the input geometry will automatically change to the external geometry from Fig. 3-27. As nodes 17 and 18 contain only so-called bubble function, the element is post-processed in the same way is it would be the element CCIsoBrick<xxxxxxxxxxxxxxxx>. Internally, all element's vectors and matrices are derived based on the internal geometry as depicted also in Fig. 3-27.

With shell elements, the best connection at edges is to cut both at 45 degrees, or a different corresponding angle if the thicknesses are not the same, or if connected at other than right angle, see Fig. 3-24 (a). Another option is to use a volume brick element at the corner, which is the only feasible way when more than two shells are connected, see Fig. 3-24 (b). The nodes on the surface connected to the volume element have to be listed in the INTERFACE subcommand in the shell geometry definition for correct behavior. Connecting like in Fig. 3-25 is not recommended, as the master-slave relations induced by the fixed thickness of the shell may cause numerical problems.

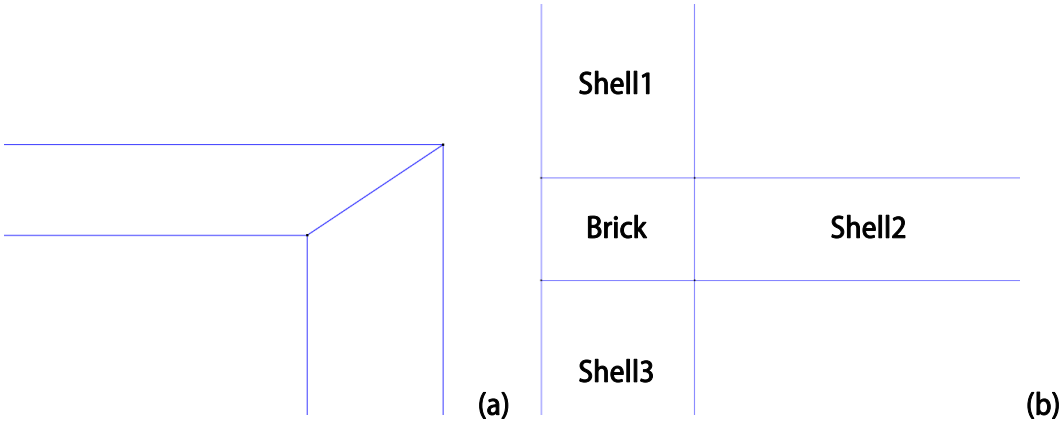


Fig. 3-24: Ahmad Shell - recommended connection (a) 2 shells (b) 3 shells

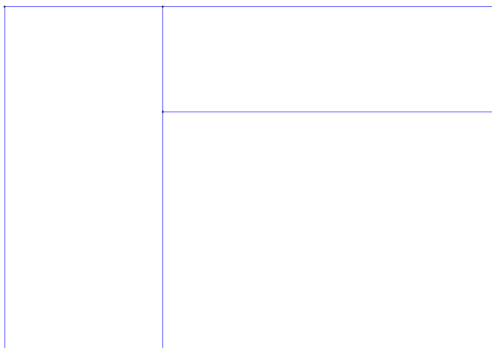


Fig. 3-25: Ahmad Shell - not recommended connection

3.12.1 Coordinate Systems.

The essential point in the element's derivation is to understand coordinate systems that are used within the derivation. These are as follows. Note that all vectors indicating coordinate systems' axes are normalized. Thus, any directional cosines are simply computed as scalar products that need not be divided by the vectors' norm.

Global coordinate system.

It is used to define the whole FE model. Global coordinates are denoted by ${}^{t}x_1, {}^{t}x_2, {}^{t}x_3$, where the index *t* referrers to time. Note that we are using Modified Lagrangian formulation, in which model configuration is updated after each time step, while within one step (for iterating) the configuration from the step beginning is employed. Thus, ${}^{0}x_1, {}^{0}x_2, {}^{0}x_3$ are a point global coordinates prior any load has been applied.

Nodal coordinate system

This coordinate system is defined at each point of element mid-plane surface, i.e. mid-nodes *1-9*. At a node *k* it is specified by vectors ${}^{t}\underline{V_1^k}, {}^{t}\underline{V_2^k}, {}^{t}\underline{V_3^k}$, see Fig. 3-26.

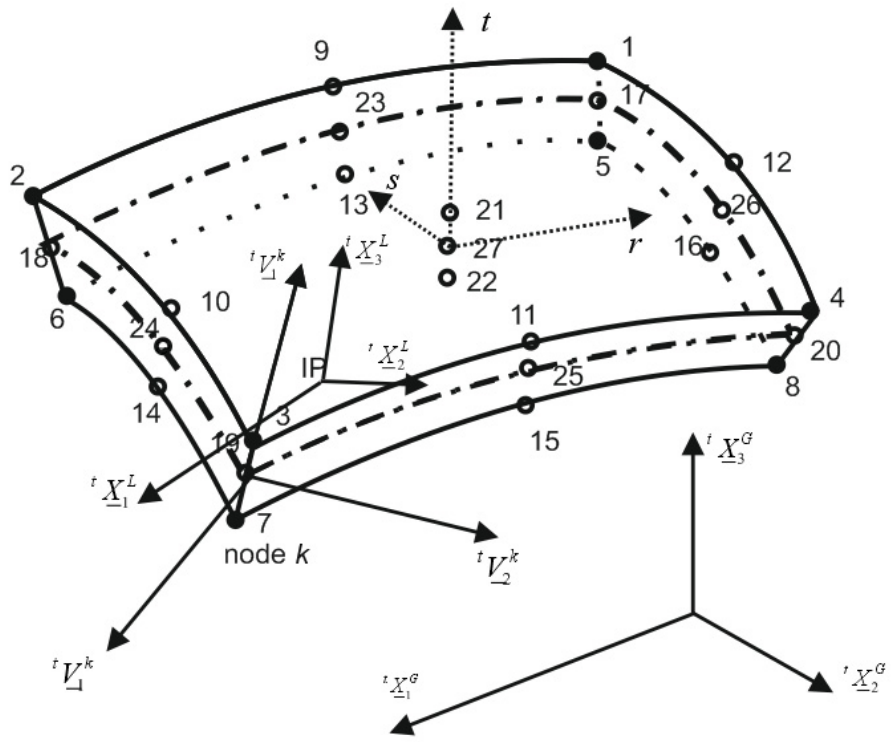


Fig. 3-26 Ahmad element coordinate systems

The vectors ${}^{t}\underline{V}_1^k, {}^{t}\underline{V}_2^k, {}^{t}\underline{V}_3^k$ are defined as follows: Firstly, two auxiliary vectors ${}^{t}\underline{\tilde{V}}_1, {}^{t}\underline{\tilde{V}}_3$ are calculated. Vector ${}^{t}\underline{\tilde{V}}_3$ at a point is defined as a line joining bottom and top coordinates at the node *k* (prior any deformations, i.e. at reference configuration). The second vector ${}^{t}\underline{\tilde{V}}_1$ is normal to ${}^{t}\underline{\tilde{V}}_3$ and is parallel to plane of global ${}^{0}\underline{X}_1^G$ and ${}^{0}\underline{X}_3^G$. Hence:

$$\begin{aligned} {}^{t}\underline{\tilde{V}}_3 &= \left[{}^{t}\underline{\tilde{V}}_{3_1}, {}^{t}\underline{\tilde{V}}_{3_2}, {}^{t}\underline{\tilde{V}}_{3_3} \right] \\ {}^{t}\underline{\tilde{V}}_1 &= \left[{}^{t}\underline{\tilde{V}}_{1_1}, {}^{t}\underline{\tilde{V}}_{1_2}, {}^{t}\underline{\tilde{V}}_{1_3} \right] = \left[{}^{t}\underline{\tilde{V}}_{3_3}, 0, -{}^{t}\underline{\tilde{V}}_{3_1} \right] \end{aligned} \tag{3.79}$$

If ${}^{t}\underline{\tilde{V}}_3$ is parallel to ${}^{0}\underline{X}_2^G$ (i.e. ${}^{t}\underline{\tilde{V}}_{3_1} = {}^{t}\underline{\tilde{V}}_{3_3} = 0$), ${}^{t}\underline{\tilde{V}}_1$ is defined by

$${}^{t}\underline{\tilde{V}}_1 = \left[-{}^{t}\underline{\tilde{V}}_{3_2}, 0, 0 \right] \tag{3.80}$$

After that, the coordinate system ${}^{t}\underline{V}_1^k, {}^{t}\underline{V}_2^k, {}^{t}\underline{V}_3^k$ itself is defined. The vector ${}^{t}\underline{V}_3^k$ is constructed in the same way as was the vector ${}^{t}\underline{\tilde{V}}_3$, however, current, i.e. deformed configuration is used. The remaining two vectors are defined as vector product:

$${}^{t}\underline{V}_2^k = {}^{t}\underline{V}_3^k \otimes {}^{t}\underline{\tilde{V}}_1 \tag{3.81}$$

$${}^{t}\underline{V}_1^k = {}^{t}\underline{V}_2^k \otimes {}^{t} {}^{t}\underline{V}_3^k \tag{3.82}$$

The vectors ${}^t\underline{V}_1^k, {}^t\underline{V}_2^k, {}^t\underline{V}_3^k$ define local nodal shell coordinate system in which the shell rotations are specified. As already mention, the original formulation of the element has 5 DOFs per nodes. These are 3 displacements, expressed in the global coordinate systems and two rotations α, β. They are rotations along the vectors ${}^t\underline{V}_1^k, {}^t\underline{V}_2^k$. It comes from definition that ${}^t\underline{V}_3$ need not be normal to the element surface. It must only connect the top and bottom nodes of the shell.

Sometimes, it is advantageous to modify the nodal coordinate system so that ${}^t\underline{V}_3^k$ remains unchanged but ${}^t\underline{V}_1^k$ and ${}^t\underline{V}_2^k$ are rotated (along ${}^t\underline{V}_3^k$) to a certain direction. Note however, that mutual orthogonality of ${}^t\underline{V}_1^k, {}^t\underline{V}_2^k, {}^t\underline{V}_3^k$ must not be damaged.

Local coordinate system

Local coordinates are denoted by ${}^t x_1^L, {}^t x_2^L, {}^t x_3^L$. The system refers to coordinate axes ${}^t\underline{X}_1^L, {}^t\underline{X}_2^L, {}^t\underline{X}_3^L$. It is used mainly at sampling (integration) points to calculate strains and stresses. The vector axes ${}^t\underline{X}_1^L, {}^t\underline{X}_2^L, {}^t\underline{X}_3^L$ are defined by:

$$ {}^t\underline{X}_3^L = \begin{bmatrix} \frac{\partial {}^t x_1}{\partial r} \\ \frac{\partial {}^t x_2}{\partial r} \\ \frac{\partial {}^t x_3}{\partial r} \end{bmatrix} \otimes \begin{bmatrix} \frac{\partial {}^t x_1}{\partial s} \\ \frac{\partial {}^t x_2}{\partial s} \\ \frac{\partial {}^t x_3}{\partial s} \end{bmatrix} \tag{3.83} $$

$$ \begin{aligned} {}^t\underline{X}_2^L &= {}^t\underline{X}_3^L \otimes {}^t\underline{V}_1^k \\ {}^t\underline{X}_1^L &= {}^t\underline{X}_2^L \otimes {}^t\underline{X}_3^L \end{aligned} \tag{3.84} $$

As the nodal coordinate system ${}^t\underline{V}_1^k, {}^t\underline{V}_2^k, {}^t\underline{V}_3^k$ can rotate along ${}^t\underline{V}_3^k$, the local coordinate system would ${}^t\underline{X}_1^L, {}^t\underline{X}_2^L, {}^t\underline{X}_3^L$ rotate simultaneously along ${}^t\underline{X}_3^L$. This definition allows for user defined shell local coordinate system that is common for all shell elements, irrespective of their incidences. Note that unlike ${}^t\underline{V}_3^k$ the vector ${}^t\underline{X}_3^L$ is always normal to the element mid-plane surface.

Curvilinear coordinate system

This system is used to calculate derivatives and integration in element integration points. Its coordinates are r, s for in-plane direction and t in direction of element thickness, see Fig. 3-26. The in-plane displacements are approximated by Lagrange, Hetherosis or Serendipity approximation similar 2D isoparametric elements. For the 3rd direction, i.e. through the depth of the element. linear approximation is used within the frame of the shell layer concept.

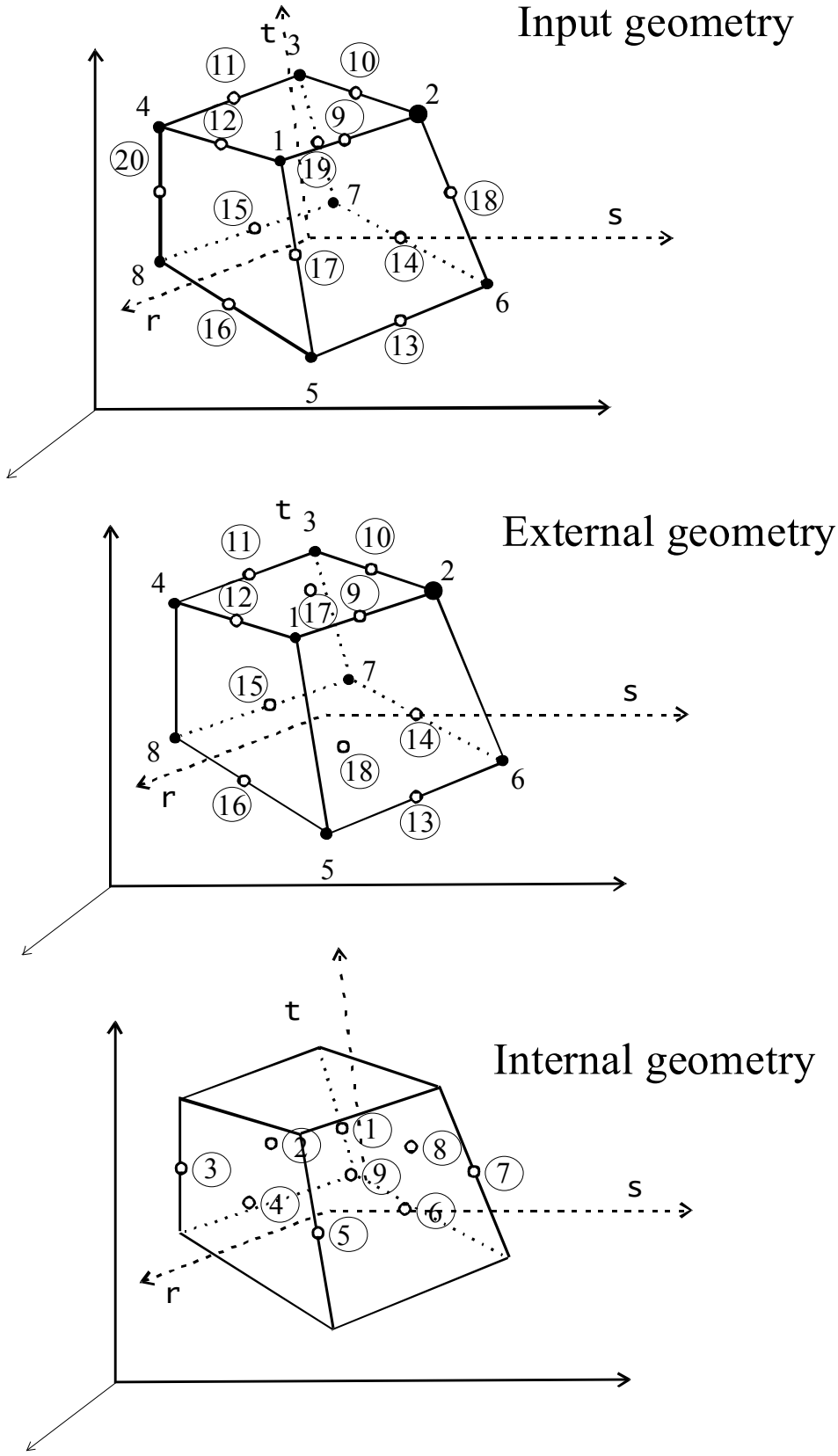


Fig. 3-27 Geometry and the element's nodes

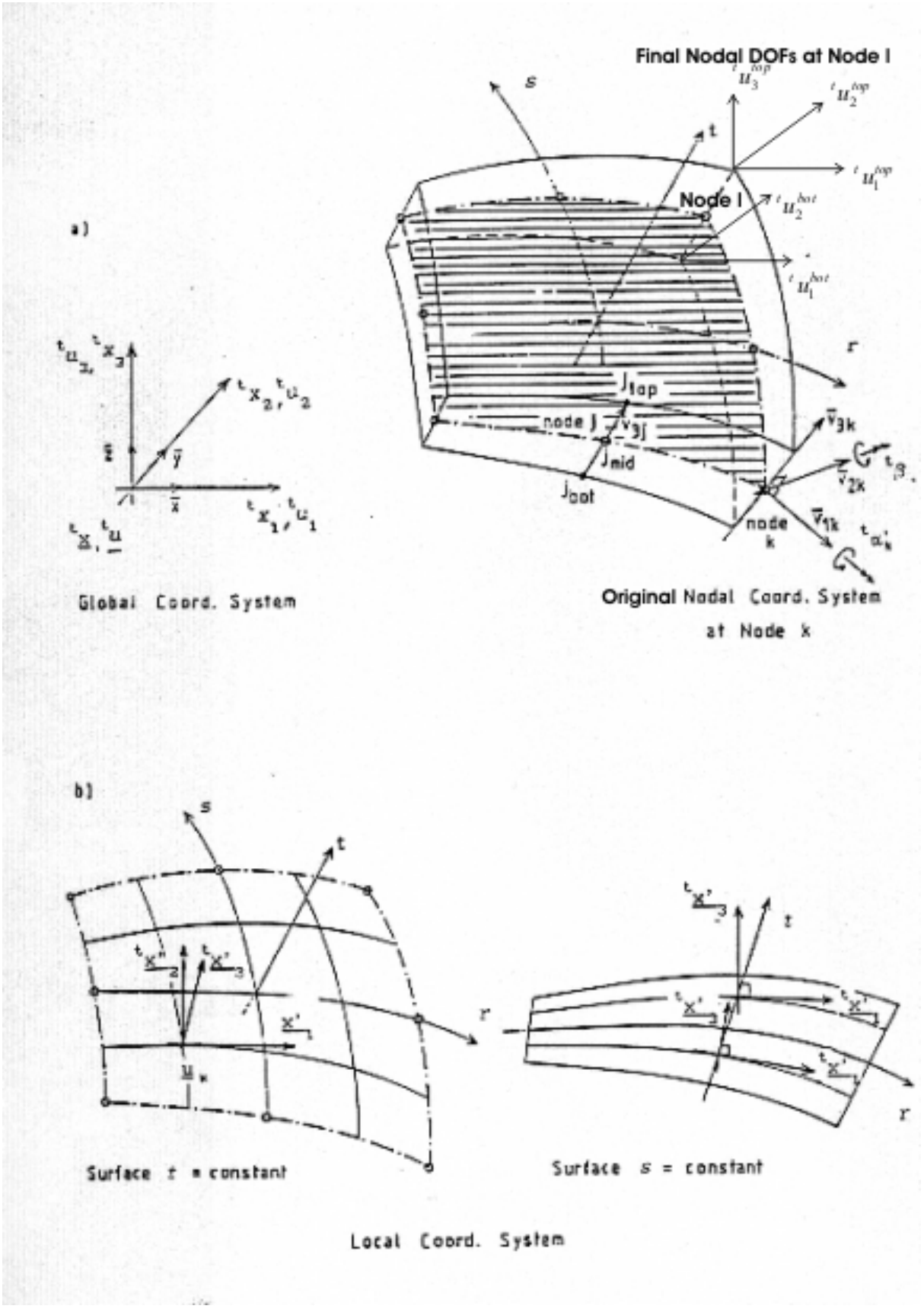


Fig. 3-28 Degenerate shell Ahmad element – coordinate systems and degree of freedoms (DOFs)

3.12.2 Geometry Approximation

The coordinates of the top and bottom element surface are used to define the element geometry:

$$\begin{bmatrix} {}^t x_1 \\ {}^t x_2 \\ {}^t x_3 \end{bmatrix} = {}^t \underline{x} = \sum_{k=1}^{N} h_k \left(\frac{1+t}{2} \begin{bmatrix} {}^t x_1^{k,top} \\ {}^t x_2^{k,top} \\ {}^t x_3^{k,top} \end{bmatrix} + \frac{1-t}{2} \begin{bmatrix} {}^t x_1^{k,bot} \\ {}^t x_2^{k,bot} \\ {}^t x_3^{k,bot} \end{bmatrix} \right) \tag{3.85}$$

where N=8 is number of nodes per element, (geometry is always interpolated by 8-nodes Serendipity interpolation, irrespective of displacement interpolation), h(r,s) is k-th interpolation function, r,s,t are isoparametric coordinates (see Fig. 3-27), $\begin{bmatrix} {}^t x_1^{k,top} \\ {}^t x_2^{k,top} \\ {}^t x_3^{k,top} \end{bmatrix}$ and $\begin{bmatrix} {}^t x_1^{k,bot} \\ {}^t x_2^{k,bot} \\ {}^t x_3^{k,bot} \end{bmatrix}$ are vector of top and bottom coordinates of point k, see Fig. 3-29.

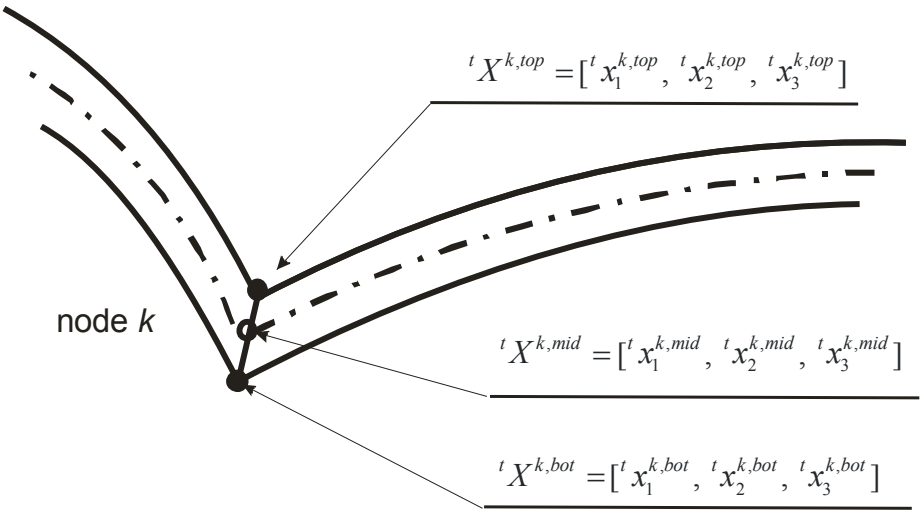


Fig. 3-29 Approximation of the element geometry

Using the above the equation (3.85) can be rewritten in the following form:

$$\begin{bmatrix} {}^t x_1 \\ {}^t x_2 \\ {}^t x_3 \end{bmatrix} = {}^t \underline{x} = \sum_{k=1}^{N} h_k \left(\begin{bmatrix} {}^t x_1^{k,mid} \\ {}^t x_2^{k,mid} \\ {}^t x_3^{k,mid} \end{bmatrix} + \frac{t}{2} \begin{bmatrix} {}^t \underline{V_3^k}_1 \\ {}^t \underline{V_3^k}_2 \\ {}^t \underline{V_3^k}_3 \end{bmatrix} \left[thick \right]_k \right) \tag{3.86}$$

where $\left[thick \right]_k$ is element thickness in node k (i.e. distance between top and bottom points) and

$$\begin{bmatrix} {}^t x_1^{mid} \\ {}^t x_2^{mid} \\ {}^t x_3^{mid} \end{bmatrix}_k = \frac{1}{2} \left[\begin{bmatrix} {}^t x_1^{top} \\ {}^t x_2^{top} \\ {}^t x_3^{top} \end{bmatrix}_k + \begin{bmatrix} {}^t x_1^{bot} \\ {}^t x_2^{bot} \\ {}^t x_3^{bot} \end{bmatrix}_k \right] \tag{3.87}$$

are coordinates of mid surface.

3.12.3 Displacement Field Approximation.

The general concept of displacement approximation is very similar, (although not identical) to geometry approximation. As already mentioned, the original version of Ahmad element uses 5 degrees of freedom per node, see Fig. 3-28. These are $\left[{}^{t}u_1^{mid}, {}^{t}u_2^{mid}, {}^{t}u_3^{mid}, {}^{t}\alpha, {}^{t}\beta \right]^T$, where ${}^{t}u_1^{mid}, {}^{t}u_2^{mid}, {}^{t}u_3^{mid}$ are displacements of the element's node at the mid-surface and ${}^{t}\alpha, {}^{t}\beta$ are rotations with respect to vectors $\underline{v1}_k$ and $\underline{v2}_k$ respectively. These degrees of freedoms (DOFs) are used throughout the whole element's development. However, in order to improve compatibility of the present shell element with other 3D elements implemented in ATENA, externally the element uses $\left[{}^{t}u_1^{top}, {}^{t}u_2^{top}, {}^{t}u_3^{top}, {}^{t}u_1^{bot}, {}^{t}u_2^{bot} \right]^T$ DOFs, i.e. displacements at the top and bottom of the element. The 6th displacement, i.e. u_3^{bot} is eliminated due to application of shell theory that assumes $\sigma_{33} = 0$.

Approximation of the original three "displacement" and two rotation degrees of freedom is independent. Nevertheless, the curvatures used in governing element equations use all of them in the sense dictated by geometric equations. This approach enables to satisfy not only equilibrium equations for membrane stresses and in-plane shear (in mid-surface) as it is the case of popular Kirchhoff hypothesis, but also to satisfy equilibrium condition for transversal shears (normal to mid-surface).

Note that in the following derivation of the element we will deal with the original set of element's DOFs , see (10). Every point thus has five degree of freedom, $\left[{}^{t}u_1^{mid}, {}^{t}u_2^{mid}, {}^{t}u_3^{mid}, {}^{t}\alpha, {}^{t}\beta \right]^T$. Displacement vector is calculated by:

$$\begin{bmatrix} {}^{t}u_1^k \\ {}^{t}u_2^k \\ {}^{t}u_3^k \end{bmatrix} = {}^{t}\underline{u} = \sum_{k=1}^{N} h_k \left(\begin{bmatrix} {}^{t}u_1^{k,mid} \\ {}^{t}u_2^{k,mid} \\ {}^{t}u_3^{k,mid} \end{bmatrix} + \frac{t}{2}[thick]_k \begin{bmatrix} -{}^{t}\underline{V_2^k}_1 & {}^{t}\underline{V_1^k}_1 \\ -{}^{t}\underline{V_2^k}_2 & {}^{t}\underline{V_1^k}_2 \\ -{}^{t}\underline{V_2^k}_3 & {}^{t}\underline{V_1^k}_3 \end{bmatrix} \begin{bmatrix} {}^{t}\alpha^k \\ {}^{t}\beta^k \end{bmatrix} \right) \tag{3.88}$$

The original displacement vector at point *k* has the form $\left[{}^{t}u_1^{mid}, {}^{t}u_2^{mid}, {}^{t}u_3^{mid}, {}^{t}\alpha, {}^{t}\beta \right]^T$. Unlike in the case of geometry approximation, were *N*=8, displacements approximation accounts also for displacement in the element mid-point, i.e. *N*=9. The ninth function *h* is so called bubble function.

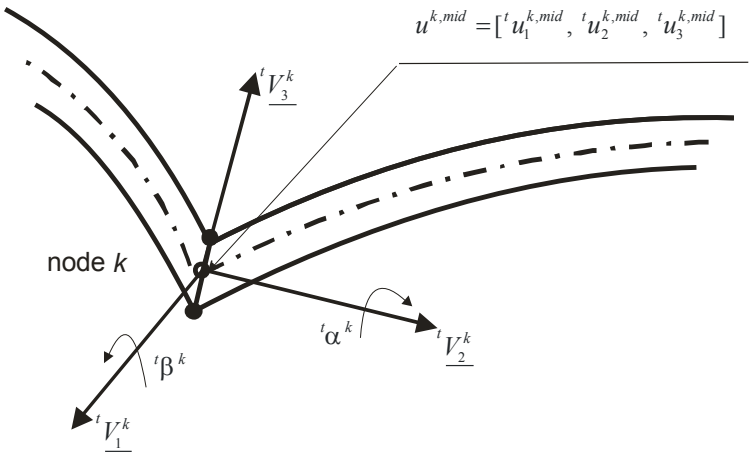


Fig. 3-30 Displacement approximation

3.12.4 Strain and Stresses Definition.

The 2nd Piolla Kirchhoff tensor and Green Lagrange strain tensor is used. They are calculated and printed in the local coordinate system ${}^{t}\underline{x'1}$, ${}^{t}\underline{x'2}$ and ${}^{t}\underline{x'3}$.

Green - Lagrange tensor.

The general definition for Green-Lagrange strain tensor has the form (see eq. (1.8)):

$$ {}^{t}_{0}\varepsilon_{ij} = \frac{1}{2}\left({}^{t}_{0}u_{i,j} + {}^{t}_{0}u_{j,i} + {}^{t}_{0}u_{k,i}\,{}^{t}_{0}u_{k,j} \right) \tag{3.89} $$

Using the above equation and applying the Von-Karman assumption, Eqn. (3.89) can be written as:

$$ \begin{bmatrix} {}^{t}_{0}\varepsilon_{11} \\ {}^{t}_{0}\varepsilon_{22} \\ 2\,{}^{t}_{0}\varepsilon_{12} \\ 2\,{}^{t}_{0}\varepsilon_{13} \\ 2\,{}^{t}_{0}\varepsilon_{23} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial^t u_1}{\partial^t x_1} \\ \dfrac{\partial^t u_2}{\partial^t x_2} \\ \dfrac{\partial^t u_1}{\partial^t x_2} + \dfrac{\partial^t u_2}{\partial^t x_1} \\ \dfrac{\partial^t u_1}{\partial^t x_3} + \dfrac{\partial^t u_3}{\partial^t x_1} \\ \dfrac{\partial^t u_2}{\partial^t x_3} + \dfrac{\partial^t u_3}{\partial^t x_2} \end{bmatrix} + \begin{bmatrix} \dfrac{1}{2}\left(\dfrac{\partial^t u_3}{\partial^t x_1} \right)^2 \\ \dfrac{1}{2}\left(\dfrac{\partial^t u_3}{\partial^t x_2} \right)^2 \\ \dfrac{\partial^t u_3}{\partial^t x_2} + \dfrac{\partial^t u_3}{\partial^t x_1} \\ 0 \\ 0 \end{bmatrix} = {}^{t}_{0}\underline{\varepsilon}_{L} + {}^{t}_{0}\underline{\varepsilon}_{NL} \tag{3.90} $$

The Von-Karman assumptions simplify the calculation of strain by accepting that:

- All strains are relatively small,
- The deflection normal to mid surface of shell is of order of thickness,
- The both curvatures are much smaller than 1.,
- The in-plane displacements are much smaller than transverse displacement and thus their derivatives in 2nd order terms can be neglected.

${}^{t}_{0}\underline{\varepsilon}_{L}$ and ${}^{t}_{0}\underline{\varepsilon}_{NL}$ represents linear and nonlinear part of strain vector, respectively. More information about their calculation is beyond the scope of this publication. It is available e.g. in (Jendele 1992).

2nd Piolla Kirchhoff tensor.

Energetically conjugated with Green - Lagrange tensor is 2nd Piolla Kirchhoff tensor, and this tensor is used by the present shell element. Remind that we account for all stresses with exclusion of normal stress which is perpendicular to shell mid surface (as it is usual practice in shell analysis). This is the reason, why we introduced local coordinate system and all expression are derived with respect to it.

Obviously, the local coordinate system varies depending on element deformation and thus it is necessary to re-compute (each iteration) the transformation matrix **T** (that relates local and global coordinate systems).

To compute internal forces, we will use 2nd Piolla Kirchhoff tensor in vector form (in a node *k*):

$$\left[{}_0^t S \right]_k = \left[{}_0^t S_{11} \quad {}_0^t S_{22} \quad {}_0^t S_{12} \quad {}_0^t S_{13} \quad {}_0^t S_{23} \right]_k \tag{3.91}$$

Note that that it is possible to abbreviate full 3 by 3 element tensor to the above vector, because of adopting Von Karmann simplifying assumption.

3.12.5 Serendipity, Lagrangian and Heterosis Variant of Degenerated Shell Element.

Until now no information about interpolation function *h* and number of integration points were given. The present shell element analysis uses Serendipity interpolation functions. Note that bubble function h_9 (used in displacement approximation only) represents relative departure of approximated function with respect to the function value calculated by previous eight approximation functions.

The interpolation functions h_i read:

$$\begin{aligned}
h_1(r,s) &= \frac{1}{4}(1-r)(1-s)(-r-s-1) \\
h_2(r,s) &= \frac{1}{2}(1-s)(1-r^2) \\
h_3(r,s) &= \frac{1}{4}(1+r)(1-s)(r-s-1) \\
h_4(r,s) &= \frac{1}{2}(1+r)(1-s^2) \\
h_5(r,s) &= \frac{1}{4}(1+r)(1+s)(r+s-1) \\
h_6(r,s) &= \frac{1}{2}(1+s)(1-r^2) \\
h_7(r,s) &= \frac{1}{4}(1+r)(1-s)(r-s-1) \\
h_8(r,s) &= \frac{1}{2}(1-r)(1-s^2) \\
h_9(r,s) &= \frac{1}{2}(1-r^2)(1-s^2)
\end{aligned} \tag{3.92}$$

The actual values in center point can be calculated by:

$$a_9 = \sum_{i=1}^{8} a_i \; h_i(r=0, s=0) + \Delta a_9 \tag{3.93}$$

where h_i are values of interpolation function at point (0,0), a_i are corresponding node values, Δa_9 is departure in the center (i.e. computed value corresponding to degree of freedom at center) and a_9 is total value in center.

Depending on combination how many nodes and integration points are used, we distinguish the Serendipity, Lagrange and Heterosis degenerated element variants, see Fig. 3-31.

Serendipity element.

This element was used in the original Ahmad work. It comprises eight nodal points (center point corresponding to bubble function is omitted).

Gauss integration scheme is used for integration. It can be integrated by full, reduced, or selective integration procedure. Using full integration, i.e. at three by three sample points, element exhibits shear locking for thin and even moderately thick element. If reduced integration is used. the problem of locking is significantly improved without creating spurious energy modes on structure level. However, in case of thin element there are two non communicable spurious energy mode on element level.

It should be noted that there were reported some difficulties if some unfavorable constraints are applied. Nevertheless, the element is popular. If reduced integration is used the provided results are relatively good.

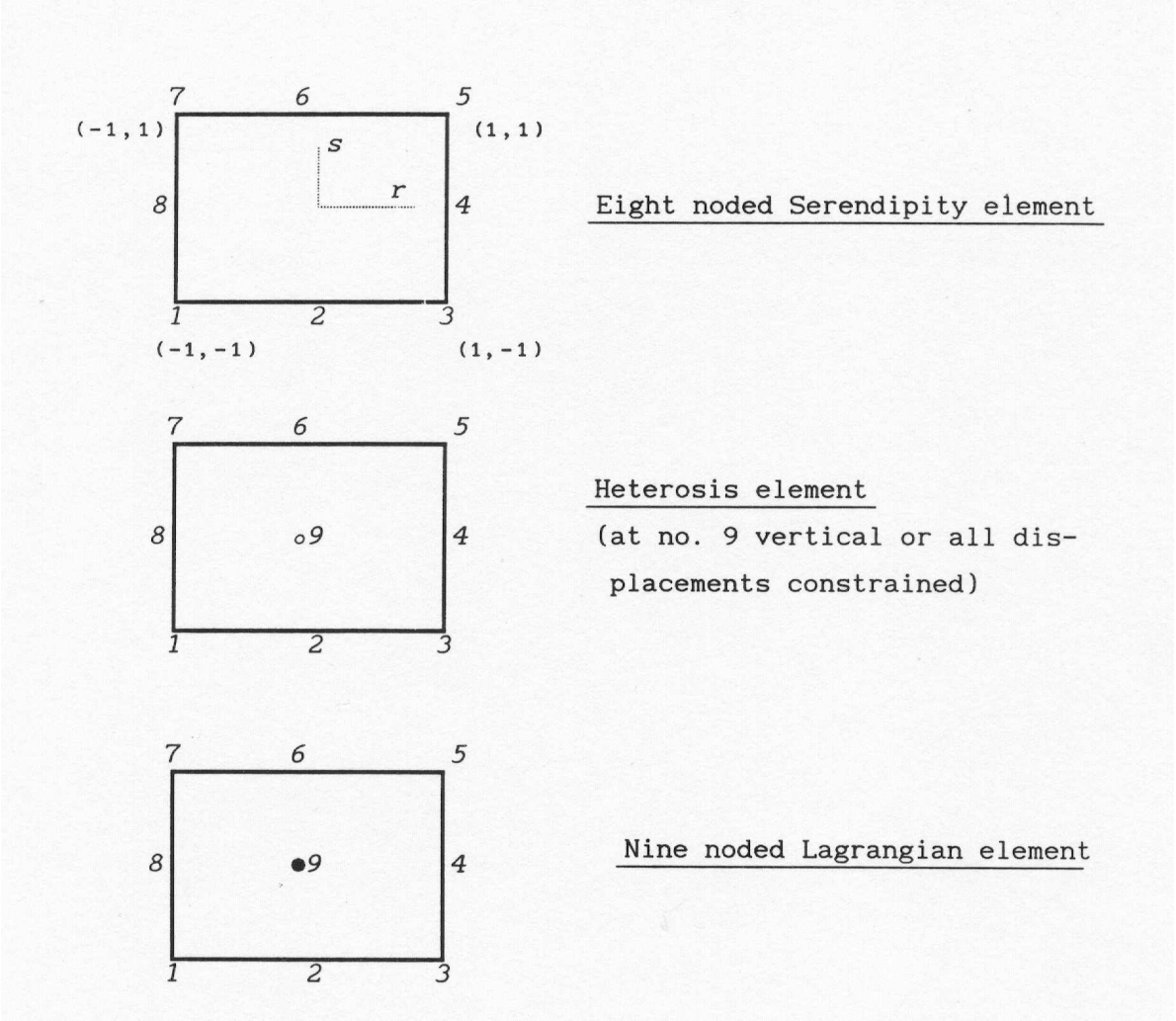


Fig. 3-31 Node notation for element variants of the Ahmad shell element

Nine node Lagrangian element.

The nine-point Lagrangian element is still considered to be the best variant of the degenerated element. This is especially because of its versatility. For full integration scheme there is no problem with membrane and shear locking in very thin plate and shell application. If element is moderately thick, shear locking can be improved by reduced integration scheme. However, in that case the element exhibits rank deficiency.

Heterosis element.

The Heterosis element is very similar to Lagrangian element. The difference is that it assumes first three DOFs at the element centre to be constrained, (i.e. only the rotations are retained)

The element exhibits better behavior compared with previous quadratic elements and especially in combination with selective integration scheme no locking is produced. With reduced integration the element provides results better than Lagrangian element. In that case there are some spurious mechanisms, but for practical solution there are not probable.

Integ. rule	Shear locking	Number of mechanisms		
		Bending	Membrane	Total
	Serendipity 8 node element			
R	$h/l<0.02$	1^*	1^*	2^*
S	no	0	0	0
	Lagrangian 9 node element			
F	$h/l<0.001$	0	0	0
R	no	$3+1^*$	$2+1^*$	$5+2^*$
S	no	1^*	$2+1^*$	$2+2^*$
	Heterosis element			
R	no	$2+1^*$	1^*	$2+2^*$
S	no	$1*$	0	$1*$

* Noncommunicable

F = full integration, S = selective integration

R = reduced integration

Fig. 3-32 Summary of locking and spurious energy modes

Problem of membrane and shear locking in linear analysis are summarized in Fig. 3-32. In the case of nonlinearity, the situation is much more complicated and depends primarily on the material state at the sampling points. For more information refer to (Jendele, Chan et al. 1992)

Element's integration

In previous paragraphs we mentioned many time full, reduced, and selective integration scheme. The sense of these procedures is best to demonstrate in Fig. 3-33.

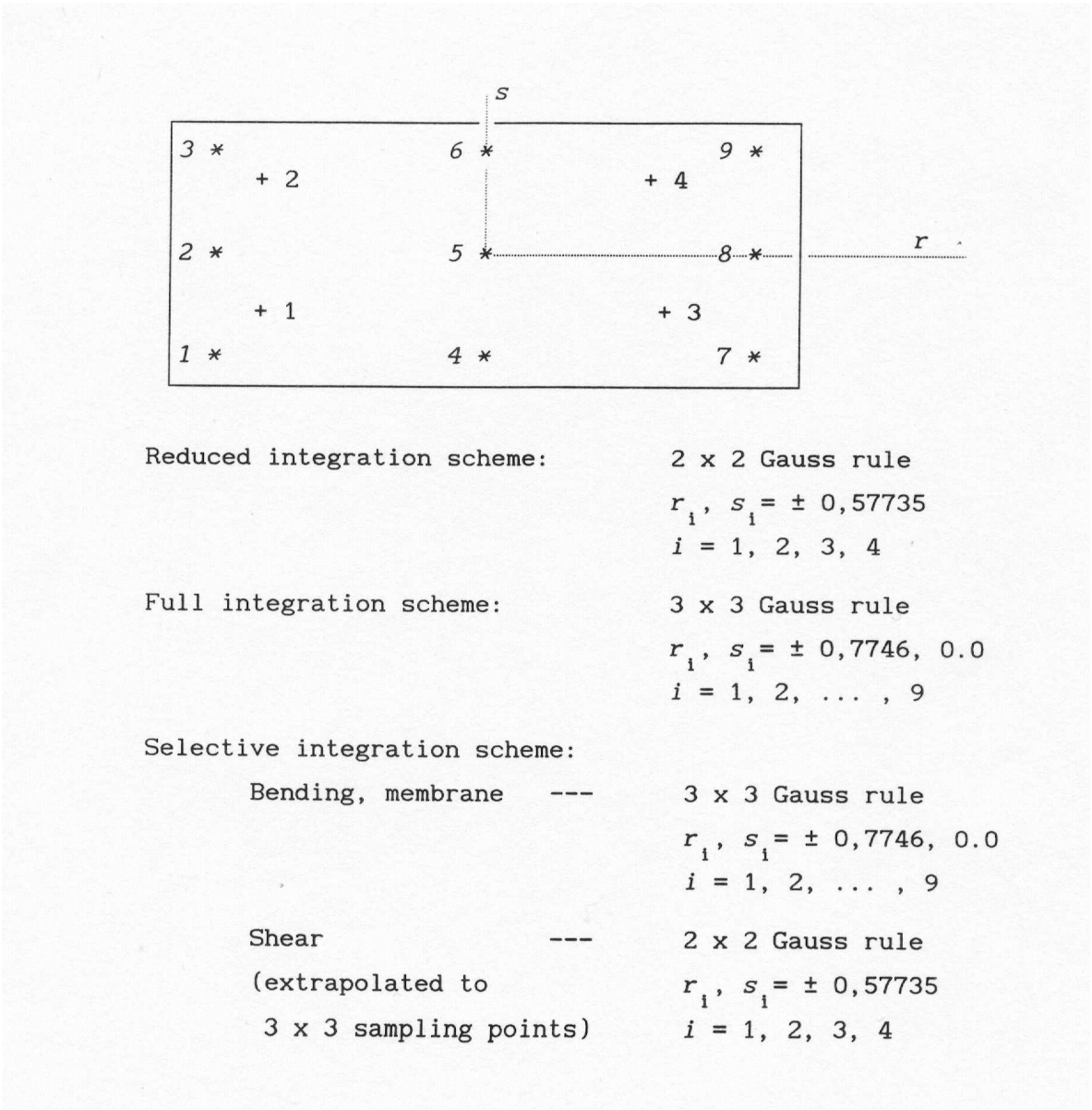


Fig. 3-33 Integration schemes and sampling point notation

The steps during selective integration of shear can been explained on example of integration arbitrary function $f(r,s)$:

1/ First we calculate the value of f at sampling points that corresponds to two-by-two integration rule, i.e

$$f(-0.5773,-0.5773),\ f(-0.5773,0.5773),\ f(0.5773,-0.5773),\ f(0.5773,0.5773)$$

2/ Using bilinear approximation we calculate the values of f at points that correspond to three-by-three integration rule. There are two possibility to that.

The first one is based on original approximation area and the main idea is that we calculate the value of function f at "corners" of isoparametric element (i.e. $r=\pm 1.,s=\pm 1.$):

$$
\begin{aligned}
f(-0.5773,-0.5773) &= \sum_{k=1}^{4} f_i h_i'(-0.5773,-0.5773) \\
f(-0.5773,0.5773) &= \sum_{k=1}^{4} f_i h_i'(-0.5773,0.5773) \\
f(0.5773,-0.5773) &= \sum_{k=1}^{4} f_i h_i'(0.5773,-0.5773) \\
f(0.5773,0.5773) &= \sum_{k=1}^{4} f_i h_i'(0.5773,0.5773)
\end{aligned}
\tag{3.94}
$$

where f_i are element nodal values of function f and h_i' are interpolation functions corresponding to two-by-two interpolation and a node i.

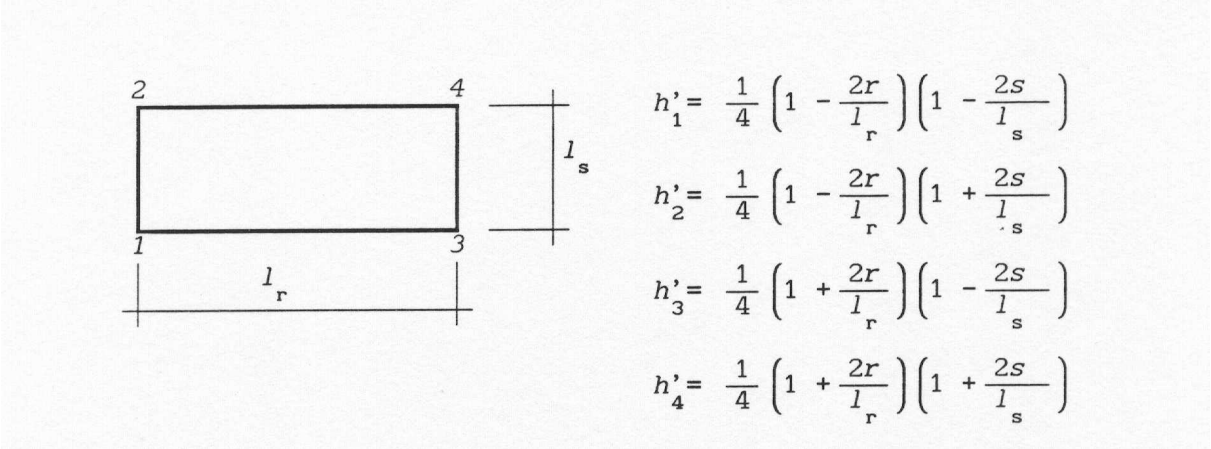


Fig. 3-34 Extension of bilinear approximation function for arbitrary rectangular

The set of equation (3.94) can be solved for f_i. Having these value, we can bi-linearly approximate function f and compute function value at any point, i.e. also at sampling points corresponding to three by three integration rule.

The second and more elegant solution is direct approximation. The interpolation function h_i are presented for a square area of the size 2x2 units, but they can be extended to a rectangular of any size, as shown in Fig. 3-34.

Since the functional values for the 2x2 sampling points in the corner of the square $l_r = l_s = 2x0.5775$ are available, the approximation functions h_i' can be used directly to calculate the values of the function f at sampling points corresponding to the 3x3 integration rule.

For integration in direction perpendicular to r - s plane, that is in t coordinate it is also possible to use Gauss integration, but due to material nonlinearity there is more advantages to use direct rectangular integration. This concept is called the Layered model, see Fig. 3-35.

The main idea of it is to divide the element along the thickness to layers whereby in particular layer the values of strain and stresses are expected to be constant and equal to their value at weight point of layer. Hence the integration in t direction is computed as a sum of integrated expressions multiplicated by adequate area of layer for all layers from bottom to top of element. It was found that to achieve good accuracy it is necessary to about six to ten layers.

This concept. i.e. layer model is advantageous because it enables us to create for example reinforcing layers in element and also we can use finer division near top and bottom of shell, where higher stress level can be expected.

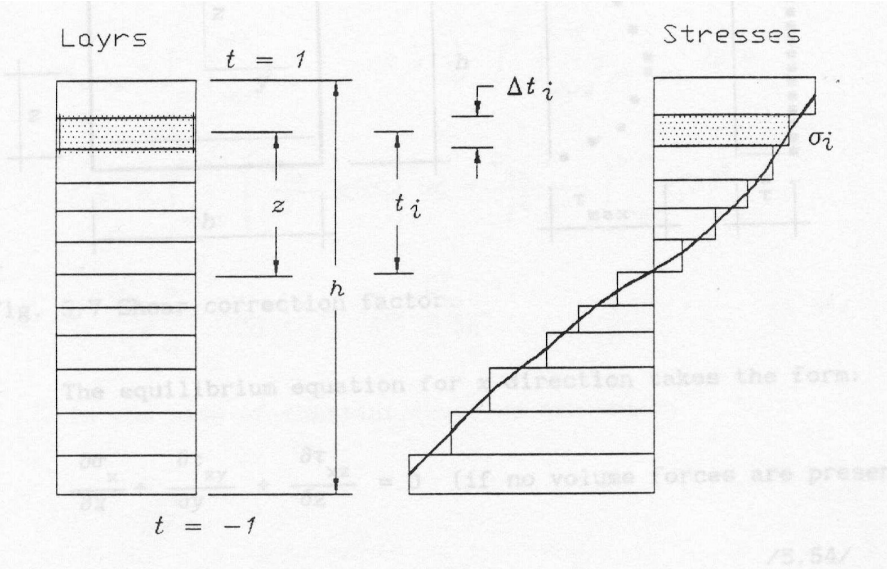


Fig. 3-35 The layer model

3.12.6 Smeared Reinforcement

The ATENA implementation of the Ahmad shell element supports embedding of smeared reinforcement layers. In this concept, reinforcement bars with the same coordinate z, (see Fig. 3-35), material and the same directions are replaced by a layer of smeared reinforcement. Such a layer is placed at the same elevation z as the original reinforcement bars and its thickness is calculated so that sum of cross-sectional area of the bars and the replacing smeared reinforcement layer is the same. The layer is usually superimposed over existing concrete layers and it employs CCSmeardReinforcement material law, which makes possible to account for the original reinforcement bars' direction.

Because each layer of the Ahmad shell can use a distinct material model, concrete and smeared reinforcement layers are treated in similar way. (Constitutive equations, i.e. material law are placed outside of ATENA finite elements' code). Description of syntax of related input commands is beyond scope of this document, but it can be found in the "ATENA Input File Format" document.

Note that the support for smeared reinforcement does not exclude use of structural discrete reinforcement. Both the type of reinforcement can be combined in one model to achieve the best likeness of the the real structure with its numerical model.

3.12.7 Transformation of the Original DOFs to Displacements at the Top and Bottom of the Element Nodal Coordinate System

This section describes in detail the whole procedure of transforming Ahmad elements from its original formulation to the new one used by ATENA SW. Just to remind you: The original formulation (described in the previous sections) differs from the new one in selection of element degree of freedom, see Section 3.12.3.

Let us start to work in nodal coordinate system first. The following equation states transformation rules for transforming three global displacements and two nodal rotations at the element mid-plane, (i.e. the original DOFs at a node k), to 6 displacements at nodal coordinate system, three at the top and three at the bottom surface of the shell. Note the right superscripts "N" that indicate nodal coordinate system.

$$\begin{bmatrix} {}^t u_1^{k,N,top} \\ {}^t u_2^{k,N,top} \\ {}^t u_3^{k,N,top} \\ {}^t u_1^{k,N,bot} \\ {}^t u_2^{k,N,bot} \\ {}^t u_3^{k,N,bot} \end{bmatrix} = \begin{bmatrix} {}^t \underline{V}_1^k{}_1 & {}^t \underline{V}_1^k{}_2 & {}^t \underline{V}_1^k{}_3 & \frac{1}{2}thick & 0 \\ {}^t \underline{V}_2^k{}_1 & {}^t \underline{V}_2^k{}_2 & {}^t \underline{V}_2^k{}_3 & 0 & -\frac{1}{2}thick \\ {}^t \underline{V}_3^k{}_1 & {}^t \underline{V}_3^k{}_2 & {}^t \underline{V}_3^k{}_3 & 0 & 0 \\ {}^t \underline{V}_1^k{}_1 & {}^t \underline{V}_1^k{}_2 & {}^t \underline{V}_1^k{}_3 & -\frac{1}{2}thick & 0 \\ {}^t \underline{V}_2^k{}_1 & {}^t \underline{V}_2^k{}_2 & {}^t \underline{V}_2^k{}_3 & 0 & \frac{1}{2}thick \\ {}^t \underline{V}_3^k{}_1 & {}^t \underline{V}_3^k{}_2 & {}^t \underline{V}_3^k{}_3 & 0 & 0 \end{bmatrix} \begin{bmatrix} {}^t u_1^{k,mid} \\ {}^t u_2^{k,mid} \\ {}^t u_3^{k,mid} \\ {}^t \alpha^k \\ {}^t \beta^k \end{bmatrix} = \boldsymbol{T_1} \begin{bmatrix} {}^t u_1^{k,mid} \\ {}^t u_2^{k,mid} \\ {}^t u_3^{k,mid} \\ {}^t \alpha^k \\ {}^t \beta^k \end{bmatrix} \tag{3.95}$$

Transformation from nodal to global coordinated system

The next step in the element's derivation is to write transformation of the left-hand side vector of (3.95) from nodal to global coordinate system. It reads:

$$\begin{bmatrix} {}^t u_1^{k,top} \\ {}^t u_2^{k,top} \\ {}^t u_3^{k,top} \\ {}^t u_1^{k,bot} \\ {}^t u_2^{k,bot} \\ {}^t u_3^{k,bot} \end{bmatrix} = \begin{bmatrix} {}^t \underline{V}_1^k{}_1 & {}^t \underline{V}_2^k{}_1 & {}^t \underline{V}_3^k{}_1 & 0 & 0 & 0 \\ {}^t \underline{V}_1^k{}_2 & {}^t \underline{V}_2^k{}_2 & {}^t \underline{V}_3^k{}_2 & 0 & 0 & 0 \\ {}^t \underline{V}_1^k{}_3 & \underline{V}_2^k{}_3 & {}^t \underline{V}_3^k{}_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & {}^t \underline{V}_1^k{}_1 & {}^t \underline{V}_2^k{}_1 & {}^t \underline{V}_3^k{}_1 \\ 0 & 0 & 0 & {}^t \underline{V}_1^k{}_2 & {}^t \underline{V}_2^k{}_2 & {}^t \underline{V}_3^k{}_2 \\ 0 & 0 & 0 & {}^t \underline{V}_1^k{}_3 & \underline{V}_2^k{}_3 & {}^t \underline{V}_3^k{}_3 \end{bmatrix} \begin{bmatrix} {}^t u_1^{k,N,top} \\ {}^t u_2^{k,N,top} \\ {}^t u_3^{k,N,top} \\ {}^t u_1^{k,N,bot} \\ {}^t u_2^{k,N,bot} \\ {}^t u_3^{k,N,bot} \end{bmatrix} = \boldsymbol{T_2} \begin{bmatrix} {}^t u_1^{k,N,top} \\ {}^t u_2^{k,N,top} \\ {}^t u_3^{k,N,top} \\ {}^t u_1^{k,N,bot} \\ {}^t u_2^{k,N,bot} \\ {}^t u_3^{k,N,bot} \end{bmatrix} \tag{3.96}$$

Complete transformation of the original DOFs to the new element formulation DOFs

The final transformation from the original to the new element DOFs at a node k is obtain by substituting (3.95) into (3.96). Thus, we can write

$$\begin{bmatrix} {}^t u_1^{k,top} \\ {}^t u_2^{k,top} \\ {}^t u_3^{k,top} \\ {}^t u_1^{k,bot} \\ {}^t u_2^{k,bot} \\ {}^t u_3^{k,bot} \end{bmatrix} = \boldsymbol{T_2}\ \boldsymbol{T_1} \begin{bmatrix} {}^t u_1^{k,mid} \\ {}^t u_2^{k,mid} \\ {}^t u_3^{k,mid} \\ {}^t \alpha^k \\ {}^t \beta^k \end{bmatrix} = \boldsymbol{T} \begin{bmatrix} {}^t u_1^{k,mid} \\ {}^t u_2^{k,mid} \\ {}^t u_3^{k,mid} \\ {}^t \alpha^k \\ {}^t \beta^k \end{bmatrix} \tag{3.97}$$

where $\boldsymbol{T}$

In a very similar way, we can define inverse transformation, i.e. from the new DOFs to original one. Without any derivation the matrix reads:

$$\begin{bmatrix} {}^{t}u_1^{k,mid} \\ {}^{t}u_2^{k,mid} \\ {}^{t}u_3^{k,mid} \\ {}^{t}\alpha^{k} \\ {}^{t}\beta^{k} \end{bmatrix} = \boldsymbol{T'} \begin{bmatrix} {}^{t}u_1^{k,top} \\ {}^{t}u_2^{k,top} \\ {}^{t}u_3^{k,top} \\ {}^{t}u_1^{k,bot} \\ {}^{t}u_2^{k,bot} \\ {}^{t}u_3^{k,bot} \end{bmatrix} \tag{3.98}$$

where $\boldsymbol{T'} = \begin{bmatrix} \frac{T_{11}}{2} & \frac{T_{12}}{2} & \frac{T_{13}}{2} & \frac{T_{11}}{2} & \frac{T_{12}}{2} & \frac{T_{13}}{2} \\ \frac{T_{21}}{2} & \frac{T_{22}}{2} & \frac{T_{23}}{2} & \frac{T_{21}}{2} & \frac{T_{22}}{2} & \frac{T_{23}}{2} \\ \frac{T_{31}}{2} & \frac{T_{32}}{2} & \frac{T_{33}}{2} & \frac{T_{31}}{2} & \frac{T_{32}}{2} & \frac{T_{33}}{2} \\ \frac{{}^{t}\underline{V}_{1\_1}^{k}}{thick^{k}} & \frac{{}^{t}\underline{V}_{1\_2}^{k}}{thick^{k}} & \frac{{}^{t}\underline{V}_{1\_3}^{k}}{thick^{k}} & \frac{-{}^{t}\underline{V}_{1\_1}^{k}}{thick^{k}} & \frac{-{}^{t}\underline{V}_{1\_2}^{k}}{thick^{k}} & \frac{-{}^{t}\underline{V}_{1\_3}^{k}}{thick^{k}} \\ \frac{-{}^{t}\underline{V}_{2\_1}^{k}}{thick^{k}} & \frac{-{}^{t}\underline{V}_{2\_2}^{k}}{thick^{k}} & \frac{-{}^{t}\underline{V}_{2\_3}^{k}}{thick^{k}} & \frac{{}^{t}\underline{V}_{2\_1}^{k}}{thick^{k}} & \frac{{}^{t}\underline{V}_{2\_2}^{k}}{thick^{k}} & \frac{{}^{t}\underline{V}_{2\_3}^{k}}{thick^{k}} \end{bmatrix}$

Constraining the redundant DOF to comply with shell theory

As noted earlier, the original set of DOFs at a node comprises 5 DOFs, whilst the new one has six DOFs. Consequently, one DOF from $\left[{}^{t}u_1^{k,top}, {}^{t}u_2^{k,top}, {}^{t}u_3^{k,top}, {}^{t}u_1^{k,bot}, {}^{t}u_2^{k,bot}, {}^{t}u_3^{k,bot} \right]^T$ must be fixed. The presented work prefers to constrain ${}^{t}u_3^{k,bot}$ but $u_1^{k,bot}$ or $u_2^{k,bot}$ are also good candidates, if ${}^{t}u_3^{k,bot}$ can not be fixed due some numerical problems, usually due to a special position of the element with respect to global coordinate system.

Derivation of the constrain is now demonstrated on the case of ${}^{t}u_3^{k,bot}$. Using (3.97)

$$\begin{aligned} {}^{t}u_3^{k,bot} &= {}^{t}u_3^{k,top} + ({}^{t}u_3^{k,bot} - {}^{t}u_3^{k,top}) = \\ &{}^{t}u_3^{k,top} + \left(\left(T_{16} - T_{13} \right)^{t} u_1^{k,mid} + \left(T_{26} - T_{23} \right)^{t} u_2^{k,mid} + ... \left(T_{56} - T_{53} \right)^{t} \beta^{k} \right) = \\ &{}^{t}u_3^{k,top} - thick^{k}\, {}^{t}\underline{V}_{1\_3}^{k}\, {}^{t}\alpha^{k} + thick^{k}\, {}^{t}\underline{V}_{2\_3}^{k}\, {}^{t}\beta^{k} \end{aligned} \tag{3.99}$$

Now in (3.99) eliminate α^{k} and ${}^{t}\beta^{k}$ using (3.98). Thus, we obtain one equation relating $\left[{}^{t}u_1^{k,top}, {}^{t}u_2^{k,top}, {}^{t}u_3^{k,top}, {}^{t}u_1^{k,bot}, {}^{t}u_2^{k,bot}, {}^{t}u_3^{k,bot} \right]^T$, which is then used to constrain ${}^{t}u_3^{k,bot}$ as a linear combination of ${}^{t}u_1^{k,top}, {}^{t}u_2^{k,top}, {}^{t}u_3^{k,top}, {}^{t}u_1^{k,bot}, {}^{t}u_2^{k,bot}$:

$$u_3^{k,bot} = c_1^{top}\, {}^{t}u_1^{k,top} + c_2^{top}\, {}^{t}u_2^{k,top} + c_3^{top}\, {}^{t}u_3^{k,top} + c_1^{bot}\, {}^{t}u_1^{k,bot} + c_2^{bot}\, {}^{t}u_2^{k,bot} \tag{3.100}$$

where:

$$
\begin{aligned}
c_1^{top} &= \frac{{}^t\underline{V}_1^k{}_3\ {}^t\underline{V}_1^k{}_1 + {}^t\underline{V}_2^k{}_3\ {}^t\underline{V}_2^k{}_1}{\left({}^t\underline{V}_1^k{}_3\right)^2 + \left({}^t\underline{V}_2^k{}_3\right)^2 - 1} \\
c_2^{top} &= \frac{{}^t\underline{V}_1^k{}_3\ {}^t\underline{V}_1^k{}_2 + {}^t\underline{V}_2^k{}_3\ {}^t\underline{V}_2^k{}_2}{\left({}^t\underline{V}_1^k{}_3\right)^2 + \left({}^t\underline{V}_2^k{}_3\right)^2 - 1} \\
c_3^{top} &= -\frac{1 - \left({}^t\underline{V}_1^k{}_3\right)^2 - \left({}^t\underline{V}_2^k{}_3\right)^2 yy_z^2}{\left({}^t\underline{V}_1^k{}_3\right)^2 + \left({}^t\underline{V}_2^k{}_3\right)^2 - 1} \\
c_1^{bot} &= -\frac{{}^t\underline{V}_1^k{}_3\ {}^t\underline{V}_1^k{}_1 + {}^t\underline{V}_2^k{}_2\ {}^t\underline{V}_2^k{}_1}{\left({}^t\underline{V}_1^k{}_3\right)^2 + \left({}^t\underline{V}_2^k{}_3\right)^2 - 1} \\
c_2^{bot} &= -\frac{{}^t\underline{V}_1^k{}_3\ {}^t\underline{V}_1^k{}_2 + {}^t\underline{V}_2^k{}_3\ {}^t\underline{V}_2^k{}_2}{\left({}^t\underline{V}_1^k{}_3\right)^2 + \left({}^t\underline{V}_2^k{}_3\right)^2 - 1}
\end{aligned}
\tag{3.101}
$$

The DOFs $u_1^{k,bot}$ or $u_2^{k,bot}$ can be eliminated in the same way. During the execution of the element, it is recommended to constrain one of ${}^t u_1^{k,bot}, {}^t u_2^{k,bot}, {}^t u_3^{k,bot}$ based on which solution is the most stable, (i.e. maximum denominator in (3.101)).

Constraining DOFs at the centre of Hetherosis element

A special attention needs to be paid to the 9th mid-plane node of Hetherosis element when we have to additionally constrain ${}^t u_1^{k,mid}, {}^t u_2^{k,mid}, {}^t u_3^{k,mid}$. Thus, of the 6 DOFs we need to constrain 4 of them.

For example, suppose we want to keep free ${}^t u_2^{k,top}$ and ${}^t u_3^{k,top}$ and we need to fix ${}^t u_1^{k,top}, {}^t u_1^{k,bot}, {}^t u_2^{k,bot}, {}^t u_3^{k,bot}$. Equation (3.99) from the previous paragraph needs to be added by three more equations. These are:

$$
\begin{bmatrix} {}^t u_1^{k,mid} \\ {}^t u_2^{k,mid} \\ {}^t u_3^{k,mid} \end{bmatrix} = \begin{bmatrix} T_{11}^{'} & T_{12}^{'} & T_{13}^{'} & T_{14}^{'} & T_{15}^{'} & T_{16}^{'} \\ T_{21}^{'} & T_{22}^{'} & T_{23}^{'} & T_{24}^{'} & T_{25}^{'} & T_{26}^{'} \\ T_{31}^{'} & T_{32}^{'} & T_{33}^{'} & T_{34}^{'} & T_{35}^{'} & T_{36}^{'} \end{bmatrix} \begin{bmatrix} {}^t u_1^{k,top} \\ {}^t u_2^{k,top} \\ {}^t u_3^{k,top} \\ {}^t u_1^{k,bot} \\ {}^t u_2^{k,bot} \\ {}^t u_3^{k,bot} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}
\tag{3.102}
$$

Equations (3.99) and (3.102) are then solved for ${}^t u_1^{k,top}, {}^t u_1^{k,bot}, {}^t u_2^{k,bot}, {}^t u_3^{k,bot}$ as a linear combination of ${}^t u_2^{k,top}$ and ${}^t u_3^{k,top}$.

$$\begin{bmatrix} {}^{t}u_1^{k,top} \\ u_1^{k,bot} \\ u_2^{k,bot} \\ {}^{t}u_3^{k,bot} \end{bmatrix} = -\begin{bmatrix} T_{11}^{'} & T_{14}^{'} & T_{15}^{'} & T_{16}^{'} \\ T_{21}^{'} & T_{24}^{'} & T_{25}^{'} & T_{26}^{'} \\ T_{31}^{'} & T_{34}^{'} & T_{35}^{'} & T_{35}^{'} \\ c_1^{top} & -1 & c_1^{bot} & c_2^{bot} \end{bmatrix}^{-1} \begin{bmatrix} T_{12}^{'}\,{}^{t}u_2^{k,top} + T_{13}^{'}\,{}^{t}u_3^{k,top} \\ T_{22}^{'}\,{}^{t}u_2^{k,top} + T_{23}^{'}\,{}^{t}u_3^{k,top} \\ T_{32}^{'}\,{}^{t}u_2^{k,top} + T_{33}^{'}\,{}^{t}u_3^{k,top} \\ c_2^{top}\,{}^{t}u_2^{k,top} + c_3^{top}\,{}^{t}u_3^{k,top} \end{bmatrix} \tag{3.103}$$

Again, there are several alternatives regarding of which of the 6 DOFs to keep and which to eliminate. The best option is chosen the same way as described in Section 0.

3.12.8 Shell Ahmad Elements Implemented in ATENA

Several modifications the Ahmad shell elements are implemented in ATENA. They are listed in the following table:

Table 3-5 Ahmad shell elements.

Element name	Type of approximation	Number of in-plane integration points per axis direction for bending	Number of in-plane integration points per axis direction for shear	Comment
CCAhmadElement33L9	Lagrange	3	3	No spurious modes, locking in this shells
CCAhmadElement32L9	Lagrange	3	2	
CCAhmadElement33H9	Heterosis	3	3	
CCAhmadElement32H9	Heterosis	3	2	Good compromise between locking and spurious energy modes
CCAhmadElement22S8	Serendipity	2	2	Fast, but spurious modes

3.13 Curvilinear Nonlinear 2D Isoparametric Layered Shell Quadrilateral Elements

This section describes shell elements that model a structure by a curvilinear 2D surface. The element uses hierarchical geometry and displacement interpolation. It can have from 4 to 9 nodes, each of them having 5 DOFs: 3 displacements in direction of global *X,Y,Z* axis and 2 rotations along user defined vectors V_1, V_2 . If the shell is in the *XY* plane, then typically $V_1 = X, V_2 = Y$.

The element uses linear geometry and displacement interpolation in the direction of its thickness and quadratic or linear approximation in the element's plane. If quadratic approximation is used, behavior of the element resembles behavior of Ahmad shell element described in the previous section. 4 nodes version of this element, i.e. the element with linear approximation, does not perform well, (the element is too stiff), and thus it is recommended only for some local links etc. On the other hand, both bending and membrane behavior of 8-9 nodes version of the elements is great.

The elements are derived based on the Shell theory, (similarly to Ahmad element). As a result, it is assumed $\varepsilon_t = 0,\ \sigma_t$ is negligible and the element cannot change its thickness. (*t* indicates local axis in the shell's thickness).

Depending on number of element nodes these finite elements call CCIsoShellQuad<xxxx> ... CCIsoShellQuad<xxxxxxxxx>

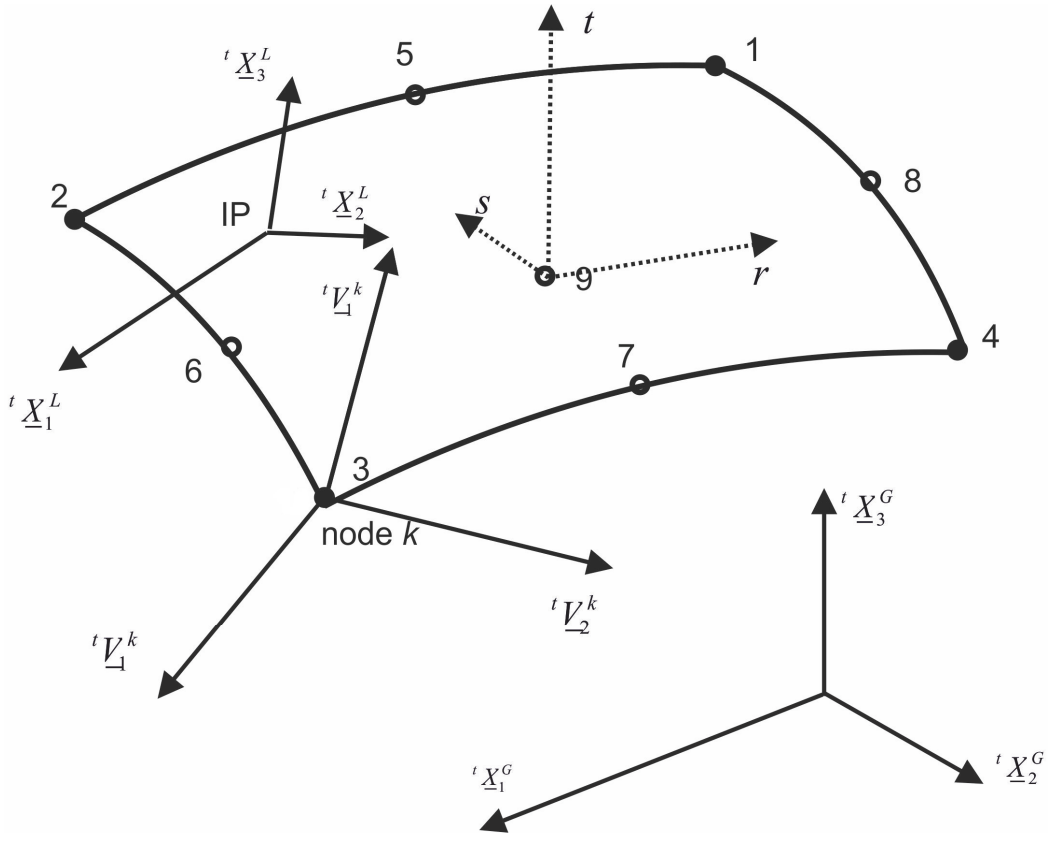


Fig. 3-36 CCIsoShell2D elements

3.13.1 Geometry and displacements

The shell's geometry at the configuration t and $t + dt$ is defined by:

$$
\begin{aligned}
{}^{t}x_i &= h_k\left({}^{t}X_i^k + \frac{t}{2}a_k {}^{t}V_i^{n_k} \right) \\
{}^{t+\Delta t}x_i^{(i-1)} &= h_k\left({}^{t+\Delta t}X_i^{k,(i-1)} + \frac{t}{2}a_k \, {}^{t+\Delta t}V_i^{n_k,(i-1)} \right) \\
{}^{t+\Delta t}x_i^{(i)} &= h_k\left({}^{t+\Delta t}X_i^{k,(i)} + \frac{t}{2}a_k \, {}^{t+\Delta t}V_i^{n_k,(i)} \right)
\end{aligned}
\tag{3.104}
$$

where *i*=1,2,3 is index relating to global axes x_1, x_2, x_3 , (i.e. *x*,*y*,*z*), $k = 1...n_G$, n_G = number of the element's nodes used to approximate geometry, typically 8 or 9. Note that due to Shell theory the shell thickness at node k ${}^{0}a_k = {}^{t}a_k = {}^{t+\Delta t}a_k^{(i-1)} = {}^{t+\Delta t}a_k^{(i)} = a_k$. The symbol ${}^{t+\Delta t}V_i^{n_k,(i)}$ is *i*th coordinate, ($i = 1,2,3$ for coordinate x, y, z), of the vector V^n at node k at time $t + \Delta t$, iteration (i). The vector V^n is normal to the shell. Later we will also use vectors V^1, V^2 , $(V^1 \perp V^2 \perp V^n)$. They will constitute base vectors for shell's bending rotations α, β .

Similarly, displacements at time $t + \Delta t$, iteration *(i-1)*:

$$
{}^{t+\Delta t}u_i^{(i-1)} = {}^{t+\Delta t}x_i^{(i-1)} - {}^{t}x_i \tag{3.105}
$$

Substituting (3.104) into (3.105)

$$
\begin{aligned}
{}^{t+\Delta t}u_i^{(i-1)} &= h_k\left(\left({}^{t+\Delta t}X_i^{k,(i-1)} + \frac{t}{2}a_k \, {}^{t+\Delta t}V_i^{n_k,(i-1)} \right) - \left({}^{t}X_i^k + \frac{t}{2}a_k {}^{t}V_i^{n_k} \right) \right) \\
&= h_k\left(\left({}^{t+\Delta t}X_i^{k,(i-1)} - {}^{t}X_i^k \right) + \frac{t}{2}a_k \left({}^{t+\Delta t}V_i^{n_k,(i-1)} - {}^{t}V_i^{n_k} \right) \right)
\end{aligned}
\tag{3.106}
$$

Note that in this case $k = 1...n$, n is number of nodes to approximate displacements. Current implementation of the shell elements assumes $n_g = n$, (which differs for Ahmads elements).

Displacement increments within an iteration (at time $t + \Delta t$) are:

$$
\begin{aligned}
u_i &= h_k\left(\left({}^{t+\Delta t}X_i^{k,(i)} - {}^{t+\Delta t}X_i^{k,(i-1)} \right) + \frac{t}{2}a_k \left({}^{t+\Delta t}V_i^{n_k,(i)} - {}^{t+\Delta t}V_i^{n_k,(i-1)} \right) \right) \\
&= h_k\left(U_i^k + \frac{t}{2}a_k \left({}^{t+\Delta t}V_i^{n_k,(i)} - {}^{t+\Delta t}V_i^{n_k,(i-1)} \right) \right)
\end{aligned}
\tag{3.107}
$$

At each node, the element has 5 DOFs: 3 displacements U_i^k and two rotations α^k, β^k described below:

Let us define at each node of the shell a local coordinate system specified by three vectors ${}^{t+\Delta t}V_i^{1_k,(i-1)}$, ${}^{t+\Delta t}V_i^{2_k,(i-1)}$, ${}^{t+\Delta t}V_i^{n_k,(i-1)}$, see Fig. 3-36. The last vector is vector normal to surface of the shell at node *k* and the first and second vectors are calculated as follows:

$$
\begin{aligned}
{}^{t+\Delta t}V_i^{1_k,(i-1)} &= \left(\overline{e}_2 \otimes {}^{t+\Delta t}V_i^{n_k,(i-1)} \right) / \left\| \overline{e}_2 \otimes {}^{t+\Delta t}V_i^{n_k,(i-1)} \right\| \\
{}^{t+\Delta t}V_i^{2_k,(i-1)} &= {}^{t+\Delta t}V_i^{n_k,(i-1)} \otimes {}^{t+\Delta t}V_i^{1_k,(i-1)}
\end{aligned}
\tag{3.108}
$$

For the next derivation let us assume a general vector $\overline{v}_L = \left[v_{1L}, v_{2L}, v_{3L}\right]^T$ with unit length that is subject to rotations

$[\alpha_L, \beta_L, \gamma_L]^T$, (where the subscript *L* indicates that both the vector and the rotations are defined with respect to the local coordinate system (defined by ${}^{t+\Delta t}V_i^{1_k,(i-1)}$, ${}^{t+\Delta t}V_i^{2_k,(i-1)}$, ${}^{t+\Delta t}V_i^{n_k,(i-1)}$). The rotations of the vector will produce displacements, (all in the local CS)

$$\begin{bmatrix} u_{1L} \\ u_{2L} \\ u_{3L} \end{bmatrix} = \begin{bmatrix} 0 & v_{3L} & -v_{2L} \\ -v_{3L} & 0 & v_{1L} \\ v_{2L} & -v_{1L} & 0 \end{bmatrix} \begin{bmatrix} \alpha_L \\ \beta_L \\ \gamma_L \end{bmatrix} = \begin{bmatrix} v_{3L}\beta_L - v_{2L}\gamma_L \\ -v_{3L}\alpha_L + v_{1L}\gamma_L \\ v_{2L}\alpha_L - v_{1L}\beta_L \end{bmatrix} \tag{3.109}$$

Transforming the displacements from local to global coordinate system

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \mathbf{T}_{L2G} \begin{bmatrix} u_{1L} \\ u_{2L} \\ u_{3L} \end{bmatrix} = \begin{bmatrix} {}^{t+\Delta t}V_1^{1_k,(i-1)} & {}^{t+\Delta t}V_1^{2_k,(i-1)} & {}^{t+\Delta t}V_1^{n_k,(i-1)} \\ {}^{t+\Delta t}V_2^{1_k,(i-1)} & {}^{t+\Delta t}V_2^{2_k,(i-1)} & {}^{t+\Delta t}V_2^{n_k,(i-1)} \\ {}^{t+\Delta t}V_3^{1_k,(i-1)} & {}^{t+\Delta t}V_3^{2_k,(i-1)} & {}^{t+\Delta t}V_3^{n_k,(i-1)} \end{bmatrix} \begin{bmatrix} u_{1L} \\ u_{2L} \\ u_{3L} \end{bmatrix} =$$

$$\begin{bmatrix} {}^{t+\Delta t}V_1^{1_k,(i-1)} & {}^{t+\Delta t}V_1^{2_k,(i-1)} & {}^{t+\Delta t}V_1^{n_k,(i-1)} \\ {}^{t+\Delta t}V_2^{1_k,(i-1)} & {}^{t+\Delta t}V_2^{2_k,(i-1)} & {}^{t+\Delta t}V_2^{n_k,(i-1)} \\ {}^{t+\Delta t}V_3^{1_k,(i-1)} & {}^{t+\Delta t}V_3^{2_k,(i-1)} & {}^{t+\Delta t}V_3^{n_k,(i-1)} \end{bmatrix} \begin{bmatrix} v_{3L}\beta_L - v_{2L}\gamma_L \\ -v_{3L}\alpha_L + v_{1L}\gamma_L \\ v_{2L}\alpha_L - v_{1L}\beta_L \end{bmatrix} = \tag{3.110}$$

$$\begin{bmatrix} {}^{t+\Delta t}V_1^{1_k,(i-1)}\left(v_{3L}\beta_L - v_{2L}\gamma_L\right) + {}^{t+\Delta t}V_1^{2_k,(i-1)}\left(-v_{3L}\alpha_L + v_{1L}\gamma_L\right) + {}^{t+\Delta t}V_1^{n_k,(i-1)}\left(v_{2L}\alpha_L - v_{1L}\beta_L\right) \\ {}^{t+\Delta t}V_2^{1_k,(i-1)}\left(v_{3L}\beta_L - v_{2L}\gamma_L\right) + {}^{t+\Delta t}V_2^{2_k,(i-1)}\left(-v_{3L}\alpha_L + v_{1L}\gamma_L\right) + {}^{t+\Delta t}V_2^{n_k,(i-1)}\left(v_{2L}\alpha_L - v_{1L}\beta_L\right) \\ {}^{t+\Delta t}V_3^{1_k,(i-1)}\left(v_{3L}\beta_L - v_{2L}\gamma_L\right) + {}^{t+\Delta t}V_3^{2_k,(i-1)}\left(-v_{3L}\alpha_L + v_{1L}\gamma_L\right) + {}^{t+\Delta t}V_3^{n_k,(i-1)}\left(v_{2L}\alpha_L - v_{1L}\beta_L\right) \end{bmatrix}$$

Now assume the same behavior for a vector normal to the shell's surface (again in the local CS and unit length), i.e. $\overline{v}_L = {}^{t+\Delta t}V_{iL}^{n_k,(i-1)} = \left[0,0,1\right]$. When this vector gets rotated, it produces displacements, (see (3.110):

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} {}^{t+\Delta t}V_1^{1_k,(i-1)}\left(\beta_L\right) + {}^{t+\Delta t}V_1^{2_k,(i-1)}\left(-\alpha_L\right) \\ {}^{t+\Delta t}V_2^{1_k,(i-1)}\left(\beta_L\right) + {}^{t+\Delta t}V_2^{2_k,(i-1)}\left(-\alpha_L\right) \\ {}^{t+\Delta t}V_3^{1_k,(i-1)}\left(\beta_L\right) + {}^{t+\Delta t}V_3^{2_k,(i-1)}\left(-\alpha_L\right) \end{bmatrix} \tag{3.111}$$

Substituting now $\alpha_L = {}^{t+\Delta t}\alpha^{k,(i-1)}$, $\beta_L = {}^{t+\Delta t}\beta^{k,(i-1)}$ and $u_k = {}^{t+\Delta t}V_i^{n_k,(i-1)}$, $k = 1..3$ we can write final equations for displacements due to rotations, (for iteration *(i-1)* and *(i)* and the difference):

$$
\begin{aligned}
{}^{t+\Delta t}V_i^{n_k,(i-1)} &= -{}^{t+\Delta t}\alpha^{k,(i-1)}\ {}^{t+\Delta t}V_i^{2_k,(i-1)} + {}^{t+\Delta t}\beta^{k,(i-1)}\ {}^{t+\Delta t}V_i^{1_k,(i-1)} \\
{}^{t+\Delta t}V_i^{n_k,(i)} &= -{}^{t+\Delta t}\alpha^{k,(i)}\ {}^{t+\Delta t}V_i^{2_k,(i-1)} + {}^{t+\Delta t}\beta^{k,(i)}\ {}^{t+\Delta t}V_i^{1_k,(i-1)}
\end{aligned}
$$

$$
\begin{aligned}
{}^{t+\Delta t}V_i^{n_k,(i)} - {}^{t+\Delta t}V_i^{n_k,(i-1)} &= -\left({}^{t+\Delta t}\alpha^{k,(i)} - {}^{t+\Delta t}\alpha^{k,(i-1)}\right){}^{t+\Delta t}V_i^{2_k,(i-1)} + \left({}^{t+\Delta t}\beta^{k,(i)} - {}^{t+\Delta t}\beta^{k,(i-1)}\right){}^{t+\Delta t}V_i^{1_k,(i-1)} \\
&= -\alpha^k\ {}^{t+\Delta t}V_i^{2_k,(i-1)} + \beta^k\ {}^{t+\Delta t}V_i^{1_k,(i-1)}
\end{aligned}
\tag{3.112}
$$

Hence, they represent rotation along two user defined vectors ${}^{t+\Delta t}V_i^{n_k,(i-1)}$. It is important to note that the vectors ${}^{t+\Delta t}V_i^{n_k,(i-1)}$ moves as the structure deforms.

Using (3.112) in (3.107) yields (and assuming shell thickness at a node k)

$$
u_i = h_k\left(U_i^k + \frac{t}{2}a_k\left(-\alpha^k\ {}^{t+\Delta t}V_i^{2_k,(i-1)} + \beta^k\ {}^{t+\Delta t}V_i^{1_k,(i-1)}\right)\right) \tag{3.113}
$$

Note that the vectors ${}^{t+\Delta t}V_i^{1_k,(i-1)}$, ${}^{t+\Delta t}V_i^{2_k,(i-1)}$, ${}^{t+\Delta t}V_i^{n_k,(i-1)}$ must be normalized. Also note that (3.111) should be used to connect dofs of shell and solid elements.

3.13.2 Connection of the shell2D to an ambient solid element

Connection of the shell2D element to an ambient structure consists of two part:

1. fix a FE node with $[u,v,w]$ displacement within the shell2D element,
2. fix two rotation dofs of the shell2D element within ambient elements.

3.13.2.1 Fixing a FE node with $[u,v,w]$ displacement within the shell2D element

Using the shell2D approximation the shell's displacement at the bottom u_i^{bot} and at the top u_i^{top} are:

$$
\begin{aligned}
u_i^{bot} &= h_k\left(U_i^k - \frac{a_k}{2}\left(-\alpha^k\ {}^{t+\Delta t}V_i^{2_k,(i-1)} + \beta^k\ {}^{t+\Delta t}V_i^{1_k,(i-1)}\right)\right) \\
u_i^{top} &= h_k\left(U_i^k + \frac{a_k}{2}\left(-\alpha^k\ {}^{t+\Delta t}V_i^{2_k,(i-1)} + \beta^k\ {}^{t+\Delta t}V_i^{1_k,(i-1)}\right)\right) \\
u_i^{top-bot} &= h_k\, a_k\left(-\alpha^k\ {}^{t+\Delta t}V_i^{2_k,(i-1)} + \beta^k\ {}^{t+\Delta t}V_i^{1_k,(i-1)}\right)
\end{aligned}
\tag{3.114}
$$

The index i is 1..3 for x..z displacements. Using the shell3D approximation displacement at the same locations can be calculated by:

$$
\begin{aligned}
uu_i^{bot} &= hh_l^{bot}\ UU_i^{bot,l} \\
uu_i^{top} &= hh_l^{top}\ UU_i^{top,l} \\
uu_i^{top-bot} &= uu_i^{top} - uu_i^{bot}
\end{aligned}
\tag{3.115}
$$

where hh_l^{bot} and hh_l^{top} are the solid's shell3D interpolation functions at location top and bottom of the shells at node i, $UU_i^{bot,l}, UU_i^{top,l}$ are corresponding nodal displacements of the solid element. Comparing (3.115) and (3.114) it can be shown that

$$
\begin{aligned}
h_k &= hh_k^{top} + hh_k^{bot} \\
t\,h_k &= hh_k^{top} + hh_k^{bot}
\end{aligned}
\tag{3.116}
$$

Thus, to fix $[u,v,w]$ doffs of a node with shell2D elements we first calculate hh_i values for the case of shell3D approximation. Then, these are used to get h_i , (see (3.115), comprised in the shell2D approximation. It remains to compute shell2D rotation α_i, β_i and this is (again) done by comparing 2D and 3D approximation in (3.114) and (3.115). After some mathematical manipulation we will arrive to the final expressions:

$$
\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \sum_k \begin{bmatrix}
hh_k^{top} + hh_k^{bot} & 0 & 0 & -\frac{a_k}{2} V_x^{2_k} \left(hh_k^{top} - hh_k^{bot} \right) & \frac{a_k}{2} V_x^{1_k} \left(hh_k^{top} - hh_k^{bot} \right) \\
0 & hh_k^{top} + hh_k^{bot} & 0 & -\frac{a_k}{2} V_y^{2_k} \left(hh_k^{top} - hh_k^{bot} \right) & \frac{a_k}{2} V_y^{1_k} \left(hh_k^{top} - hh_k^{bot} \right) \\
0 & 0 & hh_k^{top} + hh_k^{bot} & -\frac{a_k}{2} V_z^{2_k} \left(hh_k^{top} - hh_k^{bot} \right) & \frac{a_k}{2} V_z^{1_k} \left(hh_k^{top} - hh_k^{bot} \right)
\end{bmatrix} \begin{bmatrix} u_k \\ v_k \\ w_k \\ \alpha_k \\ \beta_k \end{bmatrix}
\tag{3.117}
$$

3.13.2.2 Fixing two rotation dofs of the shell2D element within ambient elements

Derivation of expressions to fix shell2D rotations in ambient elements is based (similarly to the previous section) on comparing the shell2D and shell3D approximation of top and bottom nodes. What we do is we first we fix the top and bottom in the ambient element using (solid) 3D approximation. It yields expression something like:

$$
\begin{aligned}
uu_i^{top} &= hh_l^{top}\ UU_i^{top,l} + ... \\
uu_i^{bot} &= hh_l^{top}\ UU_i^{bot,l} + ...
\end{aligned}
\tag{3.118}
$$

Note that rhs of (3.118) may also include rotations. The resulting equations for shell2D rotation α, β are:

$$D = -a_k(-V1_{k,x}{}^2 V2_{k,x} V2_{k,y} + V1_{k,x}{}^2 V2_{k,y}{}^2 + V1_{k,x}{}^2 V2_{k,z}{}^2 + V1_{k,x} V1_{k,y} V2_{k,x}{}^2 - V1_{k,x} V1_{k,y} V2_{k,x} V2_{k,y} + V1_{k,x} V1_{k,y} V2_{k,z}{}^2 - 2V1_{k,x} V1_{k,z} V2_{k,x} V2_{k,z} - V1_{k,x} V1_{k,z} V2_{k,y} V2_{k,z} - V1_{k,y} V1_{k,z} V2_{k,y} V2_{k,z} + V1_{k,z}{}^2 V2_{k,x}{}^2 + V1_{k,z}{}^2 V2_{k,y}{}^2)$$

$$\alpha cf_{k,1}^{top} = (-V1_{k,x}{}^2 V2_{k,y} + V1_{k,x} V1_{k,y} V2_{k,x} - V1_{k,x} V1_{k,z} V2_{k,z} + V1_{k,z}{}^2 V2_{k,x}) / D$$
$$\alpha cf_{k,2}^{top} = (V1_{k,x}{}^2 V2_{k,y} - V1_{k,x} V1_{k,y} V2_{k,x} - V1_{k,y} V1_{k,z} V2_{k,z} + V1_{k,z}{}^2 V2_{k,y}) / D$$
$$\alpha cf_{k,3}^{top} = (V1_{k,x}{}^2 V2_{k,z} + V1_{k,x} V1_{k,y} V2_{k,z} - V1_{k,x} V1_{k,z} V2_{k,x} - V1_{k,x} V1_{k,z} V2_{k,y}) / D$$
$$\alpha cf_{k,1}^{bot} = (V1_{k,x}{}^2 V2_{k,y} - V1_{k,x} V1_{k,y} V2_{k,x} + V1_{k,x} V1_{k,z} V2_{k,z} - V1_{k,z}{}^2 V2_{k,x}) / D$$
$$\alpha cf_{k,2}^{bot} = (-V1_{k,x}{}^2 V2_{k,y} + V1_{k,x} V1_{k,y} V2_{k,x} + V1_{k,y} V1_{k,z} V2_{k,z} - V1_{k,z}{}^2 V2_{k,y}) / D$$
$$\alpha cf_{k,3}^{bot} = (-V1_{k,x}{}^2 V2_{k,z} - V1_{k,x} V1_{k,y} V2_{k,z} + V1_{k,x} V1_{k,z} V2_{k,x} + V1_{k,x} V1_{k,z} V2_{k,y}) / D$$

$$\beta cf_{k,1}^{top} = (-V1_{k,x} V2_{k,y}{}^2 - V1_{k,x} V2_{k,z}{}^2 + V1_{k,y} V2_{k,x} V2_{k,y} + V1_{k,z} V2_{k,x} V2_{k,z} / D$$
$$\beta cf_{k,2}^{top} = (V1_{k,x} V2_{k,x} V2_{k,y} - V1_{k,y} V2_{k,x}{}^2 - V1_{k,y} V2_{k,z}{}^2 + V1_{k,z} V2_{k,y} V2_{k,z} / D$$
$$\beta cf_{k,3}^{top} = (V1_{k,x} V2_{k,x} V2_{k,z} + V1_{k,y} V2_{k,y} V2_{k,z} - V1_{k,z} V2_{k,x}{}^2 - V1_{k,z} V2_{k,y}{}^2 / D$$
$$\beta cf_{k,1}^{bot} = (V1_{k,x} V2_{k,y}{}^2 + V1_{k,x} V2_{k,z}{}^2 - V1_{k,y} V2_{k,x} V2_{k,y} - V1_{k,z} V2_{k,x} V2_{k,z} / D$$
$$\beta cf_{k,2}^{bot} = (-V1_{k,x} V2_{k,x} V2_{k,y} + V1_{k,y} V2_{k,x}{}^2 + V1_{k,y} V2_{k,z}{}^2 - V1_{k,z} V2_{k,y} V2_{k,z} / D$$
$$\beta cf_{k,3}^{bot} = (-V1_{k,x} V2_{k,x} V2_{k,z} - V1_{k,y} V2_{k,y} V2_{k,z} + V1_{k,z} V2_{k,x}{}^2 + V1_{k,z} V2_{k,y}{}^2 / D \quad (3.119)$$

$$\alpha_k = \alpha cf_{l,i}^{top}(hh_l^{top}\ UU_i^{top,l} + ...) + \alpha cf_{l,i}^{bot}(hh_k^{bot}\ UU_i^{bot,l} + ...)$$
$$\beta_k = \beta cf_{l,i}^{top}(hh_l^{top}\ UU_i^{top,l} + ...) + \beta cf_{l,i}^{bot}(hh_k^{bot}\ UU_i^{bot,l} + ...) \quad (3.120)$$

$k = 1..number\ of\ approximation\ shell\ 2D\ nodes$
$i = 1..3, (= x..z)$
$l = 1..number\ of\ approximation\ solid\ 3D\ nodes$

where $V1_{k,i} = V_i^{1_k} \quad V2_{k,i} = V_i^{2_k}$.

Note that displacement dofs are fixed by (3.117).

If either bottom or top node gets outside the ambient element, the middle point is used instead. Equation (3.120) is still valid but it is necessary to use $\widehat{D} = \frac{1}{2} D$ that replaces D to calculate the $\alpha cf_{k,1}^{top} ... \beta cf_{k,3}^{bot}$ coefficients.

3.13.3 Green-Lagrange strains

The elements are derived using Green-Lagrange strains and 2nd Piola Kirchhoff stresses. Green-Lagrange strains at (i-th iteration), i,j-axis x,y,z are calculated as follows :

$$
\begin{aligned}
{}^{t+\Delta t^{(i)}}_{\ t}\varepsilon_{ij} &= \frac{1}{2}\left({}^{t+\Delta t^{(i)}}_{\ t}u^{(i)}_{i,j} + {}^{t+\Delta t^{(i)}}_{\ t}u^{(i)}_{j,i} + {}^{t+\Delta t}_{\ t}u^{(i)}_{m,i}\ {}^{t+\Delta t}_{\ t}u^{(i)}_{m,j} \right) \\
&= \frac{1}{2}\left(({}^{t+\Delta t}_{\ t}u^{(i-1)}_{i,j} + {}_{t}u_{i,j}) + ({}^{t+\Delta t}_{\ t}u^{(i-1)}_{j,i} + {}_{t}u_{j,i}) + ({}^{t+\Delta t}_{\ t}u^{(i-1)}_{m,i} + {}_{t}u_{m,i})({}^{t+\Delta t}_{\ t}u^{(i-1)}_{m,j} + {}_{t}u_{m,j}) \right) \\
&= {}^{t+\Delta t^{(i-1)}}_{\ t}\varepsilon_{ij} + {}_{t}e_{ij} + {}_{t}\eta_{ij}
\end{aligned}
\tag{3.121}
$$

where:

$$
\begin{aligned}
{}^{t+\Delta t}_{\ t}\varepsilon^{(i-1)}_{ij} &= \frac{1}{2}\left({}^{t+\Delta t}_{\ t}u^{(i-1)}_{i,j} + {}^{t+\Delta t}_{\ t}u^{(i-1)}_{j,i} + {}^{t+\Delta t}_{\ t}u^{(i-1)}_{k,i}\ {}^{t+\Delta t}_{\ t}u^{(i-1)}_{k,j} \right) \\
{}_{t}e_{ij} &= \frac{1}{2}\left({}_{t}u_{i,j} + {}_{t}u_{j,i} + {}^{t+\Delta t}_{\ t}u^{(i-1)}_{m,i}\ {}_{t}u_{m,j} + {}^{t+\Delta t}_{\ t}u^{(i-1)}_{m,j}\ {}_{t}u_{m,i} \right) \\
&= {}_{t}e^{0}_{ij} + {}_{t}e^{1}_{ij} \\
{}_{t}e^{0}_{ij} &= \frac{1}{2}\left({}_{t}u_{i,j} + {}_{t}u_{j,i} \right) \\
{}_{t}e^{1}_{ij} &= \frac{1}{2}\left({}^{t+\Delta t}_{\ t}u^{(i-1)}_{m,i}\ {}_{t}u_{m,j} + {}^{t+\Delta t}_{\ t}u^{(i-1)}_{m,j}\ {}_{t}u_{m,i} \right) \\
{}_{t}\eta_{ij} &= \frac{1}{2}\left({}_{t}u_{m,i}\ {}_{t}u_{m,j} \right)
\end{aligned}
\tag{3.122}
$$

Element's displacements u are approximated by isoparametric interpolation. Hence, it is simple to calculate their derivatives with respect to local coordinate r,s,t. Using an arbitrary function $f(x,y,z)$ Eqn. (3.123) to (3.125) show, how to compute its derivatives with respect to global x,y,z axis.

Calculation of derivatives:

$$
\begin{bmatrix} \frac{\partial f}{\partial r} \\ \frac{\partial f}{\partial s} \\ \frac{\partial f}{\partial t} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial y}{\partial r} & \frac{\partial z}{\partial r} \\ \frac{\partial x}{\partial s} & \frac{\partial y}{\partial s} & \frac{\partial z}{\partial s} \\ \frac{\partial x}{\partial t} & \frac{\partial y}{\partial t} & \frac{\partial z}{\partial t} \end{bmatrix} \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix} = \mathbf{J} \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix} \tag{3.123}
$$

$$
\begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \frac{\partial f}{\partial r} \\ \frac{\partial f}{\partial s} \\ \frac{\partial f}{\partial t} \end{bmatrix} \tag{3.124}
$$

Derivatives of coordinates at t with respect to r,s,t to calculate **J**:

$$\begin{aligned}
\frac{\partial {}^{t}x_i}{\partial r} &= \frac{\partial h_k}{\partial r}\left({}^{t}X_i^k + \frac{t}{2}\,{}^{t}a_i\,{}^{t}V_{n_i}^k \right) \\
\frac{\partial {}^{t}x_i}{\partial s} &= \frac{\partial h_i}{\partial s}\left({}^{t}X_i^k + \frac{t}{2}\,{}^{t}a_i\,{}^{t}V_{n_i}^k \right) \\
\frac{\partial {}^{t}x_i}{\partial t} &= h_i\left(\frac{1}{2}\,{}^{t}a_i\,{}^{t}V_{n_i}^k \right)
\end{aligned} \tag{3.125}$$

Derivatives of displacement increments at time $\left\langle t \ldots t+\Delta t^{(i-1)} \right\rangle$ with respect to r,s,t:

$$\begin{aligned}
\frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial r} &= \frac{\partial h_k}{\partial r}\left(\left({}^{t+\Delta t}X_i^{k,(i-1)} - {}^{t}X_i^k \right) + \frac{t}{2} a_k \left({}^{t+\Delta t}V_i^{n_k,(i-1)} - {}^{t}V_i^{n_k} \right) \right) \\
\frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial s} &= \frac{\partial h_k}{\partial s}\left(\left({}^{t+\Delta t}X_i^{k,(i-1)} - {}^{t}X_i^k \right) + \frac{t}{2} a_k \left({}^{t+\Delta t}V_i^{n_k,(i-1)} - {}^{t}V_i^{n_k} \right) \right) \\
\frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial t} &= h_k\left(\frac{1}{2} a_k \left({}^{t+\Delta t}V_i^{n_k,(i-1)} - {}^{t}V_i^{n_k} \right) \right)
\end{aligned} \tag{3.126}$$

$$\begin{aligned}
\frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial r} &= \frac{\partial h_k}{\partial r}\left({}^{t+\Delta t}U_i^{k,(i-1)} + \frac{t}{2} a_k \; {}^{t+\Delta t}dV_i^{n_k,(i-1)} \right) \\
\frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial s} &= \frac{\partial h_k}{\partial s}\left({}^{t+\Delta t}U_i^{k,(i-1)} + \frac{t}{2} a_k \; {}^{t+\Delta t}dV_i^{n_k,(i-1)} \right) \\
\frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial t} &= h_k\left(\frac{1}{2} a_k \; {}^{t+\Delta t}dV_i^{n_k,(i-1)} \right)
\end{aligned} \tag{3.127}$$

$$\begin{aligned}
{}^{t+\Delta t}U_i^{k,(i-1)} &= {}^{t+\Delta t}X_i^{k,(i-1)} - {}^{t}X_i^k \\
{}^{t+\Delta t}dV_i^{n_k,(i-1)} &= {}^{t+\Delta t}V_i^{n_k,(i-1)} - {}^{t}V_i^{n_k}
\end{aligned}$$

Derivatives of displacement increments at time $t+\Delta t$ within iteration i with respect to r,s,t:

$$\begin{aligned}
\frac{\partial u_i}{\partial r} &= \frac{\partial h_k}{\partial r}\left(U_i^k + \frac{t}{2} a_k \left(-\alpha^k \; {}^{t+\Delta t}V_i^{2_k,(i-1)} + \beta^k \; {}^{t+\Delta t}V_i^{1_k,(i-1)} \right) \right) \\
\frac{\partial u_i}{\partial s} &= \frac{\partial h_k}{\partial s}\left(U_i^k + \frac{t}{2} a_k \left(-\alpha^k \; {}^{t+\Delta t}V_i^{2_k,(i-1)} + \beta^k \; {}^{t+\Delta t}V_i^{1_k,(i-1)} \right) \right) \\
\frac{\partial u_i}{\partial t} &= h_k\left(\frac{1}{2} a_k \left(-\alpha^k \; {}^{t+\Delta t}V_i^{2_k,(i-1)} + \beta^k \; {}^{t+\Delta t}V_i^{1_k,(i-1)} \right) \right)
\end{aligned} \tag{3.128}$$

$$U_i^k = {}^{t+\Delta t}U_i^{k,(i)} - {}^{t+\Delta t}U_i^{k,(i-1)}$$

To proceed further in the derivation of the 3D isoparametric element, we need to calculate derivatives of the displacement increments with respect to ${}^t\overline{x} = \left({}^t x_1, {}^t x_2, {}^t x_3\right)$. This is achieved using (3.124) thru (3.128).

Derivatives of displacement increments at time $\left\langle t \ldots t + \Delta t^{(i-1)} \right\rangle$ with respect x_1, x_2, x_3:

$$\frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial\, {}^t x_j} = {}^t J_{j1}^{inv,k} \frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial r} + {}^t J_{j2}^{inv,k} \frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial s} + {}^t J_{j3}^{inv,k} \frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial t} \tag{3.129}$$

$$\begin{aligned}
\frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial\, {}^t x_j} &= {}^t J_{j1}^{inv,k} \frac{\partial h_k}{\partial r}\left({}^{t+\Delta t}U_i^{k,(i-1)} + \frac{t}{2} a_k \; {}^{t+\Delta t} dV_i^{n_k,(i-1)} \right) \\
&\quad + {}^t J_{j2}^{inv,k} \frac{\partial h_k}{\partial s}\left({}^{t+\Delta t}U_i^{k,(i-1)} + \frac{t}{2} a_k \; {}^{t+\Delta t} dV_i^{n_k,(i-1)} \right) \\
&\quad + {}^t J_{j3}^{inv,k} h_k \left(\frac{1}{2} a_k \; {}^{t+\Delta t} dV_i^{n_k,(i-1)} \right) \\
&= {}^{t+\Delta t}U_i^{k,(i-1)} \left(\frac{\partial h_k}{\partial r} \, {}^t J_{j1}^{inv} + \frac{\partial h_k}{\partial s} \, {}^t J_{j2}^{inv} \right) \\
&\quad + {}^{t+\Delta t} dV_i^{n_k,(i-1)} \frac{a_k}{2} \left(t\left(\frac{\partial h_k}{\partial r} \, {}^t J_{j1}^{inv} + \frac{\partial h_k}{\partial s} \, {}^t J_{j2}^{inv} \right) + {}^t J_{j3}^{inv} h_k \right) \\
&= {}^{t+\Delta t}U_i^{k,(i-1)} \, {}^t h_j^k + {}^{t+\Delta t} dV_i^{n_k,(i-1)} \frac{a_k}{2}{}^t G_j^k
\end{aligned} \tag{3.130}$$

$$\begin{aligned}
{}^t h_j^k &= \frac{\partial h_k}{\partial r} \, {}^t J_{j1}^{inv,k} + \frac{\partial h_k}{\partial s} \, {}^t J_{j2}^{inv,k} \\
{}^t G_j^k &= \left(t \, {}^t h_j^k + {}^t J_{j3}^{inv,k} \, h_k \right)
\end{aligned}$$

Derivatives of displacement increments at time $t + \Delta t$ within iteration i with respect to x_1, x_2, x_3:

$$\frac{\partial u_i}{\partial\, {}^t x_j} = {}^t J_{j1}^{inv,k} \frac{\partial u_i}{\partial r} + {}^t J_{j2}^{inv,k} \frac{\partial u_i}{\partial s} + {}^t J_{j3}^{inv,k} \frac{\partial u_i}{\partial t} \tag{3.131}$$

$$\begin{aligned}
\frac{\partial u_i}{\partial\, {}^t x_j} &= {}^t J_{j1}^{inv,k} \frac{\partial h_k}{\partial r}\left(U_i^k + \frac{t}{2} a_k \left(-\alpha^k \; {}^{t+\Delta t} V_i^{2_k,(i-1)} + \beta^k \; {}^{t+\Delta t} V_i^{1_k,(i-1)} \right) \right) \\
&\quad + {}^t J_{j2}^{inv,k} \frac{\partial h_k}{\partial s}\left(U_i^k + \frac{t}{2} a_k \left(-\alpha^k \; {}^{t+\Delta t} V_i^{2_k,(i-1)} + \beta^k \; {}^{t+\Delta t} V_i^{1_k,(i-1)} \right) \right) \\
&\quad + {}^t J_{j3}^{inv,k} \, h_k \left(\frac{1}{2} a_k \left(-\alpha^k \; {}^{t+\Delta t} V_i^{2_k,(i-1)} + \beta^k \; {}^{t+\Delta t} V_i^{1_k,(i-1)} \right) \right)
\end{aligned} \tag{3.132}$$

After some rearrangement Eqn. (3.155) yields:

$$\frac{\partial u_i}{\partial {}^t x_j} = U_i^k \left(\frac{\partial h_k}{\partial r} {}^t J_{j1}^{inv,k} + \frac{\partial h_k}{\partial s} {}^t J_{j2}^{inv,k} \right)$$
$$- \alpha^k \frac{a_k}{2} {}^{t+\Delta t} V_i^{2_k,(i-1)} \left(t \left(\frac{\partial h_k}{\partial r} {}^t J_{j1}^{inv,k} + \frac{\partial h_k}{\partial s} {}^t J_{j2}^{inv,k} \right) + {}^t J_{j3}^{inv,k} h_k \right)$$
$$+ \beta^k \frac{a_k}{2} {}^{t+\Delta t} V_i^{1_k,(i-1)} \left(t \left(\frac{\partial h_k}{\partial r} {}^t J_{j1}^{inv,k} + \frac{\partial h_k}{\partial s} {}^t J_{j2}^{inv,k} \right) + {}^t J_{j3}^{inv,k} h_k \right)$$
$$= U_i^k {}^t h_j^k + \alpha^k {}^{t+\Delta t} g_i^{1_k,(i-1)} {}^t G_j^k + \beta^k {}^{t+\Delta t} g_i^{2_k,(i-1)} {}^t G_j^k \tag{3.133}$$

$${}^{t+\Delta t} g_i^{1_k,(i-1)} = -\frac{a_k}{2} {}^{t+\Delta t} V_i^{2_k,(i-1)}$$
$${}^{t+\Delta t} g_i^{2_k,(i-1)} = \frac{a_k}{2} {}^{t+\Delta t} V_i^{1_k,(i-1)}$$

At this place, we can derive final expression to compute linear and nonlinear strains increments. Linear strains ${}_t e_{ij}^{(i)}$ are calculated as follows:

$${}_t \overline{e}^{(i)} = \begin{bmatrix} {}_t e_{11}^{(i)} \\ {}_t e_{22}^{(i)} \\ {}_t e_{33}^{(i)} \\ 2 {}_t e_{12}^{(i)} \\ 2 {}_t e_{23}^{(i)} \\ 2 {}_t e_{13}^{(i)} \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1^{L_0} + \mathbf{B}_1^{L_1} & \dots & \mathbf{B}_k^{L_0} + \mathbf{B}_k^{L_1} & \dots & \mathbf{B}_n^{L_0} + \mathbf{B}_n^{L_1} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{u}}_1^{(i)} \\ \dots \\ \widehat{\mathbf{u}}_k^{(i)} \\ \dots \\ \widehat{\mathbf{u}}_n^{(i)} \end{bmatrix} \tag{3.134}$$

$$\mathbf{B}_k^{L_0} = \begin{bmatrix} {}^t h_1^k & 0 & 0 & {}^{t+\Delta t} g_1^{1_k,(i-1)} {}^t G_1^k & {}^{t+\Delta t} g_1^{2_k,(i-1)} {}^t G_1^k \\ 0 & {}^t h_2^k & 0 & {}^{t+\Delta t} g_2^{1_k,(i-1)} {}^t G_2^k & {}^{t+\Delta t} g_2^{2_k,(i-1)} {}^t G_2^k \\ 0 & 0 & {}^t h_3^k & {}^{t+\Delta t} g_3^{1_k,(i-1)} {}^t G_3^k & {}^{t+\Delta t} g_3^{2_k,(i-1)} {}^t G_3^k \\ {}^t h_2^k & {}^t h_1^k & 0 & {}^{t+\Delta t} g_1^{1_k,(i-1)} {}^t G_2^k + {}^{t+\Delta t} g_2^{1_k,(i-1)} {}^t G_1^k & {}^{t+\Delta t} g_1^{2_k,(i-1)} {}^t G_2^k + {}^{t+\Delta t} g_2^{2_k,(i-1)} {}^t G_1^k \\ 0 & {}^t h_3^k & {}^t h_2^k & {}^{t+\Delta t} g_2^{1_k,(i-1)} {}^t G_3^k + {}^{t+\Delta t} g_3^{1_k,(i-1)} {}^t G_2^k & {}^{t+\Delta t} g_2^{2_k,(i-1)} {}^t G_3^k + {}^{t+\Delta t} g_3^{2_k,(i-1)} {}^t G_2^k \\ {}^t h_3^k & 0 & {}^t h_1^k & {}^{t+\Delta t} g_1^{1_k,(i-1)} {}^t G_3^k + {}^{t+\Delta t} g_3^{1_k,(i-1)} {}^t G_1^k & {}^{t+\Delta t} g_1^{2_k,(i-1)} {}^t G_3^k + {}^{t+\Delta t} g_3^{2_k,(i-1)} {}^t G_1^k \end{bmatrix} \tag{3.135}$$

where

$\widehat{\mathbf{u}}_k^{(i)} = \left[U_1^{k,(i)}, U_2^{k,(i)}, U_3^{k,(i)}, \alpha^{k,(i)}, \beta^{k,(i)} \right]^T$ at node *k*.

The second part of $\mathbf{B}_k^L$, i.e. $\mathbf{B}_k^{L_1}$, is derived from ${}_2 e_{ij}^1 = \mathbf{B}_k^{L_1} \widehat{\mathbf{u}}_k^{(i)}$, at node *k*:

$$
{}_t e_{ij}^1 = \frac{1}{2}\left({}^{t+\Delta t}_{\ t}u_{m,i}^{(i-1)}\ {}_t u_{m,j} + {}^{t+\Delta t}_{\ t}u_{m,j}^{(i-1)}\ {}_t u_{m,i} \right)
$$

$$
= \frac{1}{2}\ {}^{t+\Delta t}_{\ t}u_{m,i}^{(i-1)} \left(U_m^k\ {}^t h_j^k + \alpha^k\ {}^{t+\Delta t} g_m^{1_k,(i-1)}\ {}^t G_j^k + \beta^k\ {}^{t+\Delta t} g_m^{2_k,(i-1)}\ {}^t G_j^k \right)
$$

$$
+ \frac{1}{2}\ {}^{t+\Delta t}_{\ t}u_{m,j}^{(i-1)} \left(U_m^k\ {}^t h_i^k + \alpha^k\ {}^{t+\Delta t} g_m^{1_k,(i-1)}\ {}^t G_i^k + \beta^k\ {}^{t+\Delta t} g_m^{2_k,(i-1)}\ {}^t G_i^k \right)
$$

$$
= \frac{1}{2}\left(U_m^k\ {}^{t+\Delta t}_{\ t}u_{m,i}^{(i-1)}\ {}^t h_j^k + \alpha^k\ {}^{t+\Delta t}_{\ t}u_{m,i}^{(i-1)}\ {}^{t+\Delta t} g_m^{1_k,(i-1)}\ {}^t G_j^k + \beta^k\ {}^{t+\Delta t}_{\ t}u_{m,i}^{(i-1)}\ {}^{t+\Delta t} g_m^{2_k,(i-1)}\ {}^t G_j^k \right)
$$

$$
+ \frac{1}{2}\left(U_m^k\ {}^{t+\Delta t}_{\ t}u_{m,j}^{(i-1)}\ {}^t h_i^k + \alpha^k\ {}^{t+\Delta t}_{\ t}u_{m,j}^{(i-1)}\ {}^{t+\Delta t} g_m^{1_k,(i-1)}\ {}^t G_i^k + \beta^k\ {}^{t+\Delta t}_{\ t}u_{m,j}^{(i-1)}\ {}^{t+\Delta t} g_m^{2_k,(i-1)}\ {}^t G_i^k \right)
$$

Introducing

$$
\begin{aligned}
l_{mj}^{(i-1)} &= \frac{\partial^{t+\Delta t} u_m^{(i-1)}}{\partial^t x_j} = {}^{t+\Delta t}_{\ t}u_{m,j}^{(i-1)} \\
\Phi_{1j}^{k,(i-1)} &= \sum_m {}^{t+\Delta t} g_m^{1_k,(i-1)}\ l_{mj}^{(i-1)} \\
\Phi_{2j}^{k,(i-1)} &= \sum_m {}^{t+\Delta t} g_m^{2_k,(i-1)}\ l_{mj}^{(i-1)}
\end{aligned}
\tag{3.136}
$$

we can write

$$
\begin{aligned}
{}_t e_{ij}^1 &= \frac{1}{2}\left(U_m^k\ l_{mi}^{(i-1)}\ {}^t h_j^k + \alpha^k\ \Phi_{1i}^{k,(i-1)}\ {}^t G_j^k + \beta^k\ \Phi_{2i}^{k,(i-1)}\ {}^t G_j^k \right) \\
&+ \frac{1}{2}\left(U_m^k\ l_{mj}^{(i-1)}\ {}^t h_i^k + \alpha^k\ \Phi_{1j}^{k,(i-1)}\ {}^t G_i^k + \beta^k\ \Phi_{2j}^{k,(i-1)}\ {}^t G_i^k \right)
\end{aligned}
\tag{3.137}
$$

$$
\mathbf{B}_k^{L_1} = \begin{bmatrix}
l_{11}^{(i-1)}\ {}^t h_1^k & l_{21}^{(i-1)}\ {}^t h_1^k & l_{31}^{(i-1)}\ {}^t h_1^k \\
l_{12}^{(i-1)}\ {}^t h_2^k & l_{22}^{(i-1)}\ {}^t h_2^k & l_{32}^{(i-1)}\ {}^t h_2^k \\
l_{13}^{(i-1)}\ {}^t h_3^k & l_{23}^{(i-1)}\ {}^t h_3^k & l_{33}^{(i-1)}\ {}^t h_3^k \\
l_{11}^{(i-1)}\ {}^t h_2^k + l_{12}^{(i-1)}\ {}^t h_1^k & l_{21}^{(i-1)}\ {}^t h_2^k + l_{22}^{(i-1)}\ {}^t h_1^k & l_{31}^{(i-1)}\ {}^t h_2^k + l_{32}^{(i-1)}\ {}^t h_1^k \\
l_{12}^{(i-1)}\ {}^t h_3^k + l_{13}^{(i-1)}\ {}^t h_2^k & l_{22}^{(i-1)}\ {}^t h_3^k + l_{23}^{(i-1)}\ {}^t h_2^k & l_{32}^{(i-1)}\ {}^t h_3^k + l_{33}^{(i-1)}\ {}^t h_2^k \\
l_{11}^{(i-1)}\ {}^t h_3^k + l_{13}^{(i-1)}\ {}^t h_1^k & l_{21}^{(i-1)}\ {}^t h_3^k + l_{23}^{(i-1)}\ {}^t h_1^k & l_{31}^{(i-1)}\ {}^t h_3^k + l_{33}^{(i-1)}\ {}^t h_1^k
\end{bmatrix}
$$

$$
\begin{bmatrix}
\ldots & \Phi_{11}^{k,(i-1)}\ {}^t G_1^k & \Phi_{21}^{k,(i-1)}\ {}^t G_1^k \\
\ldots & \Phi_{12}^{k,(i-1)}\ {}^t G_2^k & \Phi_{22}^{k,(i-1)}\ {}^t G_2^k \\
\ldots & \Phi_{13}^{k,(i-1)}\ {}^t G_3^k & \Phi_{23}^{k,(i-1)}\ {}^t G_3^k \\
\ldots & \Phi_{11}^{k,(i-1)}\ {}^t G_2^k + \Phi_{12}^{k,(i-1)}\ {}^t G_1^k & \Phi_{21}^{k,(i-1)}\ {}^t G_2^k + \Phi_{22}^{k,(i-1)}\ {}^t G_1^k \\
\ldots & \Phi_{12}^{k,(i-1)}\ {}^t G_3^k + \Phi_{13}^{k,(i-1)}\ {}^t G_2^k & \Phi_{22}^{k,(i-1)}\ {}^t G_3^k + \Phi_{23}^{k,(i-1)}\ {}^t G_2^k \\
\ldots & \Phi_{11}^{k,(i-1)}\ {}^t G_3^k + \Phi_{13}^{k,(i-1)}\ {}^t G_1^k & \Phi_{21}^{k,(i-1)}\ {}^t G_3^k + \Phi_{23}^{k,(i-1)}\ {}^t G_1^k
\end{bmatrix}
\tag{3.138}
$$

The energy of nonlinear strains:

Let ${}^{t+\Delta t}_{t}S^{(i-1)}$ is a matrix storing stresses s_{ij} at time $t+\Delta t$, iteration (*i-1*):

$$ {}^{t+\Delta t}_{t}S^{(i-1)} = \begin{bmatrix} S_{11} & & & & & & & & \\ 0 & S_{11} & & & & & & & \\ 0 & 0 & S_{11} & & & & SYM & & \\ S_{12} & 0 & 0 & S_{22} & & & & & \\ 0 & S_{12} & 0 & 0 & S_{22} & & & & \\ 0 & 0 & S_{12} & 0 & 0 & S_{22} & & & \\ S_{13} & 0 & 0 & S_{23} & 0 & 0 & S_{33} & & \\ 0 & S_{13} & 0 & 0 & S_{23} & 0 & 0 & S_{33} & \\ 0 & 0 & S_{13} & 0 & 0 & S_{23} & 0 & 0 & S_{33} \end{bmatrix} \tag{3.139} $$

Then matrix $\mathbf{B}^{NL} = \left[\mathbf{B}_1^{NL} \ldots \mathbf{B}_k^{NL} \ldots \mathbf{B}_n^{NL} \right]$ is composed so that (at a node *k*)

$$ \delta\eta_{ij} \; s_{ij} \; \eta_{ij} = \left(\delta\hat{\mathbf{u}}_k^{(i)} \right)^T \left(\mathbf{B}_k^{NL} \right)^T \; {}^{t+\Delta t}_{t}S^{(i-1)} \; \mathbf{B}_k^{NL} \; \hat{\mathbf{u}}_k^{(i)} \tag{3.140} $$

where δ states for variation of the following entity. It can be shown that the matrix $\mathbf{B}^{NL}$ can be set in the following shape:

$$ \mathbf{B}_k^{NL} = \begin{bmatrix} {}^{t}h_1^k & 0 & 0 & {}^{t+\Delta t}g_1^{1_k,(i-1)} \, {}^{t}G_1^k & {}^{t+\Delta t}g_1^{2_k,(i-1)} \, {}^{t}G_1^k \\ 0 & {}^{t}h_1^k & 0 & {}^{t+\Delta t}g_2^{1_k,(i-1)} \, {}^{t}G_1^k & {}^{t+\Delta t}g_2^{2_k,(i-1)} \, {}^{t}G_1^k \\ 0 & 0 & {}^{t}h_1^k & {}^{t+\Delta t}g_3^{1_k,(i-1)} \, {}^{t}G_1^k & {}^{t+\Delta t}g_3^{2_k,(i-1)} \, {}^{t}G_1^k \\ {}^{t}h_2^k & 0 & 0 & {}^{t+\Delta t}g_1^{1_k,(i-1)} \, {}^{t}G_2^k & {}^{t+\Delta t}g_1^{2_k,(i-1)} \, {}^{t}G_2^k \\ 0 & {}^{t}h_2^k & 0 & {}^{t+\Delta t}g_2^{1_k,(i-1)} \, {}^{t}G_2^k & {}^{t+\Delta t}g_2^{2_k,(i-1)} \, {}^{t}G_2^k \\ 0 & 0 & {}^{t}h_2^k & {}^{t+\Delta t}g_3^{1_k,(i-1)} \, {}^{t}G_2^k & {}^{t+\Delta t}g_3^{2_k,(i-1)} \, {}^{t}G_2^k \\ {}^{t}h_3^k & 0 & 0 & {}^{t+\Delta t}g_1^{1_k,(i-1)} \, {}^{t}G_3^k & {}^{t+\Delta t}g_1^{2_k,(i-1)} \, {}^{t}G_3^k \\ 0 & {}^{t}h_3^k & 0 & {}^{t+\Delta t}g_2^{1_k,(i-1)} \, {}^{t}G_3^k & {}^{t+\Delta t}g_2^{2_k,(i-1)} \, {}^{t}G_3^k \\ 0 & 0 & {}^{t}h_3^k & {}^{t+\Delta t}g_3^{1_k,(i-1)} \, {}^{t}G_3^k & {}^{t+\Delta t}g_3^{2_k,(i-1)} \, {}^{t}G_3^k \end{bmatrix} \tag{3.141} $$

Having the matrices (3.135), (3.138), (3.139) and (3.141) these are used to compute the element's stiffness matrix, mass matrix, element loads etc. in exactly the same way as it is done for other ATENA's element.

3.14 Curvilinear Nonlinear 2D Isoparametric Layered Shell Triangular Elements

This section describes triangular shell finite elements. Their properties and their derivation are much the same as that for quadrilateral shell finite elements CCIsoShellQuad<xxxx> ... CCIsoShellQuad<xxxxxxxxx> described in the previous chapter. The only difference in that they feature triangular shape. Their geometry is depicted in the figure below

Depending on number of element nodes these finite elements call CCIsoShellTriangle<xxx> ... CCIsoShellTriangle<xxxxxx>.

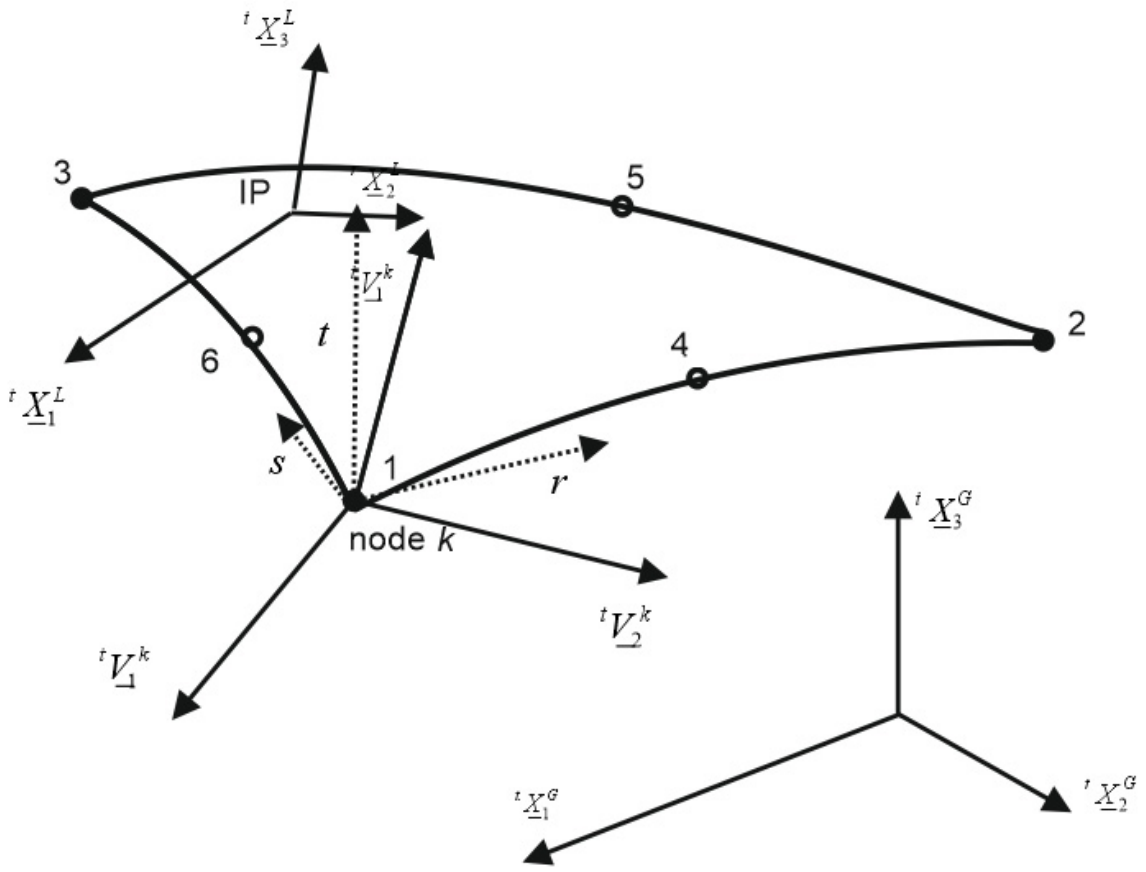


Fig. 3-37 CCIsoShell2D triangular elements

3.15 Curvilinear Nonlinear 3D Isoparametric Layered Shell Hexahedral Elements

A family of 3D isoparametric shell elements is presented, see the figure below. Their properties lie between degenerated Ahmad shell elements from Section 3.12 and full 3D brick elements from Section 3.5.

Shape and kinematic behaviour resembles that of the shell's element. All points through the shell's thickness remain located on a line passing thru the corresponding top and bottom nodes of the shell, however unlike in the classical shell theory, their distance can change. As for degrees of freedom, (DOFS), a typical 3D isoparametric shell element has 9 nodes at the top and nine nodes at the botom surface, each of them having 3 DOFS, (i.e. 3 displacements). A similar 2D

shell element would feature 9 nodes located at the shell's midplane, each of them having 5 DOFS, (3 displacements plus 2 rotations).

The new elements use full 3D static equations. i.e. the elements consider all 6 components of 3D stress and strain vector. Geometrical and material nonlinearity is supported. The governing equations are calculated and integrated in material points. Gauss integration is used in shell's plane direction, whilst layered concept is employed throughout the thickness of the shells, (i.e. rectangur quadrature). As each layer can use different material model, some layers can be employed for modelling of embedded reinforcement. The elements typically use 3 x 3 x *number_of_layers* integration (i.e. material) points.

The elements are suitable for both shallow and deep shells and are extremely simple for use, because they can be input and output as usual 3D solid hexahedral elements with 8, 20 or 27 nodes. Hence, these shells can be hadled with most 3D pre- and post-processors. They also use standard 3D material models, element loads and other boundary conditions designed for hexahedral elements.

The presented shell elements are particularly useful for structures that combine solid 3D elements and shell elements, because they do not imply any additional shell kinematic constraint that would harm an anjancent 3D solid elements. (Typical shell elements assume $\varepsilon_t = 0$ that enforces the same displacements of the corresponding top and bottom nodes in direction of their connecting line). They are designed for bent shells and to analyze these structures (with the same accuracy) they require far less finite elements compared to a similar analysis using standard hexahedral elements. On the other hand, the 3D behaviour of these elements involves a small overhead, so that standard 2D shell elements (with only 5 stress/strain components per material point) can perform in some cases slightly better. Nevertheless, the overhead is well paid off by easy of use of the presented elements, their nice 3D visualization, simple connection to adjacent 3D solid parts of the structure etc. In addition, the hiearchical isoparametric space interpolation (used for the presented 3D shell elements) ensures that finer and coarser meshes are easy to connect. Of coarse, this feature must be supported by pre- and postprocessor being used.

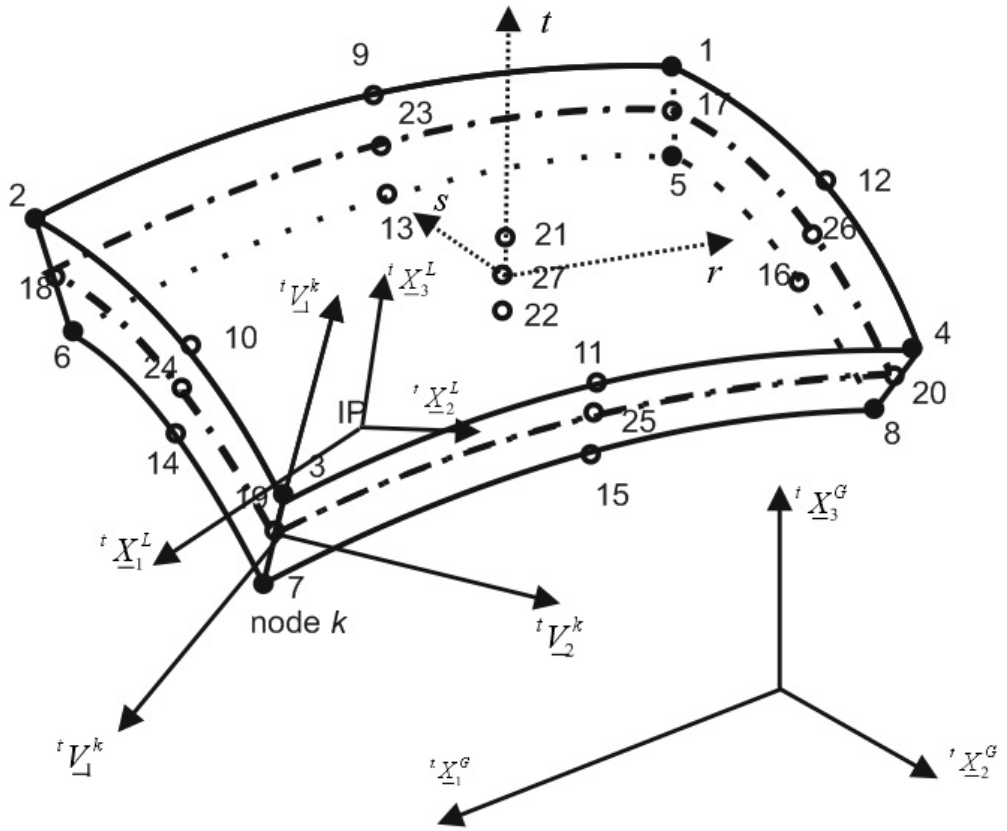


Fig. 3-38 Isoparametric 3D shell element - coordinate systems

Geometry and displacements are approximated by hiearchical isoparametric spatial interpolation, (similar to other 2D and 3D elements defined in previous sections). The elements have at minimum 4 points at its top and 4 points at its bottom surface. It corresponds to linear approximation and the element's name CCIsoShellBrick<xxxxxxxx>. The most accurate version of the elements uses nodes 1 to 16 and 21,22, see the figure above. Its name is CCIsoShellBrick<xxxxxxxxxxxxxxxxxx>. Such element can have curvilinear shape and features quadratic displacement approximation. Hierarchical concept the shell element is employed. Hence, the 3D shell element can have from 8 to 18 nodes. The nodes 1-8 are compulsory. Nodes 9-16 and 21,22 are optional. Nodes 17 to 22 are automatically removed from the element's incidences. They are considered only for the sake of compatibility with input data preprocessor. The <xxxxx..> string in the element name (following CCIsoShellBrick) specifies, which of the element's node is (or is not) included. An included node is market as "x", a node not included is marked as "_", (underscore). The shell's nodes are maped into the string as follows: <1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,21,22>. For example, CCIsoShellBrick<xxxxxxxxx_x_x_x_xx> uses nodes 1-8,9,11,13,15,21,22. Note that the bottom and top surface must use the same number and location of the optional nodes. Hence, if node 9 is included, node 13 must be included, too.

3.15.1 Geometry and displacements

The shell's geometry at the configuration time t and $t+dt$, (iteration (i-1) and (i)), is defined by:

$$
\begin{aligned}
{}^{t}x_i &= h_k\left(\frac{1+t}{2}\,{}^{t}X_i^{k,top}+\frac{1-t}{2}\,{}^{t}X_i^{k,bot}\right)\\
{}^{t+\Delta t}x_i^{(i-1)} &= h_k\left(\frac{1+t}{2}\,{}^{t+\Delta t}X_i^{k,top(i-1)}+\frac{1-t}{2}\,{}^{t+\Delta t}X_i^{k,bot(i-1)}\right)\\
{}^{t+\Delta t}x_i^{(i)} &= h_k\left(\frac{1+t}{2}\,{}^{t+\Delta t}X_i^{k,top(i)}+\frac{1-t}{2}\,{}^{t+\Delta t}X_i^{k,bot(i)}\right)
\end{aligned}
\tag{3.142}
$$

where i=1,2,3 is index relating to global axes x_1,x_2,x_3, (i.e. x,y,z), $h_k=h_k(r,s)$ is k-th interpolation function, (see Table 3-4), $k=1...n_G$ is number of the shell's nodes, n_G = number of the element's nodes used to approximate geometry, typically 8 or 9. ${}^{t}x_i$ represents i-th coordinate of a node of the element (at the specified time).

Displacements at time $t+\Delta t^{(i-1)}$, i=1,2,3 for global axes x,y,z, at iteration $(i-1)$ reads :

$$
{}^{t+\Delta t}u_i^{(i-1)} = {}^{t+\Delta t}x_i^{(i-1)} - {}^{t}x_i \tag{3.143}
$$

Substituting (3.142) into (3.143), i=1,2,3 for global axes x,y,z, we can derive

$$
\begin{aligned}
{}^{t+\Delta t}u_i^{(i-1)} &= h_k\left(\frac{1+t}{2}\,{}^{t+\Delta t}X_i^{k,top(i-1)}+\frac{1-t}{2}\,{}^{t+\Delta t}X_i^{k,bot(i-1)}\right)-h_k\left(\frac{1+t}{2}\,{}^{t}X_i^{k,top}+\frac{1-t}{2}\,{}^{t}X_i^{k,bot}\right)=\\
&=h_k\left(\frac{1+t}{2}\left({}^{t+\Delta t}X_i^{k,top(i-1)}-{}^{t}X_i^{k,top}\right)+\frac{1-t}{2}\left({}^{t+\Delta t}X_i^{k,bot(i-1)}-\frac{1-t}{2}\,{}^{t}X_i^{k,bot}\right)\right)\\
&=h_k\left(\frac{1+t}{2}\,{}^{t+\Delta t}U_i^{k,top(i-1)}+\frac{1-t}{2}\,{}^{t+\Delta t}U_i^{k,bot(i-1)}\right)
\end{aligned}
\tag{3.144}
$$

where

$$
{}^{t+\Delta t}U_i^{k,top(i-1)} = {}^{t+\Delta t}X_i^{k,top(i-1)} - {}^{t}X_i^{k},\quad {}^{t+\Delta t}U_i^{k,bot(i-1)} = {}^{t+\Delta t}X_i^{k,bot(i-1)} - {}^{t}X_i^{k}
$$

Displacement increments within i-th iteration are calculated as ${}^{t+\Delta t}u_i^{(i)} = {}^{t+\Delta t}x_i^{(i)} - {}^{t}x_i^{(i-1)}$:

$$
u_i^{(i)} = h_k\left(\frac{1+t}{2}\,{}^{t+\Delta t}U_i^{k,top(i)}+\frac{1-t}{2}\,{}^{t+\Delta t}U_i^{k,bot(i)}\right) \tag{3.145}
$$

where ${}^{t+\Delta t}U_i^{k,top(i)} = {}^{t+\Delta t}X_i^{k,top(i)} - {}^{t}X_i^{k(i-1)}$, ${}^{t+\Delta t}U_i^{k,bot(i)} = {}^{t+\Delta t}X_i^{k,bot(i)} - {}^{t}X_i^{k(i-1)}$. In the above $X_i^{k,top}$ and $X_i^{k,bot}$ is top and bottom nodal coordinate of node *i*. Similarly, ${}^{t+\Delta t}U_i^{k,top(i-1)}$, ${}^{t+\Delta t}U_i^{k,bot(i-1)}$ denotes displacements at the same node.

3.15.2 Green-Lagrange strains

The elements are derived using Green-Lagrange strains and 2nd Piola Kirchhoff stresses. Total Lagrangian formulation is employed, but after each load step we transform the analyzed model (and its stress and other tensors) to the coordinate system defined by the current shape of the model. (The standard Total Lagrangian formulation calculates all with respect to the original coordinate system without any transformation; Updated Lagrangian formulation carries all the transformation each transformation, BATHE(1982).)

The shell's total strains at time $t+\Delta t$, *i*-th iteration, are calculated: (i, j=1..3 for axis x,y,z)

$$\begin{aligned} {}^{t+\Delta t}_{t}\varepsilon_{ij}^{(i)} &= \frac{1}{2}\left({}^{t+\Delta t}_{t}u_{i,j}^{(i)} + {}^{t+\Delta t}_{t}u_{j,i}^{(i)} + {}^{t+\Delta t}_{t}u_{k,i}^{(i)}\ {}^{t+\Delta t}_{t}u_{k,j}^{(i)} \right) \\ &= \frac{1}{2}\left(\left({}^{t+\Delta t}_{t}u_{i,j}^{(i-1)} + {}_{t}u_{i,j}^{(i)} \right) + \left({}^{t+\Delta t}_{t}u_{j,i}^{(i-1)} + {}_{t}u_{j,i}^{(i)} \right) + \left({}^{t+\Delta t}_{t}u_{k,i}^{(i-1)} + {}_{t}u_{k,i}^{(i)} \right)\left({}^{t+\Delta t}_{t}u_{k,j}^{(i-1)} + {}_{t}u_{k,j}^{(i)} \right) \right) \\ &= {}^{t+\Delta t}_{t}\varepsilon_{ij}^{(i-1)} + {}_{t}e_{ij}^{(i)} + {}_{t}\eta_{ij}^{(i)} \end{aligned} \quad (3.146)$$

where ${}^{t+\Delta t}_{t}u_{i,j}^{(i)}$ is derivative of displacement ${}^{t+\Delta t}u_i^{(i)}$ with respect to axis ${}^{t}x_j$ at time *t*, i.e. at the beginning of time step. *(i)* refers to iteration number. Similarly, ${}_{t}u_{i,j}^{(i)}$ denotes displacement increment at the current iteration.

Subtracting ${}^{t+\Delta t}_{t}\varepsilon_{ij}^{(i-1)} = \frac{1}{2}\left({}^{t+\Delta t}_{t}u_{i,j}^{(i-1)} + {}^{t+\Delta t}_{t}u_{j,i}^{(i-1)} + {}^{t+\Delta t}_{t}u_{k,i}^{(i-1)}\ {}^{t+\Delta t}_{t}u_{k,j}^{(i-1)} \right)$ from (3.146) we can calculate linear and nonlinear strain increments ${}_{t}e_{ij}^{(i)}$ and ${}_{t}\eta_{ij}^{(i)}$:

$$\begin{aligned} {}_{t}e_{ij}^{(i)} &= \frac{1}{2}\left({}_{t}u_{i,j}^{(i)} + {}_{t}u_{j,i}^{(i)} + {}^{t+\Delta t}_{t}u_{k,i}^{(i-1)}\ {}_{t}u_{k,j}^{(i)} + {}^{t+\Delta t}_{t}u_{k,j}^{(i-1)}\ {}_{t}u_{k,i}^{(i)} \right) \\ {}_{t}\eta_{ij}^{(i)} &= \frac{1}{2}\left({}_{t}u_{k,i}^{(i)}\ {}_{t}u_{k,j}^{(i)} \right) \end{aligned} \quad (3.147)$$

Derivatives with respect to global $\bar{x} = (x_1, x_2, x_3)$ are calculated in standard way from derivatives with respect to curvilinear isoparametric coordinates $\bar{r} = (r, s, t) \equiv (r_1, r_2, r_3)$. For example, derivatives of a function $f(x_1, x_2, x_3)$ is:

$$\begin{bmatrix} \frac{\partial f}{\partial r} \\ \frac{\partial f}{\partial s} \\ \frac{\partial f}{\partial t} \end{bmatrix} = \begin{bmatrix} \frac{\partial x_1}{\partial r} & \frac{\partial x_2}{\partial r} & \frac{\partial x_3}{\partial r} \\ \frac{\partial x_1}{\partial s} & \frac{\partial x_2}{\partial s} & \frac{\partial x_3}{\partial s} \\ \frac{\partial x_1}{\partial t} & \frac{\partial x_2}{\partial t} & \frac{\partial x_3}{\partial t} \end{bmatrix} \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{bmatrix} = \mathbf{J} \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{bmatrix}, \text{ i.e. } \frac{\partial f}{\partial r_i} = \frac{\partial x_j}{\partial r_i}\frac{\partial f}{\partial x_j} = J_{ij}\frac{\partial f}{\partial x_j} \tag{3.148}$$

$$\begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \frac{\partial f}{\partial r} \\ \frac{\partial f}{\partial s} \\ \frac{\partial f}{\partial t} \end{bmatrix}, \textit{ i.e. } \frac{\partial f}{\partial x_i} = J_{ji}^{inv}\frac{\partial f}{\partial r_j} \tag{3.149}$$

The presented shell elements employs isoparametric hierarchical interpolation. Hence, coordinates ${}^t\overline{x}$ of a point are calculated by:

$${}^t x_i = h_k\left(\frac{1+t}{2}\,{}^t X_i^{k,top} + \frac{1-t}{2}\,{}^t X_i^{k,bot}\right) \tag{3.150}$$

where the interpolation functions $h_k(r,s)$ are enlisted in Table 1-3-1 and their derivatives $\frac{\partial {}^t x_i}{\partial r_i}$ with respect to *r*,*s*,*t* (to calculate **J**) are:

$$\begin{aligned} \frac{\partial {}^t x_i}{\partial r} &= \frac{\partial h_k}{\partial r}\left(\frac{1+t}{2}\,{}^t X_i^{k,top} + \frac{1-t}{2}\,{}^t X_i^{k,bot}\right) \\ \frac{\partial {}^t x_i}{\partial s} &= \frac{\partial h_k}{\partial s}\left(\frac{1+t}{2}\,{}^t X_i^{k,top} + \frac{1-t}{2}\,{}^t X_i^{k,bot}\right) \\ \frac{\partial {}^t x_i}{\partial t} &= \frac{h_k}{2}\left({}^t X_i^{k,top} - {}^t X_i^{k,bot}\right) \end{aligned} \tag{3.151}$$

The above expressions are employed to obtain derivatives of (total) displacements ${}^{t+\Delta t}u_i^{(i-1)}$ with respect to *r*,*s*,*t*. They are needed to calculate strains (3.147).

$$\begin{aligned} \frac{\partial {}^{t+\Delta t}u_i^{(i-1)}}{\partial r} &= \frac{\partial h_k}{\partial r}\left(\frac{1+t}{2}\,{}^{t+\Delta t}U_i^{k,top(i-1)} + \frac{1-t}{2}\,{}^{t+\Delta t}U_i^{k,bot(i-1)}\right) \\ \frac{\partial {}^{t+\Delta t}u_i^{(i-1)}}{\partial s} &= \frac{\partial h_k}{\partial s}\left(\frac{1+t}{2}\,{}^{t+\Delta t}U_i^{k,top(i-1)} + \frac{1-t}{2}\,{}^{t+\Delta t}U_i^{k,bot(i-1)}\right) \\ \frac{\partial {}^{t+\Delta t}u_i^{(i-1)}}{\partial t} &= \frac{h_k}{2}\left({}^{t+\Delta t}U_i^{k,top(i-1)} - {}^{t+\Delta t}U_i^{k,bot(i-1)}\right) \end{aligned} \tag{3.152}$$

-

Derivatives of displacement increments with respect to *r*,*s*,*t*:

$$\begin{aligned}
\frac{\partial^{t+\Delta t}u_i^{(i)}}{\partial r} &= \frac{\partial h_k}{\partial r}\left(\frac{1+t}{2}\,{}^{t+\Delta t}U_i^{k,top(i)} + \frac{1-t}{2}\,{}^{t+\Delta t}U_i^{k,bot(i)}\right)\\
\frac{\partial^{t+\Delta t}u_i^{(i)}}{\partial s} &= \frac{\partial h_k}{\partial s}\left(\frac{1+t}{2}\,{}^{t+\Delta t}U_i^{k,top(i)} + \frac{1-t}{2}\,{}^{t+\Delta t}U_i^{k,bot(i)}\right)\\
\frac{\partial^{t+\Delta t}u_i^{(i)}}{\partial t} &= \frac{h_k}{2}\left({}^{t+\Delta t}U_i^{k,top(i)} - {}^{t+\Delta t}U_i^{k,bot(i)}\right)
\end{aligned} \tag{3.153}$$

To proceed further in the derivation of the 3D isoparametric element, we need to calculate derivatives of the displacement increments with respect to ${}^t\overline{x} = \left({}^tx_1, {}^tx_2, {}^tx_3\right)$. This is achieved using (3.149) and (3.153):

$$\frac{\partial u_i^{(i)}}{\partial\,{}^tx_j} = {}^tJ_{jl}^{inv}\frac{\partial u_i^{(i)}}{\partial r_l} = {}^tJ_{j1}^{inv}\frac{\partial u_i^{(i)}}{\partial r} + {}^tJ_{j2}^{inv}\frac{\partial u_i^{(i)}}{\partial s} + {}^tJ_{j3}^{inv}\frac{\partial u_i^{(i)}}{\partial t} \tag{3.154}$$

$$\begin{aligned}
\frac{\partial u_i^{(i)}}{\partial\,{}^tx_j} &= {}^tJ_{j1}^{inv}\frac{\partial h_k}{\partial r}\left(\frac{1+t}{2}\,{}^{t+\Delta t}U_i^{k,top(i)} + \frac{1-t}{2}\,{}^{t+\Delta t}U_i^{k,bot(i)}\right)\\
&+ {}^tJ_{j2}^{inv}\frac{\partial h_k}{\partial s}\left(\frac{1+t}{2}\,{}^{t+\Delta t}U_i^{k,top(i)} + \frac{1-t}{2}\,{}^{t+\Delta t}U_i^{k,bot(i)}\right)\\
&+ {}^tJ_{j3}^{inv}\frac{h_k}{2}\left({}^{t+\Delta t}U_i^{k,top(i)} - {}^{t+\Delta t}U_i^{k,bot(i)}\right)
\end{aligned} \tag{3.155}$$

After some rearrangement Eqn. (3.155) yields:

$$\begin{aligned}
\frac{\partial u_i^{(i)}}{\partial\,{}^tx_j} &= {}^{t+\Delta t}U_i^{k,top(i)}\left({}^tJ_{j1}^{inv}\frac{\partial h_k}{\partial r}\frac{1+t}{2} + {}^tJ_{j2}^{inv}\frac{\partial h_k}{\partial s}\frac{1+t}{2} + {}^tJ_{j3}^{inv}\frac{h_k}{2}\right) +\\
&{}^{t+\Delta t}U_i^{k,bot(i)}\left({}^tJ_{j1}^{inv}\frac{\partial h_k}{\partial r}\frac{1-t}{2} + {}^tJ_{j2}^{inv}\frac{\partial h_k}{\partial s}\frac{1-t}{2} + {}^tJ_{j3}^{inv}\frac{h_k}{2}\right) =\\
&h_{k,j}^{top}\,{}^{t+\Delta t}U_i^{k,top(i)} + h_{k,j}^{bot}\,{}^{t+\Delta t}U_i^{k,bot(i)}
\end{aligned} \tag{3.156}$$

where

$$h_{k,j}^{top} = \left({}^tJ_{j1}^{inv}\frac{\partial h_k}{\partial r}\frac{1+t}{2} + {}^tJ_{j2}^{inv}\frac{\partial h_k}{\partial s}\frac{1+t}{2} + {}^tJ_{j3}^{inv}\frac{h_k}{2}\right),\ h_{k,j}^{bot} = \left({}^tJ_{j1}^{inv}\frac{\partial h_k}{\partial r}\frac{1-t}{2} + {}^tJ_{j2}^{inv}\frac{\partial h_k}{\partial s}\frac{1-t}{2} + {}^tJ_{j3}^{inv}\frac{h_k}{2}\right)$$

At this place, we can derive final expression to compute linear and nonlinear strains increments. Linear strains ${}_te_{ij}^{(i)}$ are calculated as follows, see (3.147):

$$ {}_t\overline{e}^{(i)} = \begin{bmatrix} {}_t e_{11}^{(i)} \\ {}_t e_{22}^{(i)} \\ {}_t e_{33}^{(i)} \\ 2\,{}_t e_{12}^{(i)} \\ 2\,{}_t e_{23}^{(i)} \\ 2\,{}_t e_{13}^{(i)} \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1^{L_0} + \mathbf{B}_1^{L_1} & \dots & \mathbf{B}_k^{L_0} + \mathbf{B}_k^{L_1} & \dots & \mathbf{B}_n^{L_0} + \mathbf{B}_n^{L_1} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{u}}_1^{(i)} \\ \dots \\ \widehat{\mathbf{u}}_k^{(i)} \\ \dots \\ \widehat{\mathbf{u}}_n^{(i)} \end{bmatrix} \tag{3.157} $$

$$ \mathbf{B}_k^{L_0} = \begin{bmatrix} {}_t h_{k,1}^{top} & 0 & 0 & {}_t h_{k,1}^{bot} & 0 & 0 \\ 0 & {}_t h_{k,2}^{top} & 0 & 0 & {}_t h_{k,2}^{top} & 0 \\ 0 & 0 & {}_t h_{k,3}^{top} & 0 & 0 & {}_t h_{k,3}^{top} \\ {}_t h_{k,2}^{top} & {}_t h_{k,1}^{top} & 0 & {}_t h_{k,2}^{top} & {}_t h_{k,1}^{top} & 0 \\ 0 & {}_t h_{k,3}^{top} & {}_t h_{k,2}^{top} & 0 & {}_t h_{k,3}^{top} & {}_t h_{k,2}^{top} \\ {}_t h_{k,3}^{top} & 0 & {}_t h_{k,1}^{top} & {}_t h_{k,3}^{top} & 0 & {}_t h_{k,1}^{top} \end{bmatrix} \tag{3.158} $$

where $\widehat{\mathbf{u}}_k^{(i)} = \left[U_1^{k,top(i)}, U_2^{k,top(i)}, U_3^{k,top(i)}, U_1^{k,bot(i)}, U_2^{k,bot(i)}, U_3^{k,bot(i)} \right]^T$ at node *k*.

Introducing

$$ l_{ij}^{(i-1)} = \frac{\partial^{t+\Delta t} u_i^{(i-1)}}{\partial^t x_j} = {}_t^{t+\Delta t} u_{i,j}^{(i-1)} \tag{3.159} $$

we can write

$$ \begin{aligned} &{}_t^{t+\Delta t} u_{m,i}^{(i-1)}\; {}_t u_{m,j}^{(i)} + {}_t^{t+\Delta t} u_{m,j}^{(i-1)}\; {}_t u_{m,i}^{(i)} = \\ &\quad l_{mi}^{(i-1)} \left(h_{k,j}^{top}\; {}^{t+\Delta t} U_m^{k,top(i)} + h_{k,j}^{bot}\; {}^{t+\Delta t} U_m^{k,bot(i)} \right) + l_{mj}^{(i-1)} \left(h_{k,i}^{top}\; {}^{t+\Delta t} U_m^{k,top(i)} + h_{k,i}^{bot}\; {}^{t+\Delta t} U_m^{k,bot(i)} \right) = \\ &\quad {}^{t+\Delta t} U_m^{k,top(i)} \left(l_{mi}^{(i-1)} h_{k,j}^{top} + l_{mj}^{(i-1)} h_{k,i}^{top} \right) + {}^{t+\Delta t}\; U_m^{k,bot(i)} \left(l_{mi}^{(i-1)} h_{k,j}^{bot} + l_{mj}^{(i-1)} h_{k,i}^{bot} \right) \end{aligned} \tag{3.160} $$

Finally, matrix $\mathbf{B}^{L_1}$ yields

$$ \mathbf{B}_k^{L_1} = \begin{bmatrix} l_{11}^{(i-1)}\; {}_t h_{k,1}^{top} & l_{21}^{(i-1)}\; {}_t h_{k,1}^{top} & l_{31}^{(i-1)}\; {}_t h_{k,1}^{top} & l_{11}^{(i-1)}\; {}_t h_{k,1}^{bot} \\ l_{12}^{(i-1)} h_{k,2}^{top} & l_{22}^{(i-1)} h_{k,2}^{top} & l_{32}^{(i-1)} h_{k,2}^{top} & l_{12}^{(i-1)} h_{k,2}^{bot} \\ l_{13}^{(i-1)} h_{k,3}^{top} & l_{23}^{(i-1)} h_{k,3}^{top} & l_{33}^{(i-1)} h_{k,3}^{top} & l_{13}^{(i-1)} h_{k,3}^{bot} \\ l_{11}^{(i-1)} h_{k,2}^{top} + l_{12}^{(i-1)} h_{k,1}^{top} & l_{21}^{(i-1)} h_{k,2}^{top} + l_{22}^{(i-1)} h_{k,1}^{top} & l_{31}^{(i-1)} h_{k,2}^{top} + l_{32}^{(i-1)} h_{k,1}^{top} & l_{11}^{(i-1)} h_{k,2}^{bot} + l_{12}^{(i-1)} h_{k,1}^{bot} \\ l_{12}^{(i-1)} h_{k,3}^{top} + l_{13}^{(i-1)} h_{k,2}^{top} & l_{22}^{(i-1)} h_{k,3}^{top} + l_{23}^{(i-1)} h_{k,2}^{top} & l_{32}^{(i-1)} h_{k,3}^{top} + l_{33}^{(i-1)} h_{k,2}^{top} & l_{12}^{(i-1)} h_{k,3}^{bot} + l_{13}^{(i-1)} h_{k,2}^{bot} \\ l_{11}^{(i-1)} h_{k,3}^{top} + l_{13}^{(i-1)} h_{k,1}^{top} & l_{21}^{(i-1)} h_{k,3}^{top} + l_{23}^{(i-1)} h_{k,1}^{top} & l_{31}^{(i-1)} h_{k,3}^{top} + l_{33}^{(i-1)} h_{k,1}^{top} & l_{11}^{(i-1)} h_{k,3}^{bot} + l_{13}^{(i-1)} h_{k,1}^{bot} \end{bmatrix} $$

$$
\begin{array}{lll}
\ldots & {}^{(i-1)}_{21}\,{}_{t}h^{bot}_{k,1} & {}^{(i-1)}_{31}\,{}_{t}h^{bot}_{k,1} \\
\ldots & l^{(i-1)}_{22}h^{bot}_{k,2} & l^{(i-1)}_{32}h^{bot}_{k,2} \\
\ldots & l^{(i-1)}_{23}h^{bot}_{k,3} & l^{(i-1)}_{33}h^{bot}_{k,3} \\
\ldots & l^{(i-1)}_{21}h^{bot}_{k,2}+l^{(i-1)}_{22}h^{bot}_{k,1} & l^{(i-1)}_{31}h^{bot}_{k,2}+l^{(i-1)}_{32}h^{bot}_{k,1} \\
\ldots & l^{(i-1)}_{22}h^{bot}_{k,3}+l^{(i-1)}_{23}h^{bot}_{k,2} & l^{(i-1)}_{32}h^{bot}_{k,3}+l^{(i-1)}_{33}h^{bot}_{k,2} \\
\ldots & l^{(i-1)}_{21}h^{bot}_{k,3}+l^{(i-1)}_{23}h^{bot}_{k,1} & l^{(i-1)}_{31}h^{bot}_{k,3}+l^{(i-1)}_{33}h^{bot}_{k,1}
\end{array}\Bigg] \qquad (3.161)
$$

Assembling stresses at time $t+\Delta t$, iteration (*i-1*) into matrix ${}^{t+\Delta t}_{t}\widehat{\mathbf{S}}^{(i-1)}$, participation of nonlinear strains ${}_{t}\eta^{(i)}_{ij}$ is, see (3.147)

$$
\begin{aligned}
{}^{t+\Delta t}_{t}S^{(i-1)}_{ij}\,\delta\left({}_{t}\eta^{(i)}_{ij}\right) &= {}^{t+\Delta t}_{t}S^{(i-1)}_{ij}\,\delta\left({}_{t}\eta^{(i)}_{ij}\right) = {}^{t+\Delta t}_{t}S^{(i-1)}_{ij}\,\delta\left(\frac{1}{2}\left({}_{t}u^{(i)}_{k,i}\;{}_{t}u^{(i)}_{k,j}\right)\right) \\
&= {}^{t+\Delta t}_{t}S^{(i-1)}_{ij}\left(\frac{1}{2}\left(\delta\,{}_{t}u^{(i)}_{k,i}\;{}_{t}u^{(i)}_{k,j}+{}_{t}u^{(i)}_{k,i}\;{}_{t}\delta u^{(i)}_{k,j}\right)\right) \\
&= {}^{t+\Delta t}_{t}S^{(i-1)}_{ij}\delta\,{}_{t}u^{(i)}_{k,i}\;{}_{t}u^{(i)}_{k,j} \\
&= \begin{bmatrix}\delta\widehat{\mathbf{u}}^{(i)}_{1}\\ \ldots \\ \delta\widehat{\mathbf{u}}^{(i)}_{k}\\ \ldots \\ \delta\widehat{\mathbf{u}}^{(i)}_{n}\end{bmatrix}^{T}\left(\mathbf{B}^{NL}\right)^{T}\;{}^{t+\Delta t}_{t}\widehat{\mathbf{S}}^{(i-1)}\,\mathbf{B}^{NL}\begin{bmatrix}\widehat{\mathbf{u}}^{(i)}_{1}\\ \ldots \\ \widehat{\mathbf{u}}^{(i)}_{k}\\ \ldots \\ \widehat{\mathbf{u}}^{(i)}_{n}\end{bmatrix}
\end{aligned} \qquad (3.162)
$$

$$
\mathbf{B}^{NL}\left[\mathbf{B}^{NL}_{1}\quad\ldots\quad\mathbf{B}^{NL}_{k}\quad\ldots\quad\mathbf{B}^{NL}_{n}\right]
$$

$$
\mathbf{B}^{NL}_{k}=\begin{bmatrix}
{}_{t}h^{top}_{k,1} & 0 & 0 & {}_{t}h^{bot}_{k,1} & 0 & 0 \\
0 & {}_{t}h^{top}_{k,1} & 0 & 0 & {}_{t}h^{bot}_{k,1} & 0 \\
0 & 0 & {}_{t}h^{top}_{k,1} & 0 & 0 & {}_{t}h^{bot}_{k,1} \\
{}_{t}h^{top}_{k,2} & 0 & 0 & {}_{t}h^{bot}_{k,2} & 0 & 0 \\
0 & {}_{t}h^{top}_{k,2} & 0 & 0 & {}_{t}h^{bot}_{k,2} & 0 \\
0 & 0 & {}_{t}h^{top}_{k,2} & 0 & 0 & {}_{t}h^{bot}_{k,2} \\
{}_{t}h^{top}_{k,3} & 0 & 0 & {}_{t}h^{bot}_{k,3} & 0 & 0 \\
0 & {}_{t}h^{top}_{k,3} & 0 & 0 & {}_{t}h^{bot}_{k,3} & 0 \\
0 & 0 & {}_{t}h^{top}_{k,3} & 0 & 0 & {}_{t}h^{bot}_{k,3}
\end{bmatrix} \qquad (3.163)
$$

$$
{}^{t+\Delta t}_{t}\widehat{\mathbf{S}}^{(i-1)} = \begin{bmatrix}
{}^{t+\Delta t}_{t}s_{11}^{(i-1)} & & & & & & & & \\
0 & {}^{t+\Delta t}_{t}s_{11}^{(i-1)} & & & & & & & \\
0 & 0 & {}^{t+\Delta t}_{t}s_{11}^{(i-1)} & & & & & & \\
{}^{t+\Delta t}_{t}s_{12}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{22}^{(i-1)} & & & & & \\
0 & {}^{t+\Delta t}_{t}s_{12}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{22}^{(i-1)} & & & & \\
0 & 0 & {}^{t+\Delta t}_{t}s_{12}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{22}^{(i-1)} & & & \\
{}^{t+\Delta t}_{t}s_{13}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{23}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{33}^{(i-1)} & & \\
0 & {}^{t+\Delta t}_{t}s_{13}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{23}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{33}^{(i-1)} & \\
0 & 0 & {}^{t+\Delta t}_{t}s_{13}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{23}^{(i-1)} & 0 & 0 & {}^{t+\Delta t}_{t}s_{33}^{(i-1)}
\end{bmatrix} \quad (3.164)
$$

Using (3.148) and (3.149) it follows to present final expression for computation of space derivatives of $f(x_1, x_2, x_3)$:

$$
\frac{\partial f}{\partial x_i} = J^{inv}_{ji}\left(F_k \frac{\partial h^k}{\partial r_j} \right); \quad h^k = h^k(r,s,t) = \hat{h}^k(r,s)\,\widehat{h}^k(t)
$$

$$
\frac{\partial f}{\partial x_i} = J^{inv}_{ji}\left(F_k \frac{\partial}{\partial r_j}\left(\hat{h}^k(r,s)\,\widehat{h}^k(t) \right) \right) = J^{inv}_{ji}\left(\left(F_k \left(\frac{\partial \hat{h}^k}{\partial r_j}\widehat{h}^k + \hat{h}^k \frac{\partial \widehat{h}^k}{\partial r_j} \right) \right) \right) =
$$

$$
J^{inv}_{1i}\left(\left(F_k \left(\frac{\partial \hat{h}^k}{\partial r}\widehat{h}^k + \hat{h}^k \frac{\partial \widehat{h}^k}{\partial r} \right) \right) \right) + J^{inv}_{2i}\left(\left(F_k \left(\frac{\partial \hat{h}^k}{\partial s}\widehat{h}^k + \hat{h}^k \frac{\partial \widehat{h}^k}{\partial s} \right) \right) \right) + J^{inv}_{3i}\left(\left(F_k \left(\frac{\partial \hat{h}^k}{\partial t}\widehat{h}^k + \hat{h}^k \frac{\partial \widehat{h}^k}{\partial t} \right) \right) \right) =
$$

$$
J^{inv}_{1i}\left(F_k \frac{\partial \hat{h}^k}{\partial r}\widehat{h}^k \right) + J^{inv}_{2i\ k}\left(F \frac{\partial \hat{h}^k}{\partial s}\widehat{h}^k \right) + J^{inv}_{3i}\left(F_k \hat{h}^k \frac{\partial \widehat{h}^k}{\partial t} \right) \quad (3.165)
$$

Having all the matrices and relationships above, the rest of derivation of the presented isoparametric shell elements is straightforward. Simply use the matrices $\mathbf{B}^{L_0}$, $\mathbf{B}^{L_1}$, $\mathbf{B}^{NL}$ and ${}^{t+\Delta t}_{t}\widehat{\mathbf{S}}^{(i-1)}$ to calculate structural stiffness matrices ${}_t\boldsymbol{K}_L$, ${}^{t+\Delta t}_{t}\boldsymbol{K}^{(i-1)}_{NL}$, vectors of nodal forces ${}^{t+\Delta t}\underline{F}^{(i-1)}$, and loads ${}^{t+\Delta t}\underline{R}$ as described in the Section Problem Discretisation Using Finite Element Method earlier in this document.

3.16 Curvilinear Nonlinear 3D Isoparametric Layered Shell Wedge Elements

This section describes wedge shell finite elements. Their properties and their derivation are much the same as that for hexahedral shell finite elements CCIsoShellBrick<xxxxxxxx> ... CCIsoShellBrick<xxxxxxxxxxxxxxxxxx> described in the previous chapter. The only diference in that they feature wedge shape. Their geometry is depicted in the figure below

Depending on number of element nodes these finite elements call CCIsoShellWedge<xxxxxx> ... CCIsoShellWedge<xxxxxxxxxxxx>.

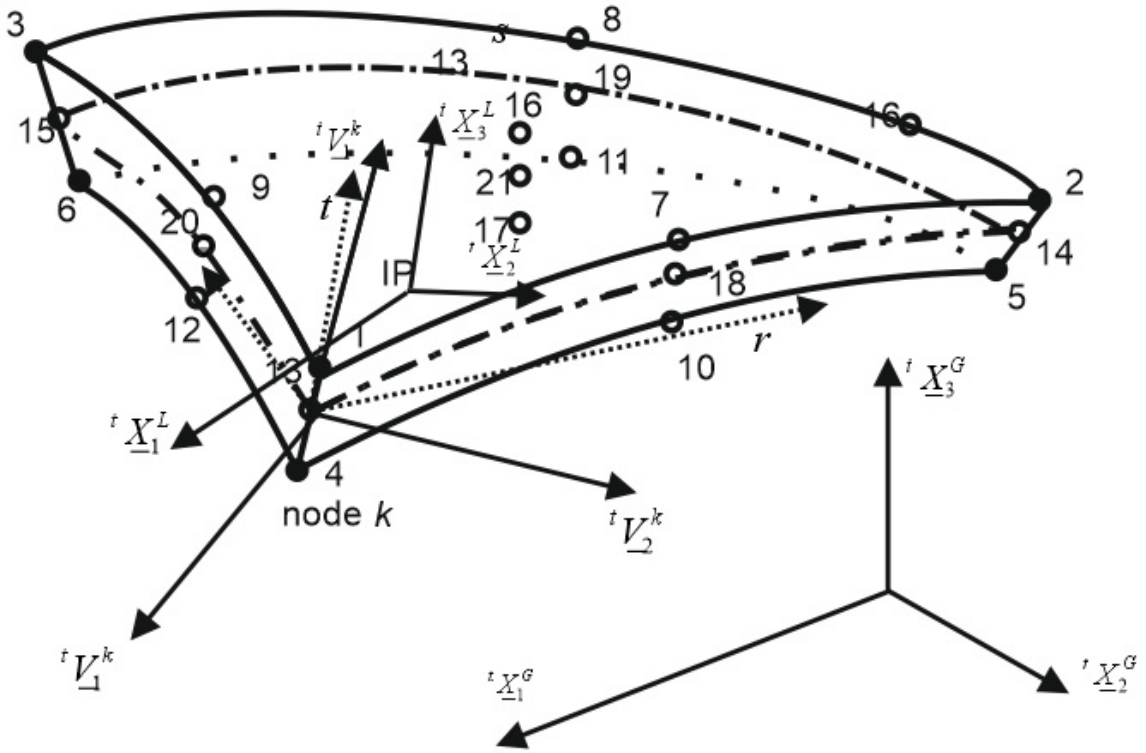


Fig. 3-39 CCIsoShell3D wedge elements

3.17 Curvilinear Nonlinear 3D Beam Element

A curvilinear 3D beam finite element CCBeamNL is described here. The element is based on a similar beam element from BATHE (1982). It is fully nonlinear, in terms of its geometry and material response. It uses quadratic approximation of its shape, so it can be curvilinear, twisted, with variable dimensions of the cross-sections. Moreover, beam's cross-sections can be of any shape, optionally even with holes.

The element belongs to the group of isoparametric elements with Gauss integration along its axis and trapezoidal (Newton-Cotes) quadrature within the cross-section. The integration (or material) points are placed in a way similar to the layered concept applied to shell elements, however, the "layers" are located in both "*s,t*" directions.

3.17.1 Geometry and Displacements and Rotations Fields

Geometry of the element is depicted in Fig. 3-40. The depicted brick nodes specification is employed to ensure compatibility of the element with ATENA preprocessor. The beam 3D nodes definition is used by ATENA postprocessor. The element response is computed within the 1D beam geometry. Thus, on input the element has 20 nodes, while during the calculation it has only 15 nodes, i.e. 12 nodes for 3D beam shape definition and 3 nodes for the 1D beam geometry. Any of the 15 nodes can be subject to a kinematic or static constraint. The 1D beam nodes have 6 degrees of freedom (dofs) – three displacements and three rotations with respect to global coordinate axes. The 3D beam nodes allocate only the three displacement dofs per node. The redundant brick nodes are ignored, and they allocate no dofs.

The element uses three configurations. The reference configuration corresponds to shape of the beam at the beginning of the step, i.e. prior any load in the current step is applied was employed. It is used as a reference coordinate system for all calculation within a loading step *t*, with respect to which all derivatives are computed. This configuration is denoted by a *t* superscript left to a referred symbol, e.g. ${}^{t}x$. The element shape after all previous iterations within the current step and prior the current iteration is denoted by $t+dt$ superscript, ${}^{t+dt}x$. Increments within the current iteration do not use any superscript, e.g. *x*.

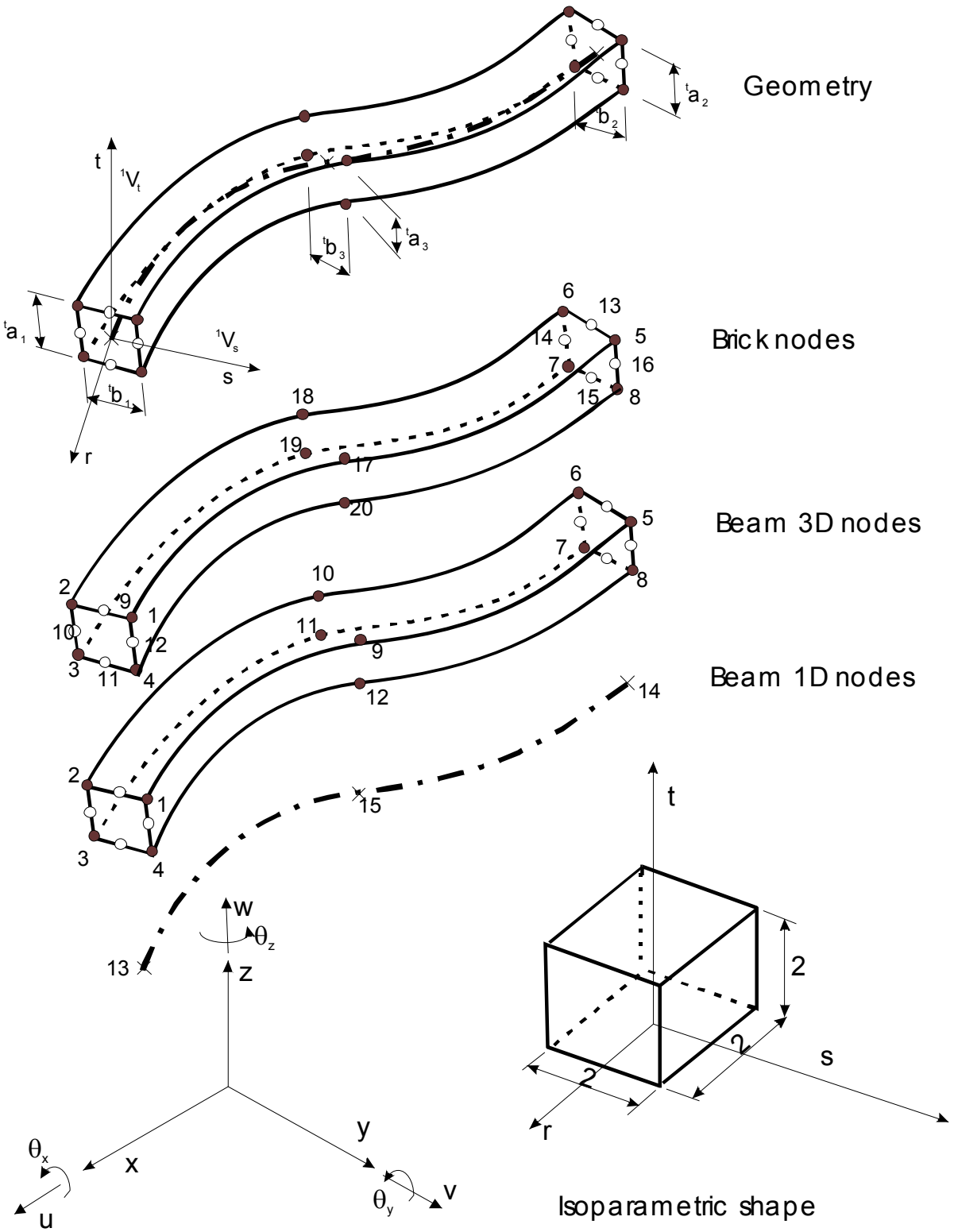


Fig. 3-40 CCBeamNL element

The beam's geometry at the configuration t and $t+dt$ is defined by:

$$
\begin{aligned}
{}^{t}x &= h_i\left({}^{t}X_i + \frac{t}{2}\,{}^{t}a_i\,{}^{t}V_i^{t_x} + \frac{s}{2}\,{}^{t}b_i\,{}^{t}V_i^{s_x} \right) \\
{}^{t}y &= h_i\left({}^{t}Y_i + \frac{t}{2}\,{}^{t}a_i\,{}^{t}V_i^{t_y} + \frac{s}{2}\,{}^{t}b_i\,{}^{t}V_i^{s_y} \right) \\
{}^{t}z &= h_i\left({}^{t}Z_i + \frac{t}{2}\,{}^{t}a_i\,{}^{t}V_i^{t_z} + \frac{s}{2}\,{}^{t}b_i\,{}^{t}V_i^{s_z} \right)
\end{aligned}
\tag{3.166}
$$

In the above *i* refers to axial nodes, i.e. $i = 1..3$ for the nodes 13,14,15, see the 1D beam nodes. $h_i = h_i(r)$ is i-th nodal interpolation function *i* described in Section 3.2. $\left[{}^{t}X_i, {}^{t}Y_i, {}^{t}Z_i \right]^T$ are global coordinates of a node *i* at time *t*. The vectors $\left[{}^{t}V_i^{t_x}, {}^{t}V_i^{t_y}, {}^{t}V_i^{t_z} \right]^T$, $\left[{}^{t}V_i^{s_x}, {}^{t}V_i^{s_y}, {}^{t}V_i^{s_z} \right]^T$ are the vectors ${}^{t}V_t$, ${}^{t}V_s$ depicted in Fig. 3-40, in a cross section *i*, at time *t*, which define local coordinate axis *s,t.* The symbols ${}^{t}a_i, {}^{t}b_i$ refers to dimensions of the cross section *i*, time *t*; see the figure, too.

Geometry of the beam at time $t+dt$ is defined in a similar way:

$$
\begin{aligned}
{}^{t+dt}x &= h_i\left({}^{t+dt}X_i + \frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_x} + \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x} \right) \\
{}^{t+dt}y &= h_i\left({}^{t+dt}Y_i + \frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_y} + \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_y} \right) \\
{}^{t+dt}z &= h_i\left({}^{t+dt}Z_i + \frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_z} + \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_z} \right)
\end{aligned}
\tag{3.167}
$$

The element's displacements at time $t+dt$ is calculated as follows:

$$
\begin{aligned}
{}^{t+dt}u &= {}^{t+dt}x - {}^{t}x \\
{}^{t+dt}v &= {}^{t+dt}y - {}^{t}y \\
{}^{t+dt}w &= {}^{t+dt}z - {}^{t}z
\end{aligned}
\tag{3.168}
$$

and displacement increments within a iteration:

$$
\begin{aligned}
u &= h_i\left(U_i + \frac{t}{2}\,{}^{t}a_i V_i^{t_x} + \frac{s}{2}\,{}^{t}b_i V_i^{s_x} \right) \\
v &= h_i\left(V_i + \frac{t}{2}\,{}^{t}a_i V_i^{t_y} + \frac{s}{2}\,{}^{t}b_i V_i^{s_y} \right) \\
w &= h_i\left(W_i + \frac{t}{2}\,{}^{t}a_i V_i^{t_z} + \frac{s}{2}\,{}^{t}b_i V_i^{s_z} \right)
\end{aligned}
\tag{3.169}
$$

In the above equation the vectors V_i^t, V_i^s are $V_i^t = {}^{t+dt}V_i^t - {}^{t}V_i^t$ and $V_i^s = {}^{t+dt}V_i^s - {}^{t}V_i^s$ are approximated by

$$
\begin{aligned}
V_i^{t_x} &= \left({}^{t+dt}V_i^{t_z}\theta_i^y - {}^{t+dt}V_i^{t_y}\theta_i^z \right) \\
V_i^{t_y} &= \left({}^{t+dt}V_i^{t_x}\theta_i^z - {}^{t+dt}V_i^{t_z}\theta_i^x \right) \\
V_i^{t_z} &= \left({}^{t+dt}V_i^{t_y}\theta_i^x - {}^{t+dt}V_i^{t_x}\theta_i^y \right) \\
\\
V_i^{s_x} &= \left({}^{t+dt}V_i^{s_z}\theta_i^y - {}^{t+dt}V_i^{s_y}\theta_i^z \right) \\
V_i^{s_y} &= \left({}^{t+dt}V_i^{s_x}\theta_i^z - {}^{t+dt}V_i^{s_z}\theta_i^x \right) \\
V_i^{s_z} &= \left({}^{t+dt}V_i^{s_y}\theta_i^x - {}^{t+dt}V_i^{s_x}\theta_i^y \right)
\end{aligned}
\tag{3.170}
$$

The parameters $\theta_i^x, \theta_i^y, \theta_i^z$ are rotations around the global axis, with respect to beginning of the current load step. Note that (3.170) is valid only approximately.

3.17.2 Strain and Stress Definition

The element uses Green-Lagrange strain and Piola-Kirchhof stresses, see Section 1.4.2 and Section 1.3.2. transformed to the local isoparametric *r,s,t* coordinate system. As the beam theory implies, only normal strain component ε_r and shear components γ_{rs}, γ_{rt} are considered. The stress vector includes the corresponding $\sigma_{rr}, \tau_{rs}, \tau_{rt}$ entries, whereby the remaining strains have to remain zero. The procedure of calculation stress-strain response is as follows:

1. Calculate all 6 components of Green-Lagrange strains **(1.8)** and their increments within global coordinate systems. The increments are computed with respect to the beginning of the current load step.
2. Transform the strains increments into local *r,s,t* coordinate system.
3. Zeroise components $\Delta\varepsilon_{ss}, \Delta\varepsilon_{tt}, \Delta\gamma_{st}$.
4. Execute material law to compute corresponding stresses.
5. Transform the stresses to the global coordinate system.

The following expressions are used to calculate displacement derivatives needed for calculation of the strains:

$$
\begin{bmatrix} \frac{df}{dx} \\ \frac{df}{dy} \\ \frac{df}{dz} \end{bmatrix} = \begin{bmatrix} \frac{dx}{dr} & \frac{dy}{dr} & \frac{dz}{dr} \\ \frac{dx}{ds} & \frac{dy}{ds} & \frac{dz}{ds} \\ \frac{dx}{dt} & \frac{dy}{dt} & \frac{dz}{dt} \end{bmatrix}^{-1} \begin{bmatrix} \frac{df}{dr} \\ \frac{df}{ds} \\ \frac{df}{dt} \end{bmatrix} = J^{-1} \begin{bmatrix} \frac{df}{dr} \\ \frac{df}{ds} \\ \frac{df}{dt} \end{bmatrix}
\tag{3.171}
$$

where *f* is a displacement function to be derived.

3.17.3 Matrices Used in the Beam Element Formulation

Substituting equations (3.166) to (3.171) into the expressions for calculating element matrices **(1.31)** to **(1.34)** all important matrices and vectors of the beam element can be calculated. Their

explicit presentation is beyond the scope of this document. Nevertheless, the most important ones are now given:

The Jacobian matrix:

$$
\begin{aligned}
J_{11} &= \frac{\partial^t x}{\partial r} = \frac{\partial h_i}{\partial r}\left({}^t X_i + \frac{t}{2}\,{}^t a_i {}^t V_i^{t_x} + \frac{s}{2}\,{}^t b_i {}^t V_i^{s_x} \right) \\
J_{12} &= \frac{\partial^t y}{\partial r} = \frac{\partial h_i}{\partial r}\left({}^t Y_i + \frac{t}{2}\,{}^t a_i {}^t V_i^{t_y} + \frac{s}{2}\,{}^t b_i {}^t V_i^{s_y} \right) \\
J_{13} &= \frac{\partial^t z}{\partial r} = \frac{\partial h_i}{\partial r}\left({}^t Z_i + \frac{t}{2}\,{}^t a_i {}^t V_i^{t_z} + \frac{s}{2}\,{}^t b_i {}^t V_i^{s_z} \right) \\
J_{21} &= \frac{\partial^t x}{\partial s} = h_i\left(\frac{1}{2}\,{}^t b_i {}^t V_i^{s_x} \right) \\
J_{22} &= \frac{\partial^t y}{\partial s} = h_i\left(\frac{1}{2}\,{}^t b_i {}^t V_i^{s_y} \right) \\
J_{23} &= \frac{\partial^t y}{\partial t} = h_i\left(\frac{1}{2}\,{}^t b_i {}^t V_i^{s_z} \right) \\
J_{31} &= \frac{\partial^t x}{\partial t} = h_i\left(\frac{1}{2}\,{}^t a_i {}^t V_i^{t_x} \right) \\
J_{32} &= \frac{\partial^t y}{\partial t} = h_i\left(\frac{1}{2}\,{}^t a_i {}^t V_i^{t_y} \right) \\
J_{33} &= \frac{\partial^t z}{\partial t} = h_i\left(\frac{1}{2}\,{}^t a_i {}^t V_i^{t_z} \right)
\end{aligned}
\qquad (3.172)
$$

The matrix ${}^{t+dt}_{t}B_{NL}$:

It is constructed in the way that

$$\begin{bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} \\ \frac{\partial w}{\partial y} \\ \frac{\partial w}{\partial z} \end{bmatrix} = {}^{t+dt}_{t}B_{NL} \begin{bmatrix} U_1 \\ V_1 \\ W_1 \\ \theta_1^x \\ \theta_1^y \\ \theta_1^z \\ U_2 \\ V_2 \\ W_2 \\ \theta_2^x \\ \theta_2^y \\ \theta_2^z \\ U_3 \\ V_3 \\ W_3 \\ \theta_3^x \\ \theta_3^y \\ \theta_3^z \end{bmatrix} = {}^{t+dt}_{t}B_{NL}\,\bar{U} \tag{3.173}$$

The detailed expressions for calculating ${}^{t+dt}_{t}B_{NL}$ are given in (3.176) and (3.177). The equations are important because they present the way, how spatial derivatives of all the displacements are calculated. The entries in ${}^{t+dt}_{t}B_{NL}$ are thus used to setup also the matrix ${}^{t+dt}_{t}B_{L0}$ and ${}^{t+dt}_{t}B_{L1}$.

These matrices are computed as follows:

$$\begin{aligned}
{}^{t+dt}_{t}B_{L0(1,i)} &= {}^{t+dt}_{t}B_{NL(1,i)} \\
{}^{t+dt}_{t}B_{L0(2,i)} &= {}^{t+dt}_{t}B_{NL(5,i)} \\
{}^{t+dt}_{t}B_{L0(3,i)} &= {}^{t+dt}_{t}B_{NL(9,i)} \\
{}^{t+dt}_{t}B_{L0(4,i)} &= {}^{t+dt}_{t}B_{NL(4,i)} + {}^{t+dt}_{t}B_{NL(2,i)} \\
{}^{t+dt}_{t}B_{L0(5,i)} &= {}^{t+dt}_{t}B_{NL(6,i)} + {}^{t+dt}_{t}B_{NL(8,i)} \\
{}^{t+dt}_{t}B_{L0(6,i)} &= {}^{t+dt}_{t}B_{NL(7,i)} + {}^{t+dt}_{t}B_{NL(3,i)}
\end{aligned} \tag{3.174}$$

$$
\begin{aligned}
{}^{t+dt}_{t}B_{L1(1,i)} &= \frac{\partial^{t+dt}u}{\partial x}\,{}^{t+dt}_{t}B_{NL(1,i)} + \frac{\partial^{t+dt}v}{\partial x}\,{}^{t+dt}_{t}B_{NL(4,i)} + \frac{\partial^{t+dt}w}{\partial x}\,{}^{t+dt}_{t}B_{NL(7,i)} \\
{}^{t+dt}_{t}B_{L1(2,i)} &= \frac{\partial^{t+dt}u}{\partial y}\,{}^{t+dt}_{t}B_{NL(2,i)} + \frac{\partial^{t+dt}v}{\partial y}\,{}^{t+dt}_{t}B_{NL(5,i)} + \frac{\partial^{t+dt}w}{\partial y}\,{}^{t+dt}_{t}B_{NL(8,i)} \\
{}^{t+dt}_{t}B_{L1(3,i)} &= \frac{\partial^{t+dt}u}{\partial z}\,{}^{t+dt}_{t}B_{NL(3,i)} + \frac{\partial^{t+dt}v}{\partial z}\,{}^{t+dt}_{t}B_{NL(6,i)} + \frac{\partial^{t+dt}w}{\partial z}\,{}^{t+dt}_{t}B_{NL(9i)} \\
{}^{t+dt}_{t}B_{L1(4,i)} &= \frac{\partial^{t+dt}u}{\partial x}\,{}^{t+dt}_{t}B_{NL(2,i)} + \frac{\partial^{t+dt}u}{\partial y}\,{}^{t+dt}_{t}B_{NL(1,i)} + \frac{\partial^{t+dt}v}{\partial x}\,{}^{t+dt}_{t}B_{NL(5,i)} + \\
&\quad \frac{\partial^{t+dt}v}{\partial y}\,{}^{t+dt}_{t}B_{NL(4,i)} + \frac{\partial^{t+dt}w}{\partial x}\,{}^{t+dt}_{t}B_{NL(8,i)} + \frac{\partial^{t+dt}w}{\partial y}\,{}^{t+dt}_{t}B_{NL(7,i)} \\
{}^{t+dt}_{t}B_{L1(5,i)} &= \frac{\partial^{t+dt}u}{\partial y}\,{}^{t+dt}_{t}B_{NL(3,i)} + \frac{\partial^{t+dt}u}{\partial z}\,{}^{t+dt}_{t}B_{NL(2,i)} + \frac{\partial^{t+dt}v}{\partial y}\,{}^{t+dt}_{t}B_{NL(6,i)} + \\
&\quad \frac{\partial^{t+dt}v}{\partial z}\,{}^{t+dt}_{t}B_{NL(5,i)} + \frac{\partial^{t+dt}w}{\partial y}\,{}^{t+dt}_{t}B_{NL(9,i)} + \frac{\partial^{t+dt}w}{\partial y}z\,{}^{t+dt}_{t}B_{NL(8,i)} \\
{}^{t+dt}_{t}B_{L1(6,i)} &= \frac{\partial^{t+dt}u}{\partial z}\,{}^{t+dt}_{t}B_{NL(1,i)} + \frac{\partial^{t+dt}u}{\partial x}\,{}^{t+dt}_{t}B_{NL(3,i)} + \frac{\partial^{t+dt}v}{\partial z}\,{}^{t+dt}_{t}B_{NL(4,i)} + \\
&\quad \frac{\partial^{t+dt}v}{\partial x}\,{}^{t+dt}_{t}B_{NL(6,i)} + \frac{\partial^{t+dt}w}{\partial z}\,{}^{t+dt}_{t}B_{NL(7,i)} + \frac{\partial^{t+dt}w}{\partial x}\,{}^{t+dt}_{t}B_{NL(9,i)}
\end{aligned}
\tag{3.175}
$$

$$
\begin{aligned}
{}^{t+dt}_{t}B_{NL(1,1)} &= J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\\
{}^{t+dt}_{t}B_{NL(1,2)} &= 0\\
{}^{t+dt}_{t}B_{NL(1,3)} &= 0\\
{}^{t+dt}_{t}B_{NL(1,4)} &= 0\\
{}^{t+dt}_{t}B_{NL(1,5)} &= J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}+\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_z}\right)+\frac{1}{2}J_{1,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_z}+\frac{1}{2}J_{1,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}\\
{}^{t+dt}_{t}B_{NL(1,6)} &= J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}-\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_y}\right)-\frac{1}{2}J_{1,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_y}+\frac{1}{2}J_{1,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}\\
{}^{t+dt}_{t}B_{NL(2,1)} &= J_{2,1}^{-1}\frac{\partial h_i}{\partial r}\\
{}^{t+dt}_{t}B_{NL(2,2)} &= 0\\
{}^{t+dt}_{t}B_{NL(2,3)} &= 0\\
{}^{t+dt}_{t}B_{NL(2,4)} &= 0\\
{}^{t+dt}_{t}B_{NL(2,5)} &= J_{2,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}+\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_z}\right)+\frac{1}{2}J_{2,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_z}+\frac{1}{2}J_{2,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}\\
{}^{t+dt}_{t}B_{NL(2,6)} &= J_{2,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}-\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_y}\right)-\frac{1}{2}J_{2,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_y}+\frac{1}{2}J_{2,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}\\
{}^{t+dt}_{t}B_{NL(3,1)} &= J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\\
{}^{t+dt}_{t}B_{NL(3,2)} &= 0\\
{}^{t+dt}_{t}B_{NL(3,3)} &= 0\\
{}^{t+dt}_{t}B_{NL(3,4)} &= 0\\
{}^{t+dt}_{t}B_{NL(3,5)} &= J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}+\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_z}\right)+\frac{1}{2}J_{3,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_z}+\frac{1}{2}J_{3,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}\\
{}^{t+dt}_{t}B_{NL(3,6)} &= J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}-\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_y}\right)-\frac{1}{2}J_{3,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_y}+\frac{1}{2}J_{3,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}\\
{}^{t+dt}_{t}B_{NL(4,1)} &= 0\\
{}^{t+dt}_{t}B_{NL(4,2)} &= J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\\
{}^{t+dt}_{t}B_{NL(4,3)} &= 0\\
{}^{t+dt}_{t}B_{NL(4,4)} &= J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}-\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right)-\frac{1}{2}J_{1,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_z}-\frac{1}{2}J_{1,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}\\
{}^{t+dt}_{t}B_{NL(4,5)} &= 0\\
{}^{t+dt}_{t}B_{NL(4,6)} &= J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_x}+\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right)+\frac{1}{2}J_{1,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_x}+\frac{1}{2}J_{1,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_x}\\
{}^{t+dt}_{t}B_{NL(5,1)} &= 0\\
{}^{t+dt}_{t}B_{NL(5,2)} &= J_{2,1}^{-1}\frac{\partial h_i}{\partial r}\\
{}^{t+dt}_{t}B_{NL(5,3)} &= 0\\
{}^{t+dt}_{t}B_{NL(5,4)} &= J_{2,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}-\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right)-\frac{1}{2}J_{2,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_z}-\frac{1}{2}J_{2,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}\\
{}^{t+dt}_{t}B_{NL(5,5)} &= 0\\
{}^{t+dt}_{t}B_{NL(5,6)} &= J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_x}+\frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right)+\frac{1}{2}J_{1,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_x}+\frac{1}{2}J_{1,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_x}
\end{aligned}
\tag{3.176}
$$

$$
\begin{aligned}
&{}^{t+dt}_{t}B_{NL(6,1)} = 0\\
&{}^{t+dt}_{t}B_{NL(6,2)} = J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\\
&{}^{t+dt}_{t}B_{NL(6,3)} = 0\\
&{}^{t+dt}_{t}B_{NL(6,4)} = J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_z} - \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right) - \frac{1}{2}J_{3,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_z} - \frac{1}{2}J_{3,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_z}\\
&{}^{t+dt}_{t}B_{NL(6,5)} = 0\\
&{}^{t+dt}_{t}B_{NL(6,6)} = J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_x} + \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right) + \frac{1}{2}J_{3,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_x} + \frac{1}{2}J_{3,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_x}\\
&{}^{t+dt}_{t}B_{NL(7,1)} = 0\\
&{}^{t+dt}_{t}B_{NL(7,2)} = 0\\
&{}^{t+dt}_{t}B_{NL(7,3)} = J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\\
&{}^{t+dt}_{t}B_{NL(7,4)} = J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_y} + \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_y}\right) + \frac{1}{2}J_{1,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_y} + \frac{1}{2}J_{1,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}\\
&{}^{t+dt}_{t}B_{NL(7,5)} = J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_x} - \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right) - \frac{1}{2}J_{1,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_x} - \frac{1}{2}J_{1,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_x}\\
&{}^{t+dt}_{t}B_{NL(7,6)} = 0\\
&{}^{t+dt}_{t}B_{NL(8,1)} = 0\\
&{}^{t+dt}_{t}B_{NL(8,2)} = 0\\
&{}^{t+dt}_{t}B_{NL(8,3)} = J_{1,1}^{-1}\frac{\partial h_i}{\partial r}\\
&{}^{t+dt}_{t}B_{NL(8,4)} = J_{2,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_y} + \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_y}\right) + \frac{1}{2}J_{2,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_y} + \frac{1}{2}J_{2,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}\\
&{}^{t+dt}_{t}B_{NL(8,5)} = J_{2,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_x} - \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right) - \frac{1}{2}J_{2,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_x} - \frac{1}{2}J_{2,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_x}\\
&{}^{t+dt}_{t}B_{NL(8,6)} = 0\\
&{}^{t+dt}_{t}B_{NL(9,1)} = 0\\
&{}^{t+dt}_{t}B_{NL(9,2)} = 0\\
&{}^{t+dt}_{t}B_{NL(9,3)} = J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\\
&{}^{t+dt}_{t}B_{NL(9,4)} = J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\left(\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_y} + \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_y}\right) + \frac{1}{2}J_{3,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_y} + \frac{1}{2}J_{3,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_y}\\
&{}^{t+dt}_{t}B_{NL(9,5)} = J_{3,1}^{-1}\frac{\partial h_i}{\partial r}\left(-\frac{t}{2}\,{}^{t}a_i^{\,t+dt}V_i^{t_x} - \frac{s}{2}\,{}^{t}b_i^{\,t+dt}V_i^{s_x}\right) - \frac{1}{2}J_{3,2}^{-1}\,h_i\,{}^{t}b_i^{\,t+dt}V_i^{s_x} - \frac{1}{2}J_{3,3}^{-1}\frac{\partial h_i}{\partial r}\,{}^{t}a_i^{\,t+dt}V_i^{t_x}\\
&{}^{t+dt}_{t}B_{NL(9,6)} = 0
\end{aligned}
\tag{3.177}
$$

The stress matrix ${}^{t+\Delta t}_{t}\boldsymbol{S}_{ij}$ from **(1.34)** has he form:

$$
{}^{t+\Delta t}_{t}\boldsymbol{S}_{ij} = \begin{bmatrix}
{}^{t+\Delta t}_{t}\sigma_{xx} & {}^{t+\Delta t}_{t}\tau_{xy} & {}^{t+\Delta t}_{t}\tau_{xz} & & & & & & \\
 & {}^{t+\Delta t}_{t}\sigma_{yy} & {}^{t+\Delta t}_{t}\tau_{yz} & & & & & & \\
 & & {}^{t+\Delta t}_{t}\sigma_{zz} & & & & & & \\
 & & & {}^{t+\Delta t}_{t}\sigma_{xx} & {}^{t+\Delta t}_{t}\tau_{xy} & {}^{t+\Delta t}_{t}\tau_{xz} & & & \\
 & & & & {}^{t+\Delta t}_{t}\sigma_{yy} & {}^{t+\Delta t}_{t}\tau_{yz} & & & \\
 & & & & & {}^{t+\Delta t}_{t}\sigma_{zz} & & & \\
 & & & sym. & & & {}^{t+\Delta t}_{t}\sigma_{xx} & {}^{t+\Delta t}_{t}\tau_{xy} & {}^{t+\Delta t}_{t}\tau_{xz} \\
 & & & & & & & {}^{t+\Delta t}_{t}\sigma_{yy} & {}^{t+\Delta t}_{t}\tau_{yz} \\
 & & & & & & & & {}^{t+\Delta t}_{t}\sigma_{zz}
\end{bmatrix} \tag{3.178}
$$

As already mentioned, stress-strain relations are calculated in *r,s,t* coordinate system, hence we need equations for their transformations from global *x,y,z* coordinate system to the isoparametric system with *r,s,t* coordinates and vice versa.

Let us denote ${}^{t+dt}T_{\varepsilon}$, ${}^{t+dt}T_{\sigma}$ transformation matrices for strain and stress transformation from global to isoparametric coordinate system, so that:

$$
\begin{bmatrix} {}^{t+\Delta t}_{t}\sigma_{rr} \\ {}^{t+\Delta t}_{t}\tau_{rs} \\ {}^{t+\Delta t}_{t}\tau_{rt} \end{bmatrix} = {}^{t+dt}T_{\sigma} \begin{bmatrix} {}^{t+\Delta t}_{t}\sigma_{xx} \\ {}^{t+\Delta t}_{t}\sigma_{yy} \\ {}^{t+\Delta t}_{t}\sigma_{zz} \\ {}^{t+\Delta t}_{t}\tau_{xy} \\ {}^{t+\Delta t}_{t}\tau_{yz} \\ {}^{t+\Delta t}_{t}\tau_{xz} \end{bmatrix}
$$

$$
\begin{bmatrix} {}^{t+\Delta t}_{t}\varepsilon_{rr} \\ {}^{t+\Delta t}_{t}\gamma_{rs} \\ {}^{t+\Delta t}_{t}\gamma_{rt} \end{bmatrix} = {}^{t+dt}T_{\varepsilon} \begin{bmatrix} {}^{t+\Delta t}_{t}\varepsilon_{xx} \\ {}^{t+\Delta t}_{t}\varepsilon_{yy} \\ {}^{t+\Delta t}_{t}\varepsilon_{zz} \\ {}^{t+\Delta t}_{t}\gamma_{xy} \\ {}^{t+\Delta t}_{t}\gamma_{yz} \\ {}^{t+\Delta t}_{t}\gamma_{xz} \end{bmatrix} \tag{3.179}
$$

Then the transformation matrices are calculated by:

$$
{}^{t+dt}T_{\sigma} = \begin{bmatrix}
\left({}^{t+dt}V^{r_x}\right)^2 & \left({}^{t+dt}V^{r_y}\right)^2 & \left({}^{t+dt}V^{r_z}\right)^2 & 2\,{}^{t+dt}V^{r_x}\,{}^{t+dt}V^{r_y} \\
{}^{t+dt}V^{r_x}\,{}^{t+dt}V^{s_x} & {}^{t+dt}V^{r_y}\,{}^{t+dt}V^{s_y} & {}^{t+dt}V^{r_z}\,{}^{t+dt}V^{s_z} & {}^{t+dt}V^{r_x}\,{}^{t+dt}V^{s_y} + {}^{t+dt}V^{r_y}\,{}^{t+dt}V^{s_x} \\
{}^{t+dt}V^{r_x}\,{}^{t+dt}V^{t_x} & {}^{t+dt}V^{r_y}\,{}^{t+dt}V^{t_y} & {}^{t+dt}V^{r_z}\,{}^{t+dt}V^{s_z} & {}^{t+dt}V^{r_x}\,{}^{t+dt}V^{t_y} + {}^{t+dt}V^{r_y}\,{}^{t+dt}V^{t_x}
\end{bmatrix}
$$

$$\begin{bmatrix} \dots & 2^{t+dt}V^{r_y}\ {}^{t+dt}V^{r_z} & 2^{t+dt}V^{r_x}\ {}^{t+dt}V^{r_z} \\ \dots & {}^{t+dt}V^{r_y}\ {}^{t+dt}V^{s_z} + {}^{t+dt}V^{r_z}\ {}^{t+dt}V^{s_y} & {}^{t+dt}V^{r_x}\ {}^{t+dt}V^{s_z} + {}^{t+dt}V^{r_z}\ {}^{t+dt}V^{s_x} \\ \dots & {}^{t+dt}V^{r_y}\ {}^{t+dt}V^{t_z} + {}^{t+dt}V^{r_z}\ {}^{t+dt}V^{t_y} & {}^{t+dt}V^{r_x}\ {}^{t+dt}V^{t_z} + {}^{t+dt}V^{r_z}\ {}^{t+dt}V^{t_x} \end{bmatrix} \tag{3.180}$$

$${}^{t+dt}T_{\varepsilon} = \begin{bmatrix} \left({}^{t+dt}V^{r_x}\right)^2 & \left({}^{t+dt}V^{r_y}\right)^2 & \left({}^{t+dt}V^{r_z}\right)^2 \\ 2^{t+dt}V^{r_x}\ {}^{t+dt}V^{s_x} & 2^{t+dt}V^{r_y}\ {}^{t+dt}V^{s_y} & 2^{t+dt}V^{r_z}\ {}^{t+dt}V^{s_z} \\ 2^{t+dt}V^{r_x}\ {}^{t+dt}V^{t_x} & 2^{t+dt}V^{r_y}\ {}^{t+dt}V^{t_y} & 2^{t+dt}V^{r_z}\ {}^{t+dt}V^{s_z} \end{bmatrix}$$

$$\begin{bmatrix} \dots & {}^{t+dt}V^{r_y}\ {}^{t+dt}V^{r_z} & {}^{t+dt}V^{r_x}\ {}^{t+dt}V^{r_z} \\ \dots & {}^{t+dt}V^{r_y}\ {}^{t+dt}V^{s_z} + {}^{t+dt}V^{r_z}\ {}^{t+dt}V^{s_y} & {}^{t+dt}V^{r_x}\ {}^{t+dt}V^{s_z} + {}^{t+dt}V^{r_z}\ {}^{t+dt}V^{s_x} \\ \dots & {}^{t+dt}V^{r_y}\ {}^{t+dt}V^{t_z} + {}^{t+dt}V^{r_z}\ {}^{t+dt}V^{t_y} & {}^{t+dt}V^{r_x}\ {}^{t+dt}V^{t_z} + {}^{t+dt}V^{r_z}\ {}^{t+dt}V^{t_x} \end{bmatrix} \tag{3.181}$$

where vectors ${}^{t+dt}V^{s} = \begin{bmatrix} {}^{t+dt}V^{s_x} & {}^{t+dt}V^{s_y} & {}^{t+dt}V^{s_z} \end{bmatrix}$, ${}^{t+dt}V^{t} = \begin{bmatrix} {}^{t+dt}V^{t_x} & {}^{t+dt}V^{t_y} & {}^{t+dt}V^{t_z} \end{bmatrix}$ are vectors of unity length from Fig. 3-40. The remaining vector is calculated as a vector product of the previous two vectors:

$${}^{t+dt}V^{r} = \begin{bmatrix} {}^{t+dt}V^{r_x} & {}^{t+dt}V^{r_y} & {}^{t+dt}V^{r_z} \end{bmatrix} = {}^{t+dt}V^{s} \otimes {}^{t+dt}V^{t} \tag{3.182}$$

Inverse transformation matrices are calculated as:

$$\begin{aligned} {}^{t+dt}T_{\sigma}^{-1} &= {}^{t+dt}T_{\varepsilon}^{T} \\ {}^{t+dt}T_{\varepsilon}^{-1} &= {}^{t+dt}T_{\sigma}^{T} \end{aligned} \tag{3.183}$$

3.17.4 The Element Integration

The element is integrated numerically. Along its longitudinal axis the element is integrated by standard two to six nodes Gaussian integration. The table below lists *r* coordinates and associated weights for utilized integration points:

Table 3-6: Gaussian integration of the beam element along the longitudinal axis

Number of integ. points	Integration point	Coordinate *r*	Weight
2	1	0.577350269189626	1.
	2	-0.577350269189626	1.

3	1	0.774596669241483	0.555555555555556
	2	0.	0.888888888888889
	3	-0.774596669241483	0.555555555555556
4	1	0.861136311594053	0.347854845137454
	2	0.339981043584856	0.652145154862546
	3	-0.339981043584856	0.652145154862546
	4	0.861136311594053	0.347854845137454
5	1	0.906179845938664	0.236926885056189
	2	0.538469310105683	0.478628670499366
	3	0.	0.568888888888889
	4	-0.538469310105683	0.478628670499366
	5	-0.906179845938664	0.236926885056189
6	1	0.932469514203152	0.171324492379170
	2	0.661209386466265	0.360761573048139
	3	0.238619186083197	0.467913934572691
	4	-0.238619186083197	0.467913934572691
	5	-0.661209386466265	0.360761573048139
	6	-0.932469514203152	0.171324492379170

In most cases the 2-nodes integration should be sufficient, for a higher order integration schemes oscillatory shear stresses and forces may be observed along the length of the beam.

As for integration within the cross-section, i.e. in s,*t* coordinates, trapezoidal quadrature is used. The element cross-section is subdivided into n_s, n_t "strips" as depicted in the following figure.

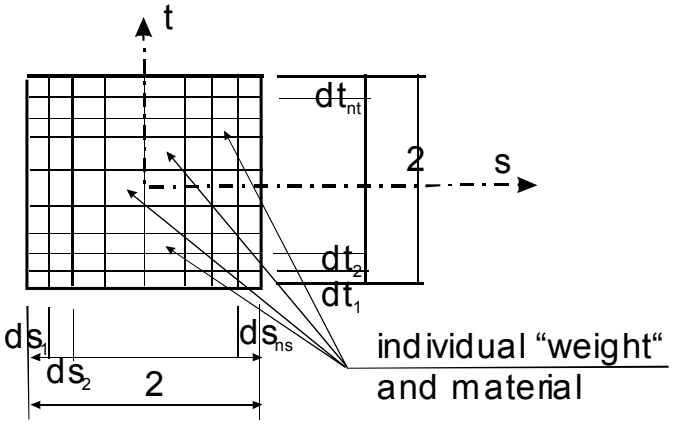


Fig. 3-41 The beam cross section integration

The integration is then carried out by summing functional values in center of all quadrilaterals multiplied by their area.

Note that the element is integrated within the isoparametric coordinate system, hence we have to use $dx\,dy\,dz = \det(J)\,dr\,ds\,dt$, see (3.171).

Nice feature of the ATENA's implementation of the beam is that each of the quadrilaterals in a cross section adopts an artificial input weight factor. By default, such a "weight" is equal to one, however, if we set its value to zero, essentially a hole is introduced. This mechanism, together with possibility of defining a customized material law in each of the quadrilaterals facilitates to analyze beams that have a arbitrary shape of cross-sections.

The present beam implementation supports also smeared reinforcement. This is done in the same way as it was for the Ahmad elements described in the previous section.

3.18 Curvilinear Nonlinear 3D Isoparametric Beam Element

CCIsoBeamBrick20_3, CCIsoBeamBrick12_3D and CCIsoBeamBrick8_3D are beam curved isoparametric elements similar to the previous CCBeamNL_3D element. They use similar geometry, node numbering etc., but differ from CCBeamNL_3D in that they account for all 6 components of 3D strains and stress vectors. They comply with all 3D static equations and no additional static or kinematic constrains are imposed. The comparison of CCBeamNL_3D vs. CCIsoBeamBrick12_3D resembles that of CCAhmad vs. CCIsoShell elements described above. The CCIsoBeamBrick20_3, CCIsoBeamBrick12_3D and CCIsoBeamBrick8_3D are easy to use, they preserve their 3D volume and they are nicely visualized during pre and post processing. They can be input, loaded, and output in the same way as CCIsoBrick hexahedral elements. CCIsoBeam8_3D features linear geometry and displacement approximation, (i.e. it has nodes 1...8, see the figure below), whilst CCIsoBeam12_3D has reduced quadratic approximation, (i.e. it has nodes 1...12). CCIsoBeam20_3D comprises 20 nodes as shown in the sub-figure "Brick nodes" below and it has full serendipity displacements approximation.

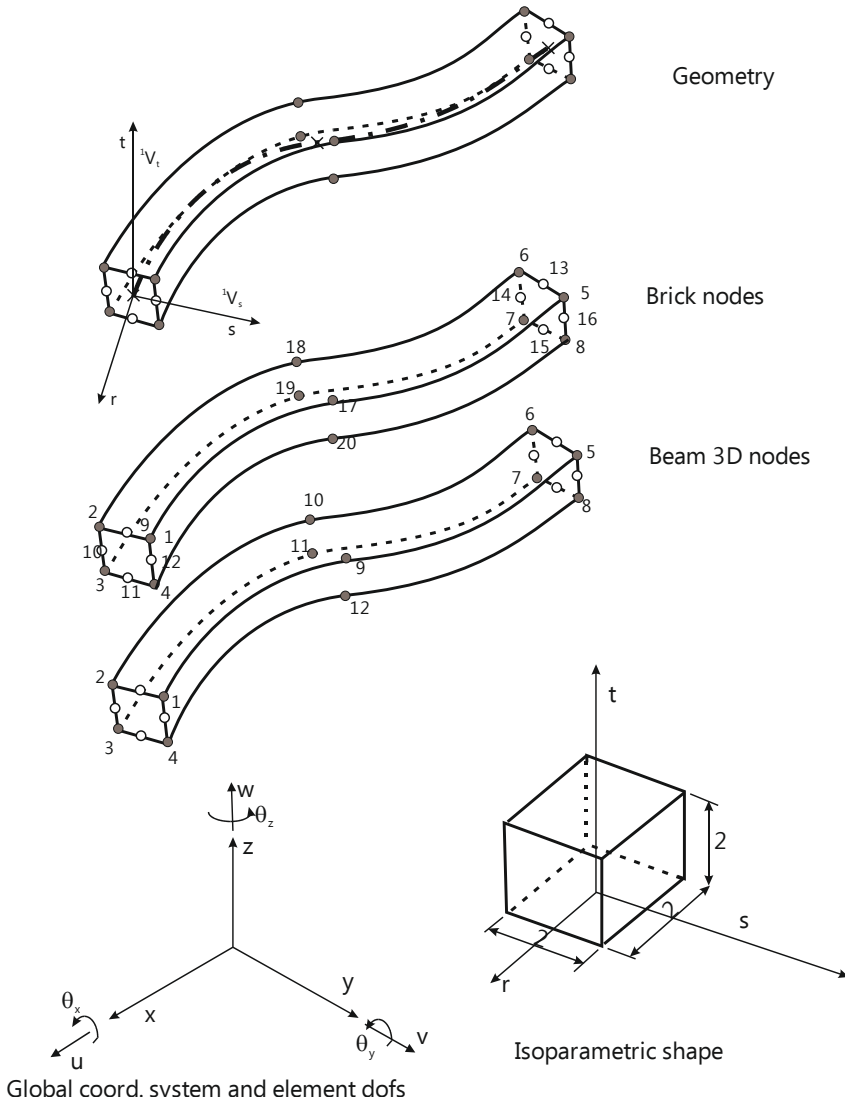


Fig. 3-42 CCIsoBeamBrick12_3D and CCIsoBeamBrick8_3D elements

Shape of cross section can be any quadrilateral, i.e. it need not be only a rectangle as depicted above. The elements are particularly useful for analyses of structures, where beam elements must be combined with 3D solid and/or shell elements.

Derivation of the element is much the same as that for CCIsoShell element, i.e. Equations (3.143) and (3.145) thru (3.165) remain valid. Geometry and displacement approximation (3.144) is replaced by:

$$
\begin{aligned}
{}^{t}x_i &= h_k\left(\frac{1+s}{2}\,{}^{t}X_i^{k,front}+\frac{1-s}{2}\,{}^{t}X_i^{k,back}\right)\left(\frac{1+t}{2}\,{}^{t}X_i^{k,top}+\frac{1-t}{2}\,{}^{t}X_i^{k,bot}\right)\\
{}^{t+\Delta t}x_i^{(i-1)} &= h_k\left(\frac{1+s}{2}\,{}^{t+\Delta t}X_i^{k,front(i-1)}+\frac{1-s}{2}\,{}^{t+\Delta t}X_i^{k,back(i-1)}\right)\left(\frac{1+t}{2}\,{}^{t+\Delta t}X_i^{k,top(i-1)}+\frac{1-t}{2}\,{}^{t+\Delta t}X_i^{k,bot(i-1)}\right) \qquad (3.184)\\
{}^{t+\Delta t}x_i^{(i)} &= h_k\left(\frac{1+ds}{2}\,{}^{t+\Delta t}X_i^{k,front(i)}+\frac{1-s}{2}\,{}^{t+\Delta t}X_i^{k,back(i)}\right)\left(\frac{1+t}{2}\,{}^{t+\Delta t}X_i^{k,top(i)}+\frac{1-t}{2}\,{}^{t+\Delta t}X_i^{k,bot(i)}\right)
\end{aligned}
$$

$h_k = h_k(r)$ are 1D interpolation functions, see the interpolation function for CCIsoTruss elements. The same notation is used for CCIsoShell Elements.

The element is calculated in integration points, (i.e. material points) that are located similar to CCBeamNL_3D elements, refer to Fig. 3-41. The element can use any 3D material model. Different materials can be specified for each material points, (or points in cross section). Some of them can be used for modelling of embedded reinforcement. (Btw. discrete reinforcement can be employed, too). The elements support both material and geometric nonlinearity.

3.19 Curvilinear Nonlinear 1D element

The elements CCIsoBeamBar<xx> and CCIsoBeamBar<xxx> are from the point of view of mechanics nearly identical to the element described in Section 3.13, the difference being only in that that these elements are specified by their axis as 1D beams. The first element has 2 nodes (and uses linear interpolation of its geometry and displacements). The latter element has 3 nodes (and uses quadratic interpolation of its geometry and displacements, which is identical to CCBeamNL element referred above). The elements can be curved and can have variable height, width and orientation of their cross section. All these parameters are input in CCBeam1D geometry in form of algebraic expressions. The expression are functions of beam's coordinates x,y,z. Similar to CCBeamNL element, these elements are also integrated by Gauss integration along the beam's axis while grid quadrature is used for the remaining 2 directions (within cross sections). The elements support embedded reinforcements, holes different materials in different integration points etc. in the same way as it is the case of CCBeamNL element. They are suitable for modeling of both shallow and deep beams. Note that CCIsoBeamBar<xx> has far worse properties compared to CCIsoBeamBar<xxx>. Hence, the linear element should be used only to model some links and connections within the structures.

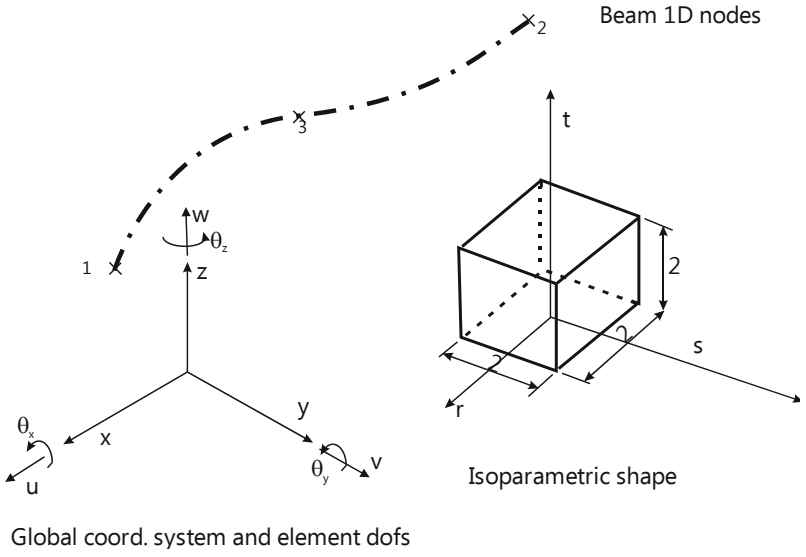


Fig. 3-43 CCIsoBeamNLBar<xxx> element

3.19.1 Connection of the beam1D to an ambient solid element

The procedure to connect beam1D's dofs to an ambient element is like that for shell2D elements, see 3.13.2. Again, it consists of two parts:

1. fix a FE node with $[u,v,w]$ displacement within the beam1D element,

2. fix three rotation dofs of the beam1D element within ambient elements

3.19.1.1 Fixing a FE node with $[u,v,w]$ displacement within the beam1D element

Using (3.169) and (3.170) write expression for beam1D displacements at the top u_i^{top} and bottom u_i^{bot} , i.e. $s=0, t=\pm 1$ of a cross section. Do the same for right u_i^{right} and left u_i^{left} point, i.e. $s=\pm 1, t=0$.

Write 3D solid approximation for the same 4 nodes. Then, if we compare the 1D and 3D approximation, after some mathematical manipulation we derive

$$
\begin{aligned}
hh_k^{14p58} &= hh_k(r,s=1,t=1)+hh_k(r,s=-1,t=1)+hh_k(r,s=1,t=-1)+hh_k(r,s=-1,t=-1)\\
hh_k^{14m58} &= hh_k(r,s=1,t=1)+hh_k(r,s=-1,t=1)-hh_k(r,s=1,t=-1)-hh_k(r,s=-1,t=-1)\\
hh_k^{15m48} &= hh_k(r,s=1,t=1)+hh_k(r,s=1,t=-1)-hh_k(r,s=-1,t=1)-hh_k(r,s=-1,t=-1)
\end{aligned}
$$

$$
\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \sum_k \begin{bmatrix} hh_k^{14p58} & 0 & 0 & 0 & \frac{a_k}{2} hh_k^{14m58} & -\frac{b_k}{2} hh_k^{15m48} \\ 0 & hh_k^{14p58} & 0 & -\frac{a_k}{2} hh_k^{14m58} & 0 & 0 \\ 0 & 0 & hh_k^{14p58} & hh_k^{15m48} & 0 & 0 \end{bmatrix} \begin{bmatrix} u_k \\ v_k \\ w_k \\ \theta_{x,k} \\ \theta_{y,k} \\ \theta_{z,k} \end{bmatrix} \tag{3.185}
$$

3.19.1.2 Fixing three rotation dofs of the beam3D element within ambient elements

Similarly, to the expressions for shell2D the resulting equations for beam1D rotation $\theta_x, \theta_y, \theta_z$ are

$$
\begin{bmatrix} \theta_x \\ \theta_y \\ \theta_z \end{bmatrix} = \mathbf{MM} \begin{bmatrix} UU_1^{top} & UU_2^{top} & UU_3^{top} & UU_1^{bot} & UU_2^{bot} & UU_3^{bot} & UU_1^{right} & UU_2^{right} & UU_3^{right} & UU_1^{left} & UU_2^{left} & UU_3^{left} \end{bmatrix}^T \tag{3.186}
$$

$$MM^T = \begin{bmatrix}
1/2\frac{Vr_xVs_x}{a} & -1/2\frac{Vs_xVr_y}{a}+\frac{Vr_xVs_y}{a} & -1/2\frac{Vs_xVr_z}{a}+\frac{Vr_xVs_z}{a} \\
-1/2\frac{Vr_xVs_y}{a}+\frac{Vs_xVr_y}{a} & 1/2\frac{Vr_yVs_y}{a} & -1/2\frac{Vs_yVr_z}{a}+\frac{Vr_yVs_z}{a} \\
-1/2\frac{Vr_xVs_z}{a}+\frac{Vs_xVr_z}{a} & -1/2\frac{Vr_yVs_z}{a}+\frac{Vs_yVr_z}{a} & 1/2\frac{Vr_zVs_z}{a} \\
-1/2\frac{Vr_xVs_x}{a} & 1/2\frac{Vs_xVr_y}{a}-\frac{Vr_xVs_y}{a} & 1/2\frac{Vs_xVr_z}{a}-\frac{Vr_xVs_z}{a} \\
1/2\frac{Vr_xVs_y}{a}-\frac{Vs_xVr_y}{a} & -1/2\frac{Vr_yVs_y}{a} & 1/2\frac{Vs_yVr_z}{a}-\frac{Vr_yVs_z}{a} \\
1/2\frac{Vr_xVs_z}{a}-\frac{Vs_xVr_z}{a} & 1/2\frac{Vr_yVs_z}{a}-\frac{Vs_yVr_z}{a} & -1/2\frac{Vr_zVs_z}{a} \\
-1/2\frac{Vr_xVt_x}{b} & 1/2\frac{Vt_xVr_y}{b}-\frac{Vr_xVt_y}{b} & 1/2\frac{Vt_xVr_z}{b}-\frac{Vr_xVt_z}{b} \\
1/2\frac{Vr_xVt_y}{b}-\frac{Vt_xVr_y}{b} & -1/2\frac{Vr_yVt_y}{b} & 1/2\frac{Vt_yVr_z}{b}-\frac{Vr_yVt_z}{b} \\
1/2\frac{Vr_xVt_z}{b}-\frac{Vt_xVr_z}{b} & 1/2\frac{Vr_yVt_z}{b}-\frac{Vt_yVr_z}{b} & -1/2\frac{Vr_zVt_z}{b} \\
1/2\frac{Vr_xVt_x}{b} & -1/2\frac{Vt_xVr_y}{b}+\frac{Vr_xVt_y}{b} & -1/2\frac{Vt_xVr_z}{b}+\frac{Vr_xVt_z}{b} \\
-1/2\frac{Vr_xVt_y}{b}+\frac{Vt_xVr_y}{b} & 1/2\frac{Vr_yVt_y}{b} & -1/2\frac{Vt_yVr_z}{b}+\frac{Vr_yVt_z}{b} \\
-1/2\frac{Vr_xVt_z}{b}+\frac{Vt_xVr_z}{b} & -1/2\frac{Vr_yVt_z}{b}+\frac{Vt_yVr_z}{b} & 1/2\frac{Vr_zVt_z}{b}
\end{bmatrix} \tag{3.187}$$

where $Vr_{k,i} = V_i^{r_k} \quad Vs_{k,i} = V_i^{s_k}$, $a = a_k, b = b_k$

Note that displacement dofs are fixed by(3.185).

If either bottom or top node gets outside the ambient element, the middle point is used instead. Equations (3.186) and (3.187) are still valid but it is necessary to use $\tilde{MM}(i,j) = \frac{1}{2}MM(i,j),\ j = 1..6,\ \tilde{MM}(i,j) = MM(i,j),\ j = 7..12$ to calculate $\begin{bmatrix}\theta_x & \theta_y & \theta_z\end{bmatrix}^T$.

Similarly, if either right or bottom node gets outside the ambient element, the middle point is used instead. Then, it is necessary to use $\tilde{MM}(i,j) = MM(i,j),\ j = 1..6,\ \tilde{MM}(i,j) = \frac{1}{2}MM(i,j),\ j = 7..12$ to calculate $\begin{bmatrix}\theta_x & \theta_y & \theta_z\end{bmatrix}^T$.

3.20 Integrated forces and moments for shells

Integrated forces for shells are computed as follows:

$$
\begin{aligned}
N_{x'} &= \int_{-t/2}^{t/2} \sigma_{x'x'}\, dz' \\
N_{y'} &= \int_{-t/2}^{t/2} \sigma_{y'y'}\, dz' \\
N_{z'} &= \int_{-t/2}^{t/2} \sigma_{z'z'}\, dz' \\
Q_{x'z'} &= \int_{-t/2}^{t/2} \tau_{x'z'}\, dz' \\
Q_{x'y'} &= \int_{-t/2}^{t/2} \tau_{x'y'}\, dz' \\
Q_{y'z'} &= \int_{-t/2}^{t/2} \tau_{y'z'}\, dz' \\
M_{y'} &= \int_{-t/2}^{t/2} \sigma_{x'x'}\, z\, dz' \\
M_{x'} &= \int_{-t/2}^{t/2} \sigma_{y'y'}(-z)\, dz' \\
K_{x'y'} &= \int_{-t/2}^{t/2} \tau_{x'y'}(-z)\, dz'
\end{aligned}
\tag{3.188}
$$

The above forces and moments act on planes indicated below:

$$
\begin{aligned}
&y'z': \quad \left[N_{x'} \quad Q_{x'y'} \quad Q_{x'z'} \qquad\qquad K_{x'y'} \quad M_{y'} \quad K_{x'z'} = 0 \right] \\
&x'z': \quad \left[Q_{y'x'} = Q_{x'y'} \quad N_{y'} \quad Q_{y'z'} \qquad\qquad M_{x'} \quad K_{y'x'} = -K_{x'y'} \quad K_{y'z'} = 0 \right] \\
&x'y': \quad \left[Q_{z'x'} = Q_{x'z'} \quad Q_{z'y'} = Q_{y'z'} \quad N_{z'} \quad K_{z'x'} = 0 \quad K_{z'y'} = 0 \quad M_{z'} = 0 \right]
\end{aligned}
$$

The actual values of the forces and moments are calculated by extrapolation of stresses from IPs into finite element nodes, (please refer to Section "Extrapolation of Stress and Strain to Element Nodes" in Chapter CONTINUUM GOVERNING EQUATIONS. The process is as follows:

Let us take an example of $N_{x'}$ that is calculated by integration of $\sigma_{x'x'}$ thru element's thickness.

The stress $\underline{\sigma}_{x'x'}$ at element nodes is extrapolated from stresses in IPs $\hat{\sigma}_{x'x'}$ by

$$
\begin{aligned}
\underline{\sigma}_{x'x'} &= \left[M \right]^{inv} \underline{P}_{x'x'} \\
P_{xx,i} &= \int_{V_e} h_i\, \hat{\sigma}_{x'x'}\, dV_e \\
M_{ij} &= \int_{V_e} h_i\, h_j\, dV_e
\end{aligned}
\tag{3.189}
$$

where V_e stands for element volume. Using (3.188) and writing (3.189) for extrapolation within shell mid-plane Ω_e, (i.e. integration over Ω_e instead of V_e) we can write

$$
\begin{aligned}
N_{x'x'} &= \left[MM \right]^{inv} \underline{PP}_{x'x'} \\
PP_{xx,i} &= \int_{\Omega_e} \left(\int_{-t/2}^{t/2} \tilde{h}_i\, \hat{\sigma}_{x'x'} dz \right) d\Omega_e = \int_{V_e} \tilde{h}_i\, \hat{\sigma}_{x'x'}\, dV_e \\
MM_{ij} &= \int_{\Omega_e} \tilde{h}_i\, \tilde{h}_j\, d\Omega_e = \int_{V_e} \frac{1}{t} \tilde{h}_i\, \tilde{h}_j\, dV_e
\end{aligned}
\tag{3.190}
$$

where $t = t(r,s)$ is element thickness at *r,s*. The integration for extrapolation is carried out over Ω_e, because the forces and moments are the same through shell thickness. Note that $\tilde{h}_k = \tilde{h}_k(r,s)$ is interpolation function in the shell mid-plane and it is independent of *t* coordinate, (unlike $h_i = h_i(r,s,t)$ in (3.189)). Therefore, we can write, (see the last equation in (3.190):

$$\begin{aligned}
\int_{V_e} \tilde{h}_i \tilde{h}_j \, dV_e &= \int_{\Omega_e} \left(\int_{-t/2}^{t/2} \tilde{h}_i(r,s) \, \tilde{h}_j(r,s) dt \right) d\Omega_e \\
&= \int_{\Omega_e} \tilde{h}_i(r,s) \, \tilde{h}_j(r,s) \left(\int_{-t/2}^{t/2} dt \right) d\Omega_e \\
&= \int_{\Omega_e} \tilde{h}_i(r,s) \, \tilde{h}_j(r,s) \, t(r,s) \, d\Omega_e = \int_{\Omega_e} \tilde{h}_i \, \tilde{h}_j \, t \, d\Omega_e \\
\int_{V_e} \frac{1}{t} \tilde{h}_i \tilde{h}_j \, dV_e &= \int_{\Omega_e} \tilde{h}_i \, \tilde{h}_j \, d\Omega_e = MM_{ij}
\end{aligned} \tag{3.191}$$

3.21 Integrated forces and moments for beams

Integrated forces for beams are computed as follows:

$$\begin{aligned}
N_{x'} &= \int_{-t/2}^{t/2} \sigma_{x'x'} \, dy' dz' \\
Q_{x'y'} &= \int_{-t/2}^{t/2} \tau_{x'y'} \, dy' dz' \\
Q_{x'z'} &= \int_{-t/2}^{t/2} \tau_{x'z'} \, dy' dz' \\
K_{y'z'} &= \int_{-t/2}^{t/2} (\tau_{x'y'}(-z') + \tau_{x'z'} \, y') \, dy' dz' \\
M_{y'} &= \int_{-t/2}^{t/2} (\sigma_{x'x'} \, z') \, dy' dz' \\
M_{z'} &= \int_{-t/2}^{t/2} \sigma_{x'x'}(-y') \, dy' dz'
\end{aligned} \tag{3.192}$$

The forces and moments act on the plane (*x'y'*). They are calculated similar way to (3.190), however, $MM_{ij} = \int_{l_e} h_i \, h_j \, dr = \int_{V_e} \frac{1}{b\,h} h_i \, h_j \, dV_e$, where $b\,h$ is area of the beam's cross section and l_e is element length.

3.22 Global and Local Coordinate Systems for Element Load

Most element loads can be defined in global or local coordinate system. Global coordinate system is always available, hence using it is usually the safest way to input a desired element load. Nevertheless, some elements are internally defined in a local coordinate system and it can be employed for an element load definition, too. Location of such a local system, (if it exists) has been described together with description of the associated finite element. For example, local coordinate systems are defined for plane 3D isoparametric elements, shell, and beam elements etc. On the other hand, elements such as tetrahedrons, bricks and others are defined in directly in

global coordinate system and therefore a local element load is treated as if it were input as a global element load.

An exception to the above are truss elements. Although they are defined in global coordinate system, they do support local element load. Their local coordinate system (for element loading only) is defined as follows:

- local X axis points in direction of the truss element,
- local Y axis is normal to local X axis and lies in the global XY plane,
- its positive orientation is chosen so that the local X and local Y forms a right-hand (2D) coordinate system in the plane defined by these local axes,
- local Z axis is vector product of the local X and local Y axes, (for 3D case only).
- if the truss is parallel to global z, then local X points in direction of global Z, local Y coincides with global Y and local Z has opposite direction of the global X, (for 3D case only).

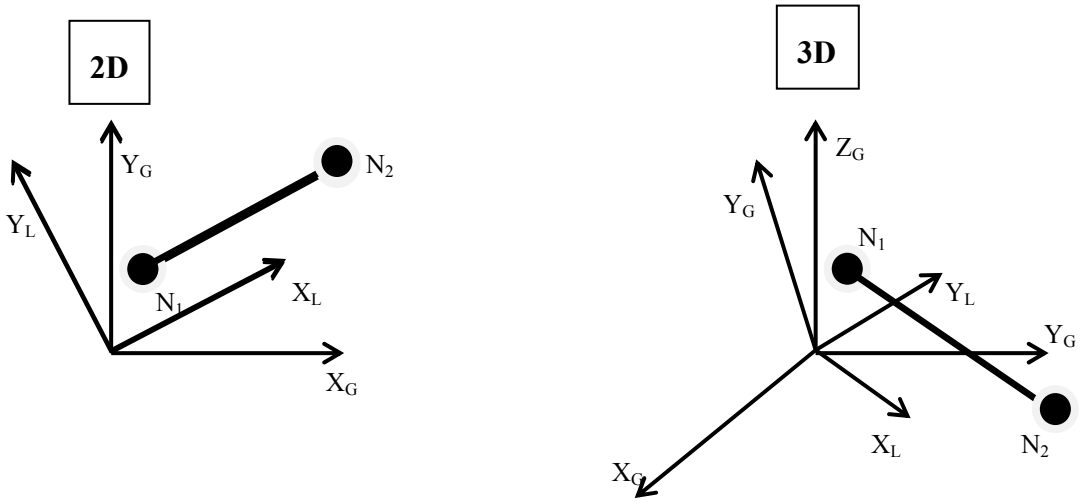


Fig. 3-44 Local and global coordinate systems for truss element N1-N2, (e.g. loaded element edge)

Specification of a boundary load deserves slightly more attention. Firstly, it is applied only to an element's edge or an element's surface, (see also the note below), as opposed to e.g. an element body load that is for the whole element. Local coordinate system is thus defined by location of the loaded edge or surface. Secondly, a boundary load definition must include a reference to a selection, which contains nodes to be loaded. Their order in the list is irrelevant, as what really matters is the order in which they appear in the element incidences. When processing a boundary load, ATENA loops thru all element's surfaces and edges, (in the order specified in the table below) and checks appropriate incidental nodes. If the tested node is present in the list of loaded boundary nodes, it is picked up and put into incidences of a new planar or line element. This element is later used to process the boundary load. It is its local coordinate system, that is (possibly) used to deal with local/global load transformations.

The table below defines the orders, in which element surfaces and edges are tested for a surface or edge element load. (It is assumed that element incidences are $(n_1, n_2, \ldots n_{num\_elem\_nodes})$). It describes linear elements, but surfaces and edges of nonlinear elements are treated in the same order.

Table 3-7: Order of element surface and nodes as they are tested within a boundary load definition.

Element shape	Type	Surface/node incidences
Truss	Edge	(n_1, n_2)
Triangle	Surface	(n_1, n_2, n_3)
	Edge	(n_1, n_2); (n_2, n_3); (n_3, n_1)
Quad	Surface	(n_1, n_2, n_3, n_4)
	Edge	(n_1, n_2); (n_2, n_3); (n_3, n_1); (n_4, n_1)
Hexahedron, (brick)	Surface	(n_1, n_2, n_3, n_4); (n_5, n_6, n_7, n_8); (n_1, n_4, n_8, n_5); (n_2, n_3, n_7, n_6); (n_1, n_2, n_6, n_5); (n_4, n_3, n_7, n_8);
	Edge	(n_1, n_2); (n_2, n_3); (n_3, n_4); (n_4, n_1); (n_5, n_6); (n_6, n_7); (n_7, n_8); (n_8, n_5); (n_1, n_5); (n_2, n_6); (n_3, n_7); (n_4, n_8)
Tetrahedron	Surface	(n_1, n_2, n_3); (n_1, n_2, n_4); (n_1, n_3, n_4); (n_2, n_3, n_4)
	Edge	(n_1, n_2); (n_2, n_3); (n_3, n_1); (n_4, n_1); (n_4, n_2); (n_4, n_3)
Wedge	Surface	(n_1, n_2, n_3); (n_4, n_5, n_6); (n_1, n_2, n_5, n_4); (n_6, n_5, n_2, n_3); (n_4, n_6, n_3, n_1)
	Edge	(n_1, n_2); (n_2, n_3); (n_3, n_1); (n_4, n_5); (n_5, n_6); (n_6, n_4); (n_1, n_4); (n_2, n_5); (n_3, n_6);

Note that only one surface or one edge of each element can be loaded in a single boundary load specification. If more element's surfaces or edges are to be loaded, use more boundary load definitions. Violation of this rule causes an error report and skipping of the offending boundary load.

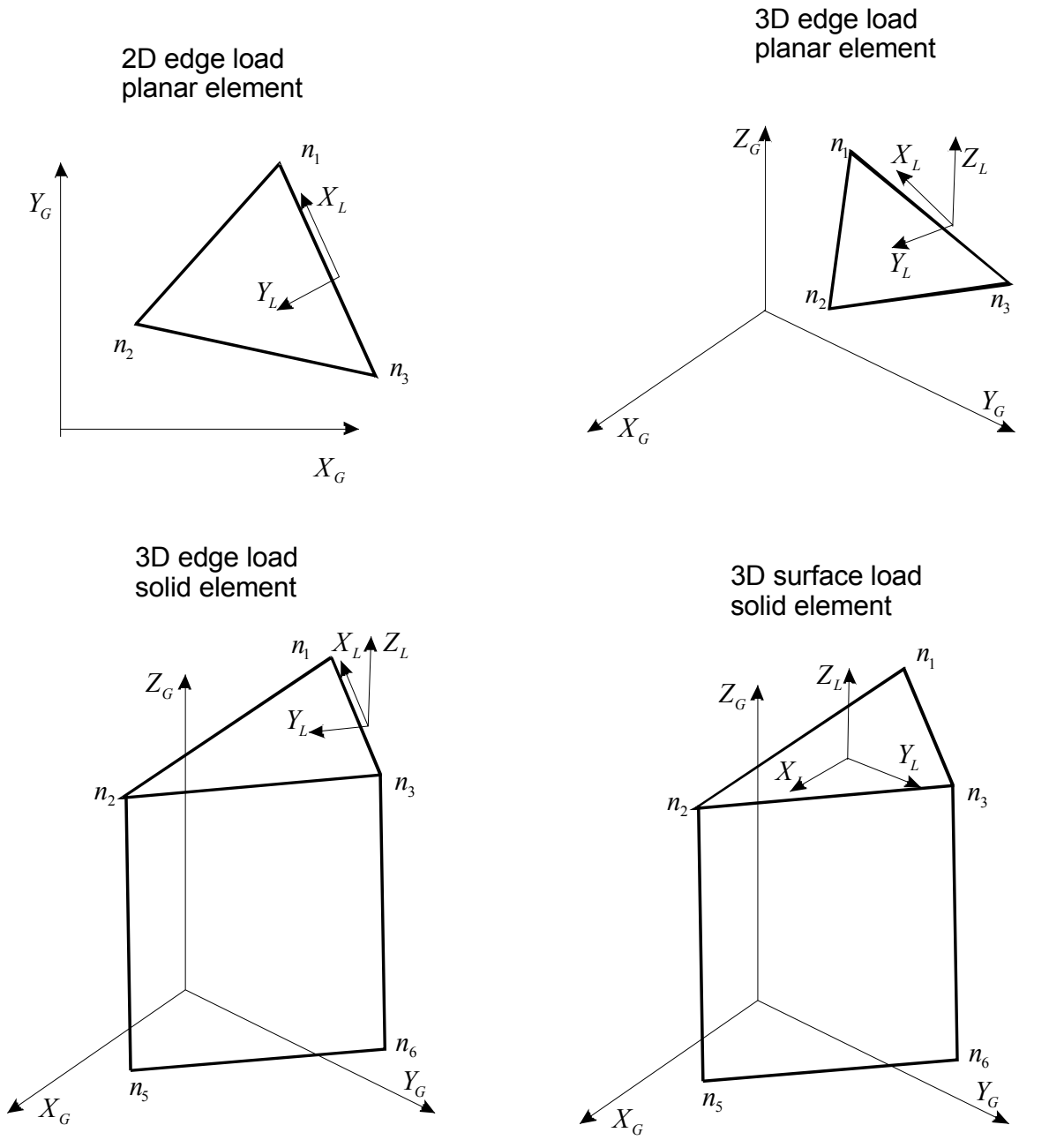


Fig. 3-45 Examples of positioning local coordinate system used by surface and element load for 2D and 3D elements

Transport analysis does not distinguish between local and global element loads. Hence, a local element "load" is treated as being a global load. The actual load value is always scalar, (unlike vectors in statics) and it is assumed positive for flow out of the element.

3.23 Digital printing of concrete structures[4]

Digital 3D printing of concrete and reinforced concrete structures seems to be an innovated, progressive, and economically effective method for building civil engineering structures in future. It has several advantages in comparison to the traditional methods in building industry.

[4] Not available in ATENA version 5.7.0 and older

For example, it allows for miscellaneous shapes of the structures, so that they can be designed more favorably for their static and functional behavior, architectonic design etc. It enables better optimization of the structures resulting in reduced cost, less labor-intensity, less waste produced, greater integration of function and increased speed of the whole construction process. Although most printing methods have not yet showed their full potentials, most engineers agree that they are the right way for civil engineering in near future, because they contribute to better design of the structures and their higher industrialization.

There exists a variety of 3D printing methods used at construction scale, name e.g. 3D extrusion, powder/particle bed printing, 3D block assembling, spraying etc. This Section presents ATENA support for analyses of printed structures using 3D extrusion and describes, how such a construction process can be modelled by this software. It is characterized by printing the structure by layers, i.e. pressing concrete mix thru the nuzzle moving alongside a stepwise linear polygon line that corresponds to individual walls of the structure. Often, some walls are too wide to be printed by one pass of the nuzzle and two or more (parallel) printing passes are needed. Once the current layer has been completed, the printing head returns to its origin, moves one layer upwards and starts printing next layer until full height walls of the structure is produced.

3.23.1 Simplified strength and stability assessments of extruded structures

This section brings preliminary considerations and requirements that should be addressed in design and fabrication of extruded concrete structures. Some derivations below are inspired by papers (Roussel 2018) and (Wolfs at.al. 2018).

3.23.1.1 Material model for stability assessments

Material behavior used for digital fabrication of concrete structures can be modelled by viscoplastic and elastoplastic materials. The former model is suitable for times when the material is being pumped and is flowing to a place of its final position. This time period is not addressed here. We will rather concentrate on the later times, when the material is still fresh, but it is already in rest. At that time, the material features approximately elastoplastic behavior.

There exist several kinds of yield surfaces that define threshold between elastic and fully plastic behavior. Using a few material parameters that are typically obtained from laboratory tests they define general 3D stress-strain conditions when the material start to yield. Uniaxial tensile strength, shear tensile strength etc. are examples of such parameters.

Stress-strain conditions in printed walls are close to 1D conditions, (with self-weight body load only) and thus, throughout all the derivation here we assume 1D elastic behavior up to the material compression f_c. Nevertheless, as some people prefer to measure and use the material shear strength f_{sh}, we will show how to convert 1D material strength f_{1D} to f_{sh} and vice versa. Using e.g. Mises yield surface, https://en.wikipedia.org/wiki/Von_Mises_yield_criterion

$$\sigma_v = \sqrt{3J_2} = \sqrt{\left(\frac{1}{2}\left[\left(\sigma_{11}-\sigma_{22}\right)^2+\left(\sigma_{22}-\sigma_{33}\right)^2+\left(\sigma_{33}-\sigma_{11}\right)^2+6\left(\sigma_{23}^2+\sigma_{31}^2+\sigma_{12}^2\right)\right]\right)} \qquad (3.193)$$

calculate equivalent von Mises stress σ_v (J_2 is the second invariant of stress deviator tensor) for uniaxial test conditions $f_{1D}=\sigma_{11}\neq 0, \sigma_{ij}=0$ for $(i\neq 1)\wedge(j\neq 1)$ and pure shear test conditions

$f_{sh} = \sigma_{12} = \sigma_{21} \neq 0$, otherwise $\sigma_{ij} = 0$. By comparing the corresponding equivalent von Mises stresses, we get the required strength conversion formula:

$$\begin{aligned} \sigma_v &= \sigma_{11} = f_{1D} \\ \sigma_v &= \sqrt{3}\,\sigma_{12} = \sqrt{3}\,f_{sh} \\ f_{1D} &= \sqrt{3}\,f_{sh} \end{aligned} \tag{3.194}$$

Another option is to use maximum shear stress theory, see http://thegateacademy.com/files/wppdf/Theories-of-failure.pdf. It defines yield surface by constraining maximum shear

$$\sigma_{sh,\max} = \max\left[\text{abs}\left(\frac{\sigma_1 - \sigma_2}{2}\right), \text{abs}\left(\frac{\sigma_2 - \sigma_3}{2}\right), \text{abs}\left(\frac{\sigma_3 - \sigma_1}{2}\right)\right] \tag{3.195}$$

where $\sigma_1, \sigma_2, \sigma_3$ are principal stresses. Substituting the above two stress test conditions in (3.195) we get

$$\begin{aligned} &\sigma_1 = \sigma_{11}, \sigma_2 = \sigma_3 = 0 \;\rightarrow\; \sigma_{sh,\max} = \frac{\sigma_{11}}{2} = \frac{f_{1D}}{2} \\ &\sigma_1 = 0, \sigma_2 = \sigma_{12}, \sigma_3 = -\sigma_{12} \;\rightarrow\; \sigma_{sh,\max} = \frac{\sigma_{12} - (-\sigma_{12})}{2} = \sigma_{12} = f_{sh} \\ &f_{1D} = 2\,f_{sh} \end{aligned} \tag{3.196}$$

Total strain theory postulates, see also the above reference:

$$\sigma_{tst} = \sqrt{\sigma_1^2 + \sigma_2^2 + \sigma_3^2 - 2\nu\left(\sigma_1\sigma_2 + \sigma_2\sigma_3 + \sigma_1\sigma_3\right)} \tag{3.197}$$

Then

$$\begin{aligned} &\sigma_1 = \sigma_{11}, \sigma_2 = \sigma_3 = 0 \;\rightarrow\; \sigma_{tst} = \sigma_{11} = f_{1D} \\ &\sigma_1 = 0, \sigma_2 = \sigma_{12}, \sigma_3 = -\sigma_{12} \;\rightarrow\; \sigma_{tst} = \sqrt{2\left(1+\nu\right)}\,\sigma_{12} = \sqrt{2\left(1+\nu\right)}\,f_{sh} \\ &f_{1D} = \sqrt{2\left(1+\nu\right)}\,f_{sh} \end{aligned} \tag{3.198}$$

Of course, a more elaborate and precise yield surface can be employed but we believe that for the preliminary assessment the above simple expressions serve enough accuracy. After all, in ATENA computer analyses one can use any material model suitable for cementitious material. It is more accurate but at the same time also computationally expensive.

3.23.1.2 Strength-based stability of an individual layer

Let as assume a simplified time development of material yield stress $f_c(t)$

$$f_c(t) = \min(f_{c,0} + \dot{f}_c\; t, f_{c,\max}) \tag{3.199}$$

where $f_{c,0}$ is yield stress at time $t = 0$, (i.e. initial value just after material depositing), $f_{c,\max}$ is maximum f_c and $\dot{f}_c$ is structuration rate. The layer is loaded primarily by its gravity self-weight and therefore

$$f_{c,0} \geq h \rho g \tag{3.200}$$

Maximum height of one printed layer is

$$h \leq \frac{f_{c,0}}{\rho g} \tag{3.201}$$

If surface tension γ is considered, it produces stresses of order $\sigma_{st} \approx \frac{\gamma}{h}$. Comparing with (3.201) we get

$$\begin{aligned} h \rho g &\approx \frac{\gamma}{h} \\ h &\approx \sqrt{\frac{\gamma}{\rho g}} \end{aligned} \tag{3.202}$$

For example, for $\gamma = 0.1 \frac{\text{Pa}}{\text{m}}$, (=water) we calculate $h \approx \sqrt{\frac{0.1}{2300\,10}} \doteq 0.002\text{m}$. Therefore, for printed structures stability contribution of $f_{c,0}$ is more important than contribution of surface tension.

3.23.1.3 Collective strength-based stability of more layers

If we consider the case of several printed layers, the lowest layer must resist vertical load $H \rho g$, where H is total height of the structure.

$$\begin{aligned} &\sigma_{11} = H \rho g \leq f_c \\ &f_c = f_{c,0} + \dot{f}_c \, t \geq H \rho g = v_v t \rho g \\ &\dot{f}_c \geq v_v \rho g - \frac{f_{c,0}}{t} \end{aligned} \tag{3.203}$$

where v_v is vertical printing speed. The last expression in (3.203) states minimum structuration rate $\dot{f}$ for being able to print the top layer at time *t*.

The total time t_{tot} for printing height *H* of the structure is

$$t_{tot} = \frac{H}{v_v} = \frac{H}{\frac{h}{t_l}} = \frac{H\,t_l}{h} = \frac{H\frac{l}{v_h}}{h} = \frac{H\,l}{v_h\,h} \tag{3.204}$$

In the above t_l is time to print a single layer, i.e. time necessary for printing head's move along the printing polygon that has total length *l*.

3.23.1.4Collective plastic collapse criterion implemented in ATENA

This section describes steps that are executed to estimate plasticity-based criterion in ATENA. The procedure is inspired by (Suiker 2020) presentation at DC2020 conference in Eindhoven in 2020.

3.23.1.4.1 Linear material curing function

The stability criterion is similar to that presented in the previous section; however, it is expressed in slightly different form. It assumes linear material curing function, i.e.

$$\sigma_p(t) = \sigma_{p,0}(1 + \xi_\sigma t) \tag{3.205}$$

where $\sigma_p(t)$ is material yield strength at time *t*, $\sigma_{p,0}$ is its initial value at $t = 0$ and ξ_σ represents material linear curing rate of the yield stress.

Vertical stress σ_v at the bottom of the wall is, (H is the wall height, ρ is concrete density and *g* states for gravity acceleration)

$$\sigma_v = H\,\rho\,g \tag{3.206}$$

and we require

$$\sigma_p \geq \sigma_v \tag{3.207}$$

For the following derivation, lets introduce dimensionless parameter

$$\overline{\xi}_\sigma = \frac{\left|\rho_{p,0}\right|}{\rho g v_v}\xi_\sigma \tag{3.208}$$

Note that vertical printing speed v_v is in the (Suiker 2020) paper (and Atena) denoted as $\dot{l}$. Substituting $H = v_v\,t$ into (3.206) we get

$$\sigma_p = \sigma_{p,0} * (1 + \xi_\sigma t) \geq v_v \rho g \tag{3.209}$$

After some mathematical manipulation it yields

$$\begin{aligned} \overline{l}_p &\le \frac{1}{1-\overline{\xi}_\sigma} \\ l_p &\le \frac{\left|\rho_{p,0}\right|}{\rho g}\overline{l}_p \end{aligned} \tag{3.210}$$

where l_p is the maximum wall height before the collapse.

If $\sigma_{p,0} = 0$, then the wall is stable for $\frac{\partial \sigma_p}{\partial t} \ge v_v \rho g$, i.e. the case, when time rate of increase of material strength is higher than the rate of increase of vertical stress during printing of the wall. This condition also indicates unlimited wall height.

The paper (Suiker 2020) also discusses, how to calculate σ_p . For the case of pressure-dependent shear failure, they recommend Mohr-Coulomb theory

$$\sigma_p = \frac{2c\cos(\phi)}{1-K-(1+K)\sin(\phi)} \tag{3.211}$$

In the above ϕ is material frictional angle, *c* states for material cohesion and $K = \min(K_y, K_z)$ is minimum of coefficient of lateral stresses $K_y = \sigma_y / \sigma_x,\ K_z = \sigma_z / \sigma_x$, (axis *x* is vertical, axes *y,z* are lateral, i.e. horizontal.

Substituting (3.211) into (3.205) yields

$$\sigma_p = \sigma_{p,0}(1+\xi_p t) = \sigma_{p,0} + \left(\frac{\partial \sigma_p}{\partial \phi}\frac{\partial \phi}{\partial t} + \frac{\partial \sigma_p}{\partial c}\frac{\partial c}{\partial t}\right)t = \sigma_{p,0}\left(1 + \frac{1}{\sigma_{p,0}}\left(\frac{\partial \sigma_p}{\partial \phi}\frac{\partial \phi}{\partial t} + \frac{\partial \sigma_p}{\partial c}\frac{\partial c}{\partial t}\right)t\right) \tag{3.212}$$

From the above

$$\xi_p = \frac{1}{\sigma_{p,0}}\left(\frac{\partial \sigma_p}{\partial \phi}\frac{\partial \phi}{\partial t} + \frac{\partial \sigma_p}{\partial c}\frac{\partial c}{\partial t}\right) \tag{3.213}$$

where

$$\begin{aligned} \frac{\partial \sigma_p}{\partial \phi} &= \frac{-2c\sin(\phi)}{1-K-(1+K)\sin(\phi)} + \frac{2c\cos(\phi)^2(1+K)}{(1-K-(1+K)\sin(\phi))^2} \\ \frac{\partial \sigma_p}{\partial c} &= \frac{2\cos(\phi)}{1-K-(1+K)\sin(\phi)} \end{aligned} \tag{3.214}$$

3.23.1.4.2 Exponential material curing function

This section describes a similar stability assessment; however, exponential decaying curing process is assumed now. This means that Eqn. (3.205) changes to

$$\sigma_p(t) = \sigma_{p,0}(\gamma_p + (1-\gamma_p)\mathrm{e}^{-\xi_\sigma t}) \tag{3.215}$$

where $\gamma_p = \dfrac{\sigma_p(\infty)}{\sigma_{p,0}}$ and ξ_σ is now coefficient of compression strength exponential curing rate.

Substituting (3.215) into (3.206) and (3.207) yields

$$t_{collapse} = \frac{1}{\rho v_v g \xi_\sigma}\left(g v_v \rho \mathrm{W}\left(-\frac{\xi_\sigma \sigma_{p0}\left(\gamma_p - 1\right)}{g v_v \rho}\mathrm{e}^{-\frac{\xi_\sigma \gamma_p \sigma_{p0}}{g v_v \rho}} \right) + \xi_\sigma \gamma_p \sigma_{p0} \right) \tag{3.216}$$

W(z) states for Lambert W(z) function. The maximal wall height at collapse is

$$l_p = t_{collapse} v_v \tag{3.217}$$

3.23.1.5 Buckling stability

Buckling stability of the printed structures may limit the structure even more that strength-base stability. It is computed using Euler Buckling Theory, see http://www.continuummechanics.org/columnbuckling.html. Let us start our derivation with classic beam bending equation that reads

$$E\,I\,\ddot{u} = M \tag{3.218}$$

where E, I state for Young modulus and quadratic moment of inertia, x is longitudinal coordinate of the beam with its origin at the bottom, $u = u(x)$ is deformation and $\ddot{u} = \dfrac{d^2u}{dx^2}$ is its second derivation with respect to x. M is loading moment. Let us assume 1m long section of the wall. It can be modelled by a vertical beam supported at the bottom and loaded by a vertical force P at its top, i.e. $M = -Pu$. Solving differential equation (3.218) yields

$$u = A\sin\left(\sqrt{\frac{P}{E\,I}}x\right) + B\cos\left(\sqrt{\frac{P}{E\,I}}x\right) \tag{3.219}$$

A, B are two constants to be solved from the beam's boundary conditions $u(0) = u(H) = 0$ and $\dot{u}(0) = -\dot{u}(H) \neq 0$. It yields $B = 0$ and when looking for a nontrivial solution, we get $\sqrt{\dfrac{P}{E\,I}}H = \pi$, from which we derive the well-known final expression for critical force

$$P = \frac{\pi^2 E\,I}{H^2} \tag{3.220}$$

The same applies for boundary conditions $u(0) = \dot{u}(0) = \dot{u}(H) = 0$ and $u(H) \neq 0$. For a general case

$$P = \frac{\pi^2 EI}{(kH)^2} \tag{3.221}$$

Substituting $P = H \rho g A$, (A is cross section of the 1m long wall section), we get

$$\begin{aligned} H^2 &= \frac{\pi^2 E I}{Pk^2} = \frac{\pi^2 E \frac{1}{12} 1 w^3}{k^2 H \rho g 1 w} = \frac{\pi^2 E w^2}{k^2 12 H \rho g} \\ H &= \frac{1}{k} \sqrt[3]{\frac{\pi^2 E w^2}{12\ \rho g}} \end{aligned} \tag{3.222}$$

Equation (3.222) states critical height of a printed wall to prevent its collapse due to losing stability. W states for the wall width.

Finally, using (3.222) and (3.203) calculate a threshold H, below which the strength-based stability criterion (3.203) is dominant whilst above it the buckling limit is more restrictive.

$$\begin{aligned} H &= \frac{1}{k} \sqrt[3]{\frac{\pi^2 E w^2}{12\ \rho g}} = \frac{1}{k} \sqrt[3]{\frac{\pi^2 E w^2}{12\ \frac{f_c}{H}}} \\ H &= \frac{\pi w}{k} \sqrt{\frac{E}{12 f_c}} \end{aligned} \tag{3.223}$$

3.23.1.6 Elastic buckling collapse criterion implemented in ATENA

The paper by (Suiker 2020), (Suiker 2018) also presents an estimation of elastic buckling stability of the printed walls. It is more accurate than the criterion from the previous section because it allows for clamp or simple support boundary conditions along the wall vertical edges.

3.23.1.6.1 Linear material curing function

Like 3.23.1.4.1 the material linear curing rate is assumed

$$E(t) = E_0(1 + \xi_E)t \tag{3.224}$$

where $E(t)$ is material Young modulus at time t, E_0 is its initial value at $t = 0$ and ξ_E represents material linear curing rate of elasticity modules.

The employed method is in detail derived in (Suiker 2018). It presents a semi numerical-analytical solution expressed in forms of easily useable plots. The recommended procedure is implemented in ATENA.

The solution uses three dimensionless parameters

$$\begin{aligned} \overline{l}_{cr} &= \sqrt[3]{\frac{\rho g h}{D_0}}\, l_{cr} \\ \overline{b}_{cr} &= \sqrt[3]{\frac{\rho g h}{D_0}}\, b \\ \overline{\xi}_E &= \sqrt[3]{\frac{D_0}{\rho g h}}\, \frac{\xi_E}{v_v} \end{aligned} \tag{3.225}$$

In the above equations l_{cr}, b, h is critical buckling height, horizontal length, (i.e. width), and thickness of the wall, respectively. Vertical printing speed is:

$$v_v = \dot{l} = \frac{q}{v_n h T_l} = \frac{t_l}{T_l} \tag{3.226}$$

with $q = v_n h\, t_l$ being the material volume discharged from the printing nozzle per unit time, T_l is the period required for printing an individual layer and t_l is height, (i.e. thickness) of the printed layer, see the figure below

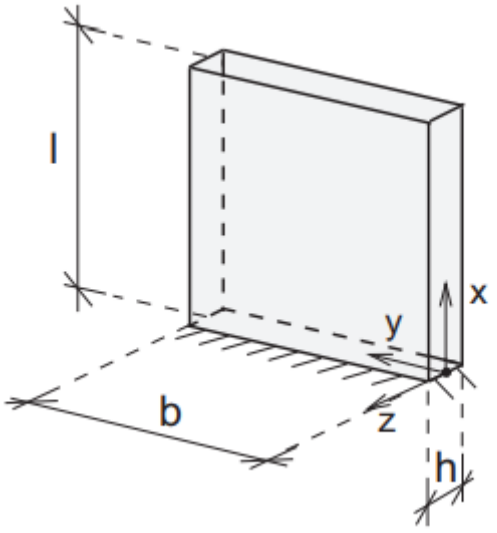


Fig. 3-46 The buckling wall

D_0 states for initial wall bending stiffness defined by

$$D_0 = \frac{E_0 h^3}{12(1-\nu^2)} \tag{3.227}$$

where E_0 is initial Young modulus and ν is Poisson ratio of the material.

The procedure to calculate critical wall height l_{cr} is as follows

1. Calculate D_0 , (3.227).
2. Calculate $\bar{\xi}_E, \bar{b}_{cr}$, (3.225).
3. For the particular support conditions along vertical edges of the wall use Fig. 3-47 and find $\bar{l}_{cr}$ that corresponds to the above $\bar{\xi}_E$, $\bar{b}_{cr}$.
4. Using inverse of the expression for $\bar{l}_{cr}$ calculate l_{cr} , (3.225).

If the printed wall is not supported along its vertical edges, use the dash line for free wall in Fig. 3-47, (i.e. for $\bar{b}_{cr} = \infty$). The dash lines for the case of clamped and simply supported wall yield the same the same $\bar{l}_{cr}$.

Alternatively, (Suiker 2020) recommends $\bar{l}_{cr} = 1.98635 + 0.996\bar{\xi}_E^{\,0.793}$.

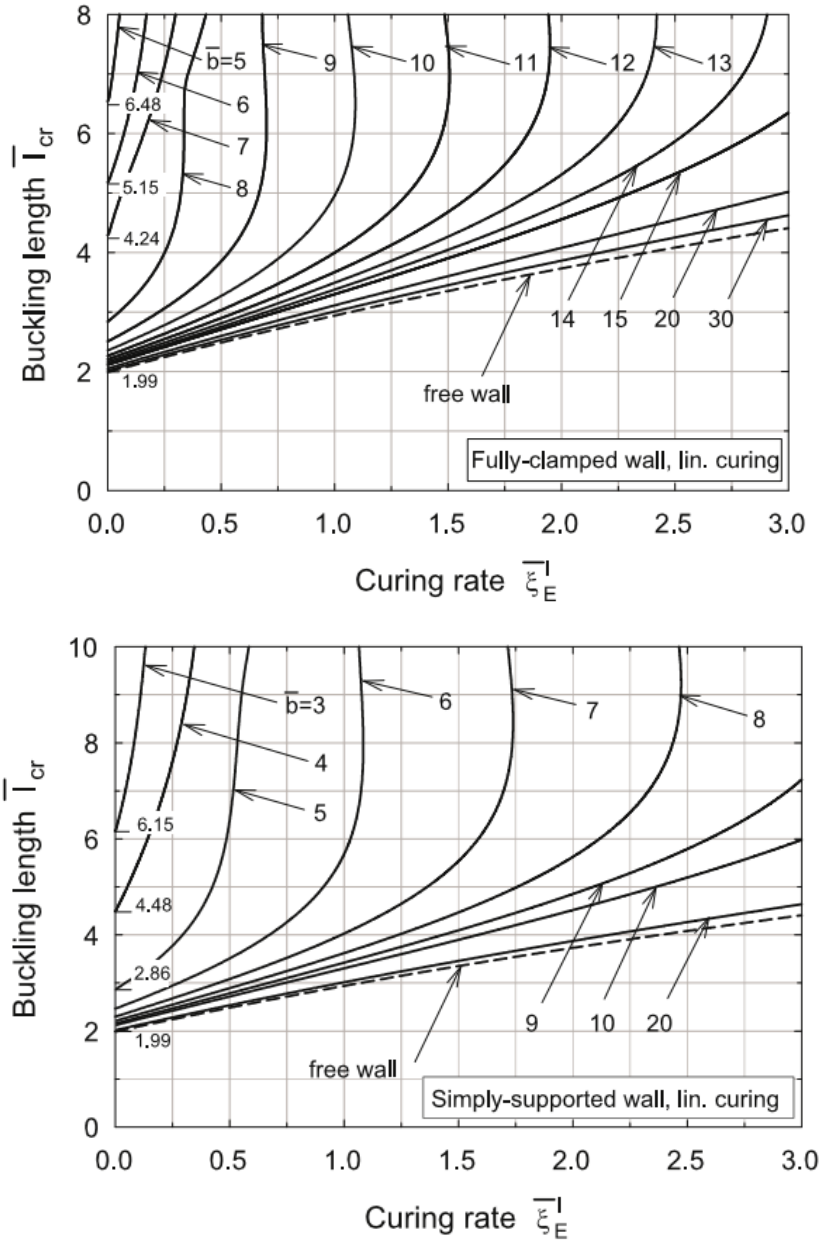


Fig. 3-47 Critical dimensionless buckling length versus dimensionless linear curing rate for the case of fully clamped and simply supported wall.

3.23.1.6.2 Exponential material curing function

This section provides solution for buckling stability subject to exponential material curing rate

$$E(t) = E_0(\gamma_E + (1-\gamma_E)\mathrm{e}^{-\xi_E t}) \tag{3.228}$$

Notation used is similar to the above, i.e. $\gamma_E = \dfrac{E(\infty)}{E_0}$ and ξ_E is now coefficient of exponential Young modulus curing rate.

The overall solution is the same as it was for the case of linear curing, only instead of Fig. 3-47 the solution with exponential curing rate requires to use plots Fig. 48 thru Fig. 50. These plots also comes from (Suiker 2018).

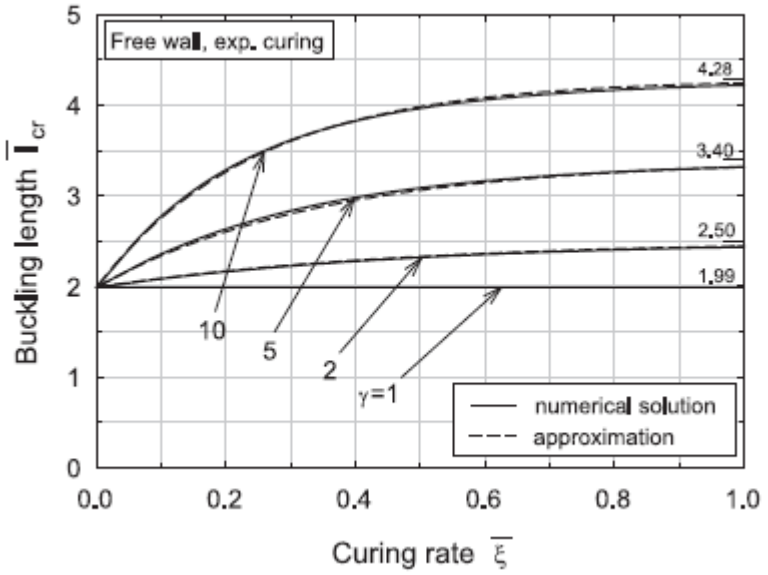
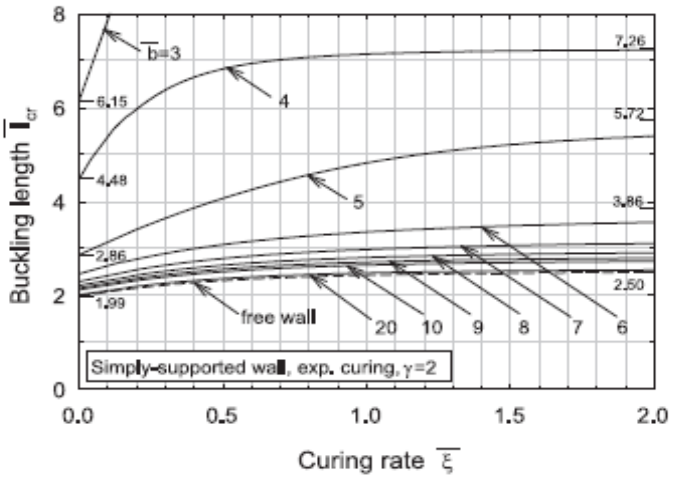
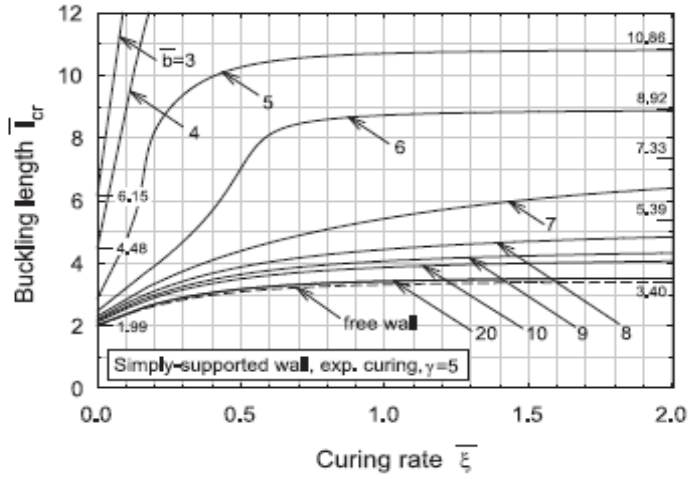


Fig. 48 Critical dimensionless buckling length versus dimensionless exponential curing rate for the case of free wall.

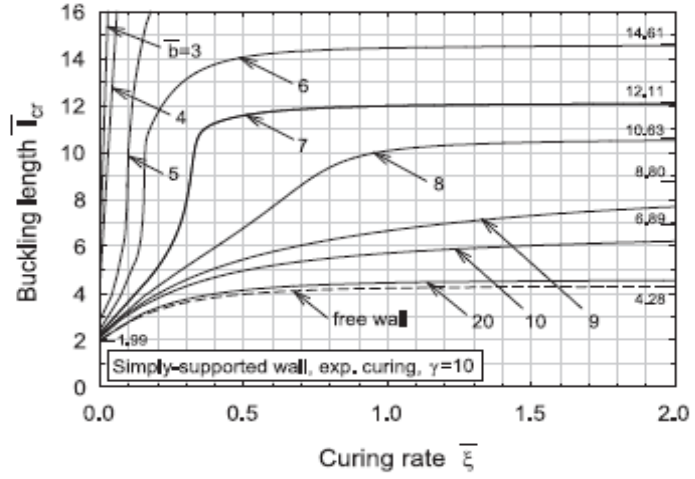
Note that (Suiker 2018) provides the above plots only for $\gamma_E = \{2..10\}$. It is sufficient for modelling some laboratory experiments, but practical analyses typically require values of γ_P much higher.



(a) Curing stiffness ratio $\gamma = \gamma_E = E_\infty / E_0 = 2$.

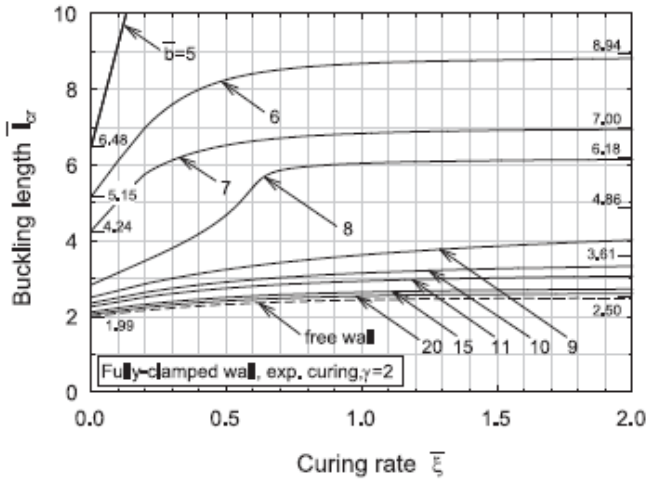


(b) Curing stiffness ratio $\gamma = \gamma_E = E_\infty / E_0 = 5$.

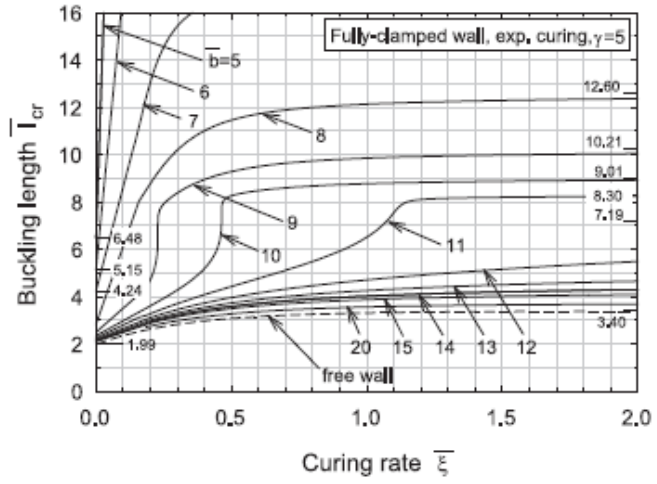


(c) Curing stiffness ratio $\gamma = \gamma_E = E_\infty / E_0 = 10$.

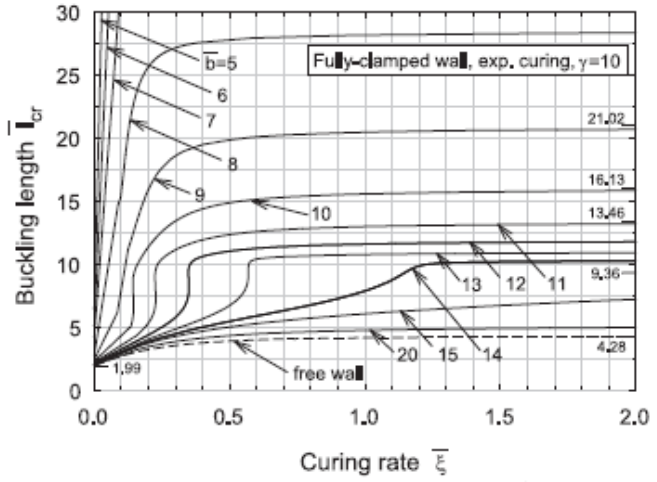
Fig. 49 Critical dimensionless buckling length versus dimensionless exponential curing rate for the case of simply supported wall.



(a) Curing stiffness ratio $\gamma = \gamma_E = E_\infty / E_0 = 2.$



(b) Curing stiffness ratio $\gamma = \gamma_E = E_\infty / E_0 = 5.$



(c) Curing stiffness ratio $\gamma = \gamma_E = E_\infty / E_0 = 10.$

Fig. 50 Critical dimensionless buckling length versus dimensionless exponential curing rate for the case of fully clamped wall.

3.23.1.7 Comparison of buckling stability results provided in Section3.23.1.5 and Section 3.23.1.6 – linear material curing rate

If we assume $f_c = \sigma_p$, then the previously presented strength stability criteria in Section 3.23.1.3 and Section 3.23.1.4 yield the same results. However, the buckling stability criterion in Section 3.23.1.6 is more sophisticated than that from Section 3.23.1.5. It is mainly improved in that it can account for additional boundary conditions along the printed wall's vertical edges. Nevertheless, for the case of unsupported, (i.e. free) vertical edges the two models should yield similar results. This is checked here.

Using Young modulus from (3.224) and vertical printing speed v_v from (3.226) we can write, (see Section 3.23.1.5)

$$H = \frac{1}{k}\sqrt[3]{\frac{\pi^2 E_0 (1+\xi_E) t\, w^2}{12\ \rho g}} = v_v\, t \tag{3.229}$$

where the wall width $w = h$, see Fig. 3-46. Solving the above equation for *t* yields critical wall height l_{cr}

$$\omega = \sqrt[3]{4.5\times10^{14} h^2 E_0 q^3 g^2 k^3 \rho^2 + h^2 E_0 q^2 g^2 k^2 \rho^2 \sqrt{-2.4649\times10^{28}\frac{E_0 T_l^3 h^5 v_n{}^3 \xi_E{}^3}{qgk\rho} + 2.025\times10^{29} q^2 k^2}}$$

$$t = 0.000009700895963\frac{v_n h T_l}{Q^2 g k^2 \rho}\omega + 28232.21472\frac{T_l^2 h^4 v_n{}^2 E_0 \xi_E}{Qk}\frac{1}{\omega}$$

$$l_{cr} = t\, v_v = \frac{t\, Q}{v_n h T_l} \tag{3.230}$$

Example: substituting wall parameters

$$
\begin{aligned}
T_l &= \frac{b}{v_n} \\
E_0 &= 48500 \text{ Pa} \\
\xi_E &= 0.000895 \frac{1}{\text{sec}} \\
q &= 54427 \cdot 0.001^3 \frac{\text{m}^3}{\text{sec}} \\
\rho &= 2100 \frac{\text{kg}}{\text{m}^3} \\
v_n &= \frac{6.25}{60} \frac{\text{m}}{\text{sec}} \\
g &= 10. \frac{\text{m}}{\text{sec}^2} \\
k &= 1 \\
h &= 0.055 \text{ m} \\
b &= 1 \text{ m}
\end{aligned}
\tag{3.231}
$$

the expression (3.230) and (3.225) calculates critical wall height 0.188m an 0.1825m respectively. For the case of $h = 0.1\text{m}$ the expression (3.230) and (3.225) results in 0.305m and 0.292 m.

3.23.2 Steps to carry on analyses of extruded structures

A typical analysis of a structure built by 3D extrusion slightly differs from usual analyses. All the required steps are now described:

Step 1. Prepare a FE model of the structure neglecting the printing process:

The analysis starts by creating a full FE models whereby the process of the printing is ignored. It means that we model the final geometry, properties, and conditions of the structure. Any available FE preprocessor can be used to achieve the goal. Use appropriate (time independent) material model and supply parameters that correspond to the final (long age) material properties.

Step 2. Calculate time of construction t_i^{constr} of each part of the structure, i.e. for each individual element:

Use ATENA UPDATE_ELEMENT_CONSTRUCT_TIME command to accomplish this step. It requires the following data:

- List of element groups that are printed. It is assumed that all elements of the groups are constructed in this way. Actual group's ids are entered via an ATENA selection list.
- Horizontal velocity of the printing head v_h , about 1-10 cm/s.

- Thickness of one printed layer *h*, usually 1-10 cm.
- Width of the printed layer *w*, typically 5-25 cm
- Vector of vertical move from one layer to the next layer $\overline{n}$.
- Track polygon of the printing head's motion. It is specified as an ATENA selection containing ids of FE nodes thru which the printing head passes. The track consists of any number of linear segments. If some segments are not mutually connected, i.e. the track is broken, separate the corresponding segments by inserting *id*=0 between their adjacent end nodes.
- Set start time t_{start} of the track polygon. Typically, $t_{start} = 0$, however if the structure is printed using several track polygons (with e.g. different *width*), then t_{start} of the current polygon equals to time corresponding to the last point of the previous polygon.

Having all the above it follows to calculate time t_i^{constr} of each element. Let $P_{EC} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ are coordinates of center of the element. The element is printed when the head is at the closest position. The track polygon of the moving head is input by setting location of its bottom right edge. Hence, in the following derivations we work with a point *P*, (instead of P_{EC}):

$$P = \begin{bmatrix} x - \left(h n_x + w v_x \right) / 2 \\ y - \left(h n_y + w v_y \right) / 2 \\ z - \left(h n_z + w v_z \right) / 2 \end{bmatrix} \tag{3.232}$$

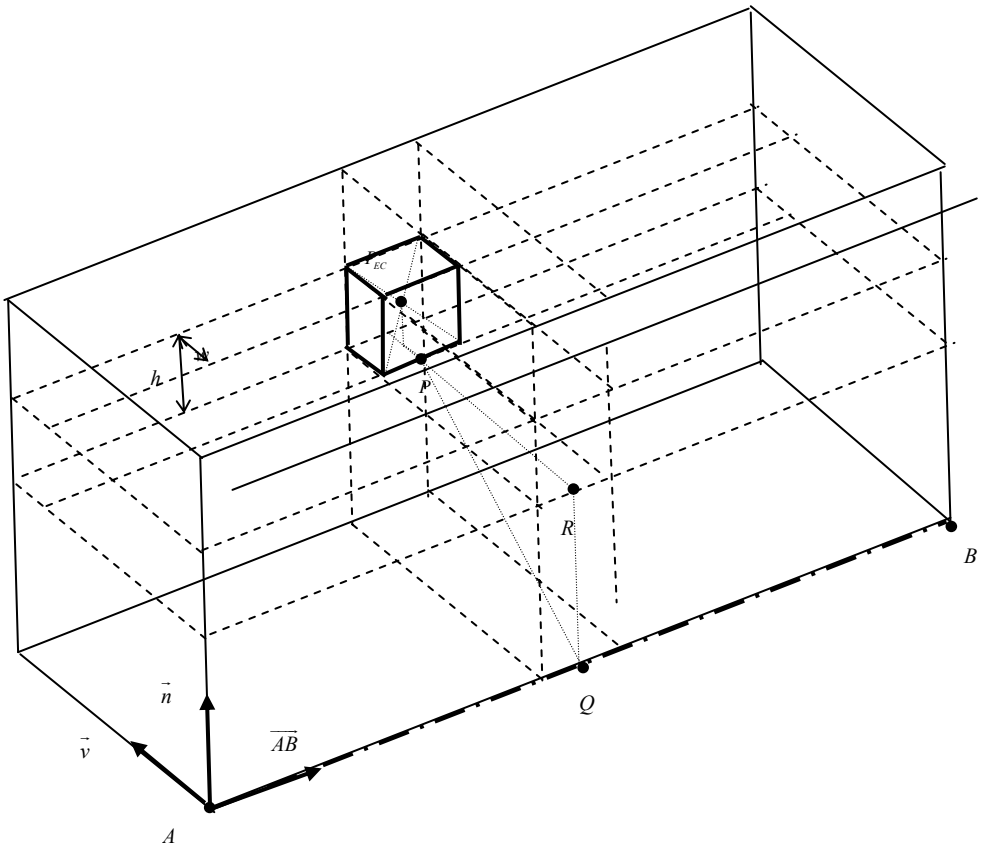


Fig. 3-51 Calculation of age of a particular printed element.

Element construction time is calculated as follows:

$$\left\|\overrightarrow{PQ}\right\| = \frac{\left\|\overrightarrow{AB} \otimes \overrightarrow{AP}\right\|}{\left\|\overrightarrow{AB}\right\|}$$

$$\left\|\overrightarrow{AQ}\right\| = \sqrt{\left(\left\|\overrightarrow{AP}\right\|^2 - \left\|\overrightarrow{PQ}\right\|^2\right)}$$

$$\vec{Q} = \vec{A} + \left\|\overrightarrow{AQ}\right\| \frac{\overrightarrow{AB}}{\left\|\overrightarrow{AB}\right\|} \tag{3.233}$$

$$\left\|\overrightarrow{QR}\right\| = \cos(\overrightarrow{PQ}, \vec{n})\left\|\overrightarrow{PQ}\right\| = \frac{\vec{n} \bullet \overrightarrow{PQ}}{\left\|\vec{n}\right\| \bullet \left\|\overrightarrow{PQ}\right\|}\left\|\overrightarrow{PQ}\right\|$$

$$\vec{R} = \vec{Q} + \left\|\overrightarrow{QR}\right\| \frac{\vec{n}}{\left\|\vec{n}\right\|}$$

The element is printed as a part of a segment $\overrightarrow{AB}$, if point $\vec{Q} \in \overrightarrow{AB}$ and its distance $\left\|\overrightarrow{PR}\right\| <= h/2$. It is printed in a layer id $l = \text{int}\left(\left\|\overrightarrow{QR}\right\| / t\right) + 1$ and has construction time

$$t_i^{constr} = (l-1)\, t_{layer} + t_{prev\_segs} + t_{cur\_seg} \frac{\left\|\overrightarrow{AR}\right\|}{\left\|\overrightarrow{AB}\right\|} \tag{3.234}$$

where t_{layer} is total time to print one layer, i.e. its length divided by v_h , t_{prev_segs} is time to print element in the current layer up to point A and t_{cur_seg} is time to print the current segment $\left\|\overrightarrow{AB}\right\|$. The symbol $\otimes$ and $\bullet$ stand for cross and dot product, respectively. The remaining symbols in the equations are depicted in Fig. 3-104.

3. Account for construction time t_i^{constr} during the analysis:

ATENA calculates structures step by step. Each step has its time t and it stepwise increases. When executing an analysis step, its time is compared with t_i^{constr} of each printed element. If $t \geq t_i^{constr}$, then the element's contribution is assembled as usually, i.e. at its full values. For elements with $t < t_i^{constr}$ ATENA offers two options:

- The element is calculated as usually, i.e. neglecting its t_i^{constr} . It yields unreduced stresses (corresponding to deformation), vector of element forces and matrix of element stiffness. However, before their assembly into global data structures, the vector and matrix is multiplied by a reduction coefficient $\xi \ll 1$. This simulates that the element does not yet exist. The coefficient is defined by ATENA command NEGLIGIBLE_ELEMENT_CONTRIBUTION_COEFF ξ. If $\xi = 0$, the element does not contribute at all.
  Although this approach is simple, it has several disadvantages: it is computationally inefficient because it calculates at each time step all elements despite their contribution to the whole structure is possibly later minimized by the coefficient ξ. The next disadvantage is that it involves some element forces' redistribution, (i.e. some additional iterations), when the element transfers from $t < t_i^{constr}$ to $t \geq t_i^{constr}$ status. Note that it happens in spite of ATENA uses incremental solution technique.
  As discussed previously, the stresses are computed always in full value, i.e. neglecting t_i^{constr} . Now at $t = t_i^{constr}$ we calculate element forces by something like $\overline{F}_i = \int \sigma_i \mathbf{B}_i^T dV = \int \left(\sigma_{i-1} + \mathbf{E}_i \Delta\varepsilon_i\right) \mathbf{B}_i^T dV$. If the structures does not exhibit any deformation increment at the current step, then $\Delta\varepsilon_i = 0 \rightarrow \overline{F}_i = \int \sigma_{i-1} \mathbf{B}_i^T dV$, which is differs from what we used in the previous step, (= $\overline{F}_{i-1} = \xi \int \sigma_{i-1} \mathbf{B}_i^T dV$)!
  On the other hand, this solution approach simulates better the case, when we require print layers having a constant height, (although not quite exactly).
- The second method is to mark all elements active only on condition $t \geq t_i^{constr}$. Use an ATENA command something like
  SELECTION "SOLID_BOX_ELEMENTS"
  CONSTRUCT_TIME_DEPENDENT_ACTIVE GROUP 1

It ensures that elements with $t < t_i^{constr}$ are skipped. They are not computed, not assembled, they don't contribute the structure. They also do not deform, unless dictated by their adjacent elements. This solution is more effective because it calculates only "printed" parts of the structure. Also, no additional iterations are needed. It corresponds to the case when we keep constant top position of each layer, (while its height slightly increases). This method is preferable over the previous one.

4. Account for time dependent material behavior

For this kind of analysis, it is essential to use a material model whose properties vary in time. Mechanical properties of a fresh concrete are certainly significantly different from those for the mature material. For this purpose, ATENA offers CCMaterialWithVariableProperties material model. It builds up on any ATENA material model, but it updates its parameters using explicitly given time functions. Of course, CCMaterialWithVariableProperties accounts for t_i^{constr}, i.e. the time functions receive $(t - t_i^{constr})$ argument. If creep and shrinkage analysis is required, one should use ATENA MATERIAL *id* MAT_CONSTR_TIME Δt command. The material model then calculates behavior of the material being by Δt younger, i.e. current and load time t, t' is replaced by $t - \Delta t, t' - \Delta t$.

5. Loading

A structure produced by digital 3D extrusion requires typically three kinds of boundary conditions:

- Kinematic boundary condition, i.e. definitions of supports etc. They are much the same as for traditionally built structure.
- Self-weight loading: This is modelled by element BODY LOAD option. Use its new "INSIDE_T_TDT_ONLY" flag to add the element's weight only once and at the proper time. For example, use the command something like LOAD BODY group 1 INSIDE_T_TDT_ONLY VALUE Z -0.023 ;
- Material shrinkage: This loading is input as element INITIAL STRAIN load, whereby we must consider element construction time t_i^{constr}. It is achieved by using a new element load's flag CONSIDER_CONSTR_TIME VALUE. At a particular time, younger elements will exhibit a smaller shrinkage than the older ones. For example, use the command something like LOAD TOTAL FUNCTION 100 INITIAL STRAIN group 1 CONSIDER_CONSTR_TIME VALUE X 1. Y 1.000 Z 1.000 ; Note that for the sake of convenience it is recommended to input the load as total load. Therefore, the loading function is defined as TOTAL. (By default, ATENA assumes incremental load, i.e. LOAD INCREMENTAL FUNCTION….).

6. Visualization of printing process

By default, ATENA draws only elements that are active and/or elements active on condition provided $t \geq t_i^{constr}$. However, it can be overridden by checking a special switch, in which case ATENA draws active element only if $t \geq t_i^{constr}$ and/or it draws conditionally active elements

despite their $t < t_i^{constr}$ status. As such, it is always possible to view full or only printed part of the structure.

3.24 References

AHMAD, S., B. M. IRONS, ET AL. (1970). "Analysis of Thick and Thin Shell Structures by Curved Finite Elements." International Journal of Numerical Methods in Engineering 2: 419-451.

BATHE, K.J.(1982), Finite Element Procedures In Engineering Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632, ISBN 0-13-317305-4.

CRISFIELD, M.A. (1983) - An Arc-Length Method Including Line Search and Accelerations, International Journal for Numerical Methods in Engineering, Vol.19,pp.1269-1289.

FELIPPA, C. (1966) - Refined Finite Element Analysis of Linear and Nonlinear Two-Dimensional Structures, Ph.D. Dissertation, University of California, Engineering, pp.41-50.

HINTON, E. AND D. R. J. OWEN (1984). Finite Element Software for Plates and Shells, Peridge Press.

JENDELE, L. (1981). Thick Plate Finite Element based on Mindlin's Theory. Prague, student research work.

JENDELE, L. (1992). Nonlinear Analysis of 2D and Shell Reinforced Concrete Structures Including Creep and Shrinkage. Civil Engineering Department. Glasgow, University of Glasgow: 393.

JENDELE, L., A. H. C. CHAN, ET AL. (1992). "On the Rank Deficiency of Ahmad's Shell Element." Engineering Computations 9(6): 635-648.

RAMM, E. (1981) - Strategies for Tracing Non- linear Responses Near Limit Points, Non- linear Finite Element Analysis in Structural Mechanics, (Eds. W.Wunderlich, E.Stein, K.J.Bathe)

ROUSSEL, N. (2018). - Rheological Requirements for Printable Concretes, Cement and Concrete Research **112**: 76-85.

SUIKER, A. S. (2018). "Mechanical Performance of Wall Structures in 3D Printing

Processes: Theory, Design Tools and Experiments." International Journal of Mechanical Sciences, **137**: 145-170.

SUIKER, A. S., R. J. M. WOLFS, ET AL. (2020). "Elastic Buckling and Plastic Collapse During 3D Concrete Printing." Cement and Concrete Research **135**: 1-16.

WOLFS, R. J. M., F. P. BOS, ET AL. (2018). Early Age Mechanical Behaviour of 3D Printed Concrete: Numerical Modelling and Experimental Testing, Cement and Concrete Research **106**: 103-116.

Suiker, A. S. (2018). "Mechanical Performance of Wall Structures in 3D Printing

Processes: Theory, Design Tools and Experiments." International Journal of Mechanical Sciences, **137**: 145-170.

4 SOLUTION OF NONLINEAR EQUATIONS

The main objective of this chapter is to review methods for the solution of a set of nonlinear equations. Several methods, which are implemented in ATENA are described later in this Chapter. However, all of them need to solve a set of linear algebraic equations in the form

$$\boldsymbol{A}\underline{x} = \underline{b} \tag{4.1}$$

where $\boldsymbol{A}, \underline{x}, \underline{b}$ stands for a global structural matrix and vectors of unknown variables and rhs of the problem, respectively. Hence, this problem is discussed first.

4.1 Linear Solvers

Two types of solvers are supported: direct and iterative, each of them having some pros and cons. Without going into details, a direct solver is recommended for smaller problems or problems. It is more robust and manages better ill-posed equations systems. On the other hand, iterative solvers are typically more efficient to solve large (well-posed) 3D analyses. In addition, two sparse direct solvers are provided. They intend to borrow advantages from both direct and iterative solvers.

The two approaches (i.e., direct and iterative) differ in the way they store the structural matrix $\boldsymbol{A}$. It comes from the nature of FEM that the structural matrices have sparse character, with most of nonzero elements located near the diagonal. The matrix has banded pattern and ATENA works with band of variable width.

If a direct solver is used, then each column of matrix $\boldsymbol{A}$ stores all entries between the diagonal element and the last nonzero element in the column. This structure is sometimes called sky-line profile structure. The matrix $\boldsymbol{A}$

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & & a_{15} & & \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & & \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & & \\ & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} & \\ & & & a_{64} & a_{65} & a_{66} & a_{67} \\ & & & & & a_{76} & a_{77} \end{bmatrix} \tag{4.2}$$

is thus stored in three vectors $\underline{d}, \underline{u}, \underline{l}$ with actual data and one vector $\underline{p}$ with information about matrix's profile:

$$\begin{aligned} \underline{d} &= \begin{bmatrix} a_{11} & a_{22} & a_{33} & a_{44} & a_{55} & a_{66} & a_{77} \end{bmatrix}^T \\ \underline{u} &= \begin{bmatrix} a_{12} & a_{13} & a_{23} & a_{13} & a_{24} & a_{34} & a_{15} & a_{25} & a_{35} & a_{45} & a_{46} & a_{56} & a_{67} \end{bmatrix}^T \\ \underline{l} &= \begin{bmatrix} a_{21} & a_{31} & a_{32} & a_{13} & a_{43} & a_{43} & a_{51} & a_{52} & a_{53} & a_{54} & a_{64} & a_{65} & a_{76} \end{bmatrix}^T \end{aligned} \tag{4.3}$$

$$\underline{p} = \begin{bmatrix} 0 & 1 & 3 & 5 & 9 & 11 & 12 \end{bmatrix}^T$$

For each column i of the matrix $\boldsymbol{A}$ the vector $\underline{p}$ stores location of $a_{i(i-1)}$ within the array $\underline{u}$, resp. $\underline{l}$. If $\boldsymbol{A}$ is symmetric, then $\underline{u} = \underline{l}$ and only $\underline{l}$ is stored. Note the a direct solver we have to

store all elements within the bandwidth, even though some of them may be equal to zero, because that they can become nonzero in the process of solution, (i.e., matrix factorization).

Iterative solver can store only true nonzero elements, irrespective of whether they are located above or below the skyline. Suppose the matrix $\boldsymbol{A}$ from (4.2) that stores some zero elements below the skyline

$$\boldsymbol{A}=\begin{bmatrix} a_{11} & 0 & a_{13} & & a_{15} & & \\ 0 & a_{22} & a_{23} & a_{24} & 0 & & \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 & & \\ & a_{42} & a_{43} & a_{44} & 0 & a_{46} & \\ a_{51} & 0 & 0 & 0 & a_{55} & 0 & \\ & & & a_{64} & a_{65} & a_{66} & a_{67} \\ & & & & & a_{76} & a_{77} \end{bmatrix} \tag{4.4}$$

All iterative solvers would store the matrix $\boldsymbol{A}$ in three vectors. All the data are stored in a vector $\underline{a}$ and location of the stored element is maintained in vectors $\underline{r},\underline{c}$. The above matrix is stored as follows:

$$\begin{aligned} \underline{a} &= \left[a_{11}\ a_{31}\ a_{51}\ a_{22}\ a_{32}\ a_{42}\ a_{33}\ a_{13}\ a_{23}\ a_{43}\ \ldots\ a_{77}\ a_{67} \right] \\ \underline{c} &= \left[\ 1 \quad 3 \quad 5 \quad 2 \quad 3 \quad 4 \quad 3 \quad 1 \quad 2 \quad 3 \ldots \quad 7 \quad 6 \ \right] \\ \underline{r} &= \left[\ 1 \qquad\qquad 4 \qquad\qquad 7 \qquad\qquad\qquad\qquad 21 \qquad 23 \ \right] \end{aligned} \tag{4.5}$$

The vector $\underline{a}$ stores for each column of $\boldsymbol{A}$ first diagonal element, followed by all nonzero elements, from the top to the bottom of the column. The vector $\underline{c}$ stores row index of each entry in the vector $\underline{a}$. $\underline{r}$ stores location of all diagonal elements a_{ii} within $\underline{a}$ appended by an artificial pointer to a_{n+1n+1}, where $n=\dim(\boldsymbol{A})$.

4.1.1 Direct Solver

The well-known Cholesky decomposition is used to solve the problem. The matrix $\boldsymbol{A}$ is decomposed into

$$\boldsymbol{A}=\boldsymbol{LDU} \tag{4.6}$$

where $\boldsymbol{L},\boldsymbol{U}$ is lower and upper matrix and $\boldsymbol{D}$ is diagonal matrix. The method to compute the decomposition is described elsewhere, e.g. (Bathe 1982). Equation (4.1) is then solved in two steps:

$$\begin{aligned} \underline{v} &= \boldsymbol{L}^{-1}\underline{b} \\ \underline{x} &= \left(\boldsymbol{DU}\right)^{-1}\underline{v} \end{aligned} \tag{4.7}$$

Both of the above equations are computed easily, because the involved matrices have triangular pattern. Hence, the solution of (4.7) represents back substitution only. If $\boldsymbol{A}$ is symmetric, (which is usually the case), then

$$\boldsymbol{U} = \boldsymbol{L}^T \tag{4.8}$$

4.1.2 Direct Sparse Solver

Direct sparse solvers are similar to the above Direct solvers; however, they should work more economically both in terms of RAM and CPU requirements. They belong to a group of direct (i.e., non-iterative) solution methods. They are based on matrix decomposition similar to (4.6). The decomposition can be LU or LDU for non-symmetric matrices and/or LL^T or LDL^T decomposition for symmetric matrices.

The main difference between these solvers and those from Section 4.1.3 is that they run the so-called pre-factorization procedure before the actual factorization is executed. Such a pre-factorization has two jobs:

1. Find out, what initially zero a_{ij} entries of the matrix $\boldsymbol{A}$ (that are stored below the skyline) become nonzero due to factorization of $\boldsymbol{A}$. Such entries are called fill-in.
2. Per mutate lines and columns of $\boldsymbol{A}$ so that the filling gets minimum.

Once a map of fill-in is known, it is added to the originally nonzero data of $\boldsymbol{A}$ and only these data are to be stored and maintained in the next operations. Hence, as it is not necessary to store and work upon all data below the skyline of $\boldsymbol{A}$ (as it is he case of solvers in Section 4.1.1); we can use here a sparse matrix storage scheme. The incurred savings in both RAM and CPU resources is significant and it pays off well for a computation overhead caused by the pre-factorization phase and a bit more complicated storage scheme in use.

It is beyond the scope of this document to describe all details about the implementation of this solver. It is based on (Vondracek, 2006) and (Davis et. al, 1995). A number of optimization techniques are used to speed up the solution procedure, such as the problem (4.6) can be solved using a block structure. This applies to pre-factorization, factorization as well as for backward/forward substitution phases. The typical size of such a block is 2x2 .. 6x6. The bigger block size, the smaller overhead for pre-factorization and mapping of the matrix and the faster the operation to actually factorize and solve the problem (4.6). Use of a bigger block, however, results also in a higher waste of RAM because all nonzero data and fill-in are rounded into a storage with block pattern.

Direct sparse solvers are a compromise between Direct Solvers and Sparse Solvers. They typically need more RAM and CPU than Sparse solvers do (and less than Direct Solvers), however, they never diverge and bring uncertainties as what precoditioner to use, etc. Therefore, they are recommended for middle size (may-be ill-conditioned) problems, the solution of which would not fit into RAM subject a Direct Solver is used, and for which Sparse solvers are not sufficiently robust.

4.1.3 Iterative Solver

The table below lists all solvers in ATENA that can solve the problem (4.1) iteratively. Although the list is long, from the practical point of view only a few of them are recommended, see the column "Description". In addition, only the methods DCG and ICCG are designed to take full advantage of symmetry of $\boldsymbol{A}$ (if present)**.** The remaining solvers would store only the symmetric part of $\boldsymbol{A}$**,** however, they will operate on it in the same way as it is not symmetric. Therefore, for symmetric problems, the solvers DCG and ICCG are preferable.

Each of the iterative solvers typically consists of two routines, one for "preparation" of the solution and the other for the solution itself, i.e., "execution" phase. The former routine is

particularly important for the case of preconditioned iterative solvers. This is where a preconditioning matrix is created.

The most efficient preconditioning routine are based on incomplete Cholesky decomposition (Rektorys 1995). The preconditioning matrix $\boldsymbol{A'}$ is decomposed in the same way as (4.6), i.e.

$$\boldsymbol{A' = L'D'U'} \tag{4.9}$$

Comparing $\boldsymbol{A}$ and $\boldsymbol{A'}$, it can be written

$$\begin{aligned} &for\ a_{ij} \neq 0 \qquad a'_{ij} = a_{ij} \\ &for\ a_{ij} = 0 \qquad a'_{ij} \neq a_{ij} \end{aligned} \tag{4.10}$$

The incomplete Cholesky decomposition is carried out in the same way as complete Cholesky decomposition (4.6), however, entries in $\boldsymbol{A}$, which were originally zero and became nonzero during the factorization are ignored, i.e., they stay zero even after the factorization. The incurred inaccuracy is the penalty for memory savings due to usage of the iterative solvers' storage scheme. For symmetric problem, use ssics routine, for non-symmetric problems the ssilus is available to construct $\boldsymbol{A' = L'D'(L')^T}$ or $\boldsymbol{A' = L'D'U'}$.

Last but not least, note that each solver needs some temporary memory. Such requirements are included in the table below. Typically, the more advanced the iterative solver, the more extra memory it needs and the fewer the number of iterations needed to achieve the same accuracy.

Table 4.1-1 SOLVER TYPES.

Type	*D/I*	*Prep. phase*	*Exec. phase*	*Sym/Non-sym*	*Temporary memory required*	*Description*
LU	*D*	---	---	*S,NS*	-----	*For smaller or ill-posed probems*
JAC	*I*	*ssds*	*sir*	*S,NS*	4*(11)+8*(1+4*n)	*Simple, not recommended*
GS	*I*	---	*sir*	*S,NS*	4*(11+nel+n+1)+8*(1+3*n+nel)	
ILUR	*I*	*ssilus*	*sir*	*S,NS*	4*(13+4*n+nu+nl)+8*(1+4*n+nu+nl)	
DCG	*I*	*ssds*	*scg*	*S*	4*(11)+8*(1+5*n)	*For large symmetric well-posed problems*
ICCG	*I*	*ssics*	*scg*	*S*	4*(12+nel+n)+8*(1+5*n+nel)	*For large symmetric problems, recommended*
DCGN	*I*	*ssd2s*	*scgn*	*S,NS*	4*(11)+8*(1+8*n)	*For large non-symmetric well-posed problems*
LUCN	*I*	*ssilus*	*scgn*	*S,NS*	4*(13+4*n+nl+nl)+8*(1+8*n+nl+nu)	*For large non-symmetric problems, recommended*

DBCG	*I*	*ssds*	*sbcg*	*S,NS*	4*(11)+8*(1+8*n)	
LUBC	*I*	*ssilus*	*sbcg*	*S,NS*	4*(13+4*n+nl+nu)+8*(1+8*n+nu+nl)	
DCGS	*I*	*ssds*	*scgs*	*S,NS*	4*(11)+8*(1+8*n)	
LUCS	*I*	*ssilus*	*scgs*	*S,NS*	4*(13+4*n+nl+nu)+8*(1+8*n+nu+nl)	
DOMN	*I*	*ssds*	*somn*	*S,NS*	4*(11)+8*(1+4*n+nsave+3*n*(nsave+1))	
LUOM	*I*	*ssilus*	*somn*	*S,NS*	4*(13+4*n+nu+nl)+8*(1+nl+nu+4*n+nsave+3*n*(nsave+1))	
DGMR	*I*	*ssds*	*sgmres*	*S,NS*	4*(31)+8*(2+n+n*(nsave+6)+nsave*(nsave+3))	
LUGM	*I*	*ssilus*	*sgmres*	*S,NS*	4*(33+4*n+nl+nu)+8*(2+n+nu+nl+n*(nsave+6)+nsave*(nsave+3))	

In the above:

n is the number of degree of freedom of the problem. nel is the number of nonzeros in the lower triangle of the problem matrix (including the diagonal). *nl* and *nu* is the number of nonzeros in the lower resp. upper triangle of the matrix (excluding the diagonal).

Table 4.1-2: EXECUTION PHASES.

Phase name	Description
sir	Preconditioned Iterative Refinement sparse Ax = b solver. Routine to solve a general linear system Ax = b using iterative refinement with a matrix splitting.
scg	Preconditioned Conjugate Gradient iterative Ax=b solver. Routine to solve a symmetric positive definite linear system Ax = b using the Preconditioned Conjugate Gradient method.
scgn	Preconditioned CG Sparse Ax=b Solver for Normal Equations. Routine to solve a general linear system Ax = b using the Preconditioned Conjugate Gradient method applied to the normal equations AA'y = b, x=A'y.
sbcg	Solve a Non-Symmetric system using Preconditioned BiConjugate Gradient.
scgs	Preconditioned BiConjugate Gradient Sparse Ax=b solver. Routine to solve a Non-Symmetric linear system Ax = b using the Preconditioned BiConjugate Gradient method.
somn	Preconditioned Orthomin Sparse Iterative Ax=b Solver. Routine to solve a general linear system Ax = b using the Preconditioned Orthomin method.

sgmres	Preconditioned GMRES iterative sparse Ax=b solver. This routine uses the generalized minimum residual (GMRES) method with preconditioning to solve non-symmetric linear systems of the form: A*x = b.

Table 4.1-3: PREPARATION PHASES.

Phase name	Description
ssds	Diagonal Scaling Preconditioner SLAP Set Up. Routine to compute the inverse of the diagonal of a matrix stored in the SLAP Column format.
ssilus	Incomplete LU Decomposition Preconditioner SLAP Set Up.Routine to generate the incomplete LDU decomposition of a matrix. The unit lower triangular factor L is stored by rows and the unit upper triangular factor U is stored by columns. The inverse of the diagonal matrix D is stored. No fill in is allowed.
ssics	Incompl Cholesky Decomposition Preconditioner SLAP Set Up. Routine to generate the Incomplete Cholesky decomposition, L*D*L-trans, of a symmetric positive definite matrix, A, which is stored in SLAP Column format. The unit lower triangular matrix L is stored by rows, and the inverse of the diagonal matrix D is stored.
ssd2s	Diagonal Scaling Preconditioner SLAP Normal Eqns Set Up. Routine to compute the inverse of the diagonal of the matrix A*A'. Where A is stored in SLAP-Column format.

As for the solution procedure, i.e., the latter of the two solution phases, the most commonly used method is the Conjugate gradient method (with incomplete Cholesky preconditioner) (Rektorys 1995). The flow of execution is as follows:

$$
\begin{aligned}
&\underline{r}^1 = \underline{b} - \mathbf{A}\underline{x}^1 \\
&\underline{z}^1 = \mathbf{M}^{-1}\underline{r}^1 \\
&\\
&\beta^i = \frac{\underline{r}^i \underline{z}^i}{\underline{r}^{i-1} \underline{z}^{i-1}} \\
&\underline{p}^i = \underline{z}^i + \beta^i \underline{p}^{i-1} \\
&\alpha^i = \frac{\underline{r}^i \underline{z}^i}{\underline{p}^i \left(\mathbf{A}\underline{p}^i\right)} \\
&\underline{x}^{i+1} = \underline{x}^i + \alpha^i \underline{p}^i \\
&\underline{r}^{i+1} = \underline{r}^i - \alpha^i \left(\mathbf{A}\underline{p}^i\right) \\
&\underline{z}^{i+1} = \mathbf{M}^{-1}\underline{r}^{i+1} \\
&i = i + 1
\end{aligned}
\tag{4.11}
$$

This solution procedure is implemented in scg routine.

The iterative solvers in ATENA are based on SLAP package (Seager and Greenbaum 1988) that were modified to fit into ATENA framework. The authors of the package refer to (Hageman and Young 1981), where all of the implemented solution techniques are fully described.

4.1.4 Parallel Direct Sparse Solver PARDISO[5]

This solver uses PARDISO parallel direct sparse solver from the Math Kernel Library (MKL) provided by Intel together with Intel Composer XE 2011. The solver has been developed within the PARDISO Project, (see for example http://www.pardiso-project.org/). It is aimed for large sparse symmetric and un-symmetric linear systems with shared memory. It offers direct or iterative solver algorithms. The solver is well established and used by many software packages. A lot of literature is related to the PARDISO project. For more information, refer to http://fgb.informatik.unibas.ch/people/oschenk/index.html. Also, basic information is given in the Intel Composer XE 2011 manuals.

A simplified version of this solver is also included in Atena. For the sake of simplicity, most solution parameters are kept with their default value. The exception to that is the parameter "PARDISO_REQUIRED_ACCURACY". It is input via the Atena "SET" input command. It specifies, whether use of direct method with LU decomposition or iterative method with CGS preconditioning is preferred. In the latter case, it also set a required solution accuracy. (For more information refer to the Atena Input File Manual).

The following solver description is taken from the MKL manual provided by with Intel Composer XE 2011, (also at http://software.intel.com/sites/products/documentation/hpc/mkl/mklman/GUID-7E829836-0FEF-46B2-8943-86A022193462.htm.

Symmetric Matrices:

The solver first computes a symmetric fill-in reducing permutation *P* based on either the minimum degree algorithm (Liu, 1985) or the nested dissection algorithm from the METIS package (Karypis, 1998) (both included with Intel MKL), followed by the parallel left-right looking numerical Cholesky factorization (Schenk, 2000) of *PAPT = LLT* for symmetric

[5] Available starting from ATENA version 5.

positive-definite matrices, or *PAPT = LDLT* for symmetric indefinite matrices. The solver uses diagonal pivoting, or 1x1 and 2x2 Bunch and Kaufman pivoting for symmetric indefinite matrices, and an approximation of *X* is found by forward and backward substitution and iterative refinements.

Whenever numerically acceptable 1x1 and 2x2 pivots cannot be found within the diagonal super-node block, the coefficient matrix is perturbed. One or two passes of iterative refinements may be required to correct the effect of the perturbations. This restricting notion of pivoting with iterative refinements is effective for highly indefinite symmetric systems. Furthermore, for a large set of matrices from different application areas, this method is as accurate as a direct factorization method that uses complete sparse pivoting techniques(Schenk, 2004).

Another method of improving the pivoting accuracy is to use symmetric weighted matching algorithms. These algorithms identify large entries in the coefficient matrix *A* that, if permuted close to the diagonal, permit the factorization process to identify more acceptable pivots and proceed with fewer pivot perturbations. These algorithms are based on maximum weighted matchings and improve the quality of the factor in a complementary way to the alternative idea of using more complete pivoting techniques.

The inertia is also computed for real symmetric indefinite matrices.

Unsymmetric Matrices:

The solver first computes a non-symmetric permutation *PMPS* and scaling matrices *Dr* and *Dc* with the aim of placing large entries on the diagonal to enhance reliability of the numerical factorization process (Duff and Koster 1999). In the next step the solver computes a fill-in reducing permutation *P* based on the matrix *PMPSA* + (*PMPSA*)*T* followed by the parallel numerical factorization

QLUR = PPMPSDrADcP

with super-node pivoting matrices *Q* and *R*. When the factorization algorithm reaches a point where it cannot factor the super-nodes with this pivoting strategy, it uses a pivoting perturbation strategy similar to (Li and Demmel 1999). The magnitude of the potential pivot is tested against a constant threshold of alpha = eps*||*A2*||inf , where eps is the machine precision, *A2 = P*PMPS*Dr*A*Dc*P*, and ||*A2*||inf is the infinity norm of the scaled and permuted matrix *A*. Any tiny pivots encountered during elimination are set to the sign (*lII*)*eps*||*A2*||inf, which trades off some numerical stability for the ability to keep pivots from getting too small. Although many failures could render the factorization well-defined but essentially useless, in practice the diagonal elements are rarely modified for a large class of matrices. The result of this pivoting approach is that the factorization is, in general, not exact and iterative refinement may be needed.

Direct-Iterative Preconditioning.

The solver enables to use a combination of direct and iterative methods (Sonneveld 1989) to accelerate the linear solution process for transient simulation. Most of the applications of sparse solvers require solutions of systems with gradually changing values of the nonzero coefficient matrix, but the same identical sparsity pattern. In these applications, the analysis phase of the solvers has to be performed only once and the numerical factorizations are the important time-consuming steps during the simulation. PARDISO uses a numerical factorization *A = LU* for the first system and applies the factors *L* and *U* for the next steps in a preconditioned Krylow-Subspace iteration. If the iteration does not converge, the solver automatically switches back to the numerical factorization. This method can be applied to un-symmetric and structurally symmetric matrices in PARDISO. For symmetric matrices, Conjugate-Gradients method is applied. You can select the method using only one input parameter.

Separate Forward and Backward Substitution.

The solver execution step can be divided into two or three separate substitutions: forward, backward, and possible diagonal. This separation can be explained by the examples of solving systems with different matrix types.

A real symmetric positive definite matrix A is factored by PARDISO as $A = L*LT$. In this case the solution of the system $A*x=b$ can be found as a sequence of substitutions: $L*y=b$ (forward substitution) and$LT*x=y$ (backward substitution).

A real unsymmetric matrix A is factored by PARDISO as $A = L*U$. In this case the solution of the system $A*x=b$ can be found by the following sequence: $L*y=b$ (forward substitution) and $U*x=y$ (backward substitution).

Note that different pivoting (1x1, 2x2...) produces different *LDLT* factorization. Therefore results of forward, diagonal and backward substitutions with diagonal pivoting can differ from results of the same steps with Bunch and Kaufman pivoting. Of course, the final results of sequential execution of forward, diagonal and backward substitution are equal to the results of the full solving step regardless of the pivoting used.

Sparse Data Storage.

Sparse data storage in PARDISO follows the scheme described above.

4.2 Full Newton-Raphson Method

Using the concept of incremental step by step analysis, we obtain the following set of nonlinear equations:

$$\boldsymbol{K}(\underline{p})\,\Delta\underline{p} = \underline{q} - f(\underline{p}) \tag{4.12}$$

where:

$\underline{q}$ is the vector of total applied joint loads,

$f(\underline{p})$ is the vector of internal joint forces,

$\Delta\underline{p}$ is the deformation increment due to loading increment,

$\underline{p}$ are the deformations of the structure prior to load increment,

$\boldsymbol{K}(\underline{p})$ is the stiffness matrix, relating loading increments to deformation increments.

The R.H.S. of (4.12) represents out-of-balance forces during a load increment, i.e., the total load level after applying the loading increment minus internal forces at the end of the previous load step. Generally, the stiffness matrix is deformation dependent, i.e., a function of $\underline{p}$, but this is usually neglected within a load increment in order to preserve linearity. In this case, the stiffness matrix is calculated based on the value of $\underline{p}$ pertaining to the level prior to the load increment.

The set of equations (4.12) is nonlinear because of the nonlinear properties of the internal forces:

$$f(k\underline{p}) \neq kf(\underline{p}) \tag{4.13}$$

and nonlinearity in the stiffness matrix

$$\boldsymbol{K}(\underline{p}) \neq \boldsymbol{K}(\underline{p} + \Delta\underline{p}) \tag{4.14}$$

where k is an arbitrary constant.

The set of equations represents the mathematical description of structural behavior during one step of the solution. Re-writing equations (4.12) for the *i-th* iteration within a distinct loading increment we obtain:

$$\boldsymbol{K}(\underline{p}_{i-1})\,\Delta\underline{p}_i = \underline{q} - f(\underline{p}_{i-1}) \tag{4.15}$$

All the quantities for the (*i-1)*-th iteration have already been calculated during previous solution steps. Now we solve for $\underline{p}_i$ at load level q using:

$$\underline{p}_i = \underline{p}_{i-1} + \Delta\underline{p}_i \tag{4.16}$$

As pointed out earlier, equation (4.15) is nonlinear, and therefore it is necessary to iterate until some convergence criterion is satisfied. The following possibilities are supported in ATENA (k marks k-th component of the specified vector):

$$\sqrt{\frac{\Delta\underline{p}_i^T\,\Delta\underline{p}_i}{\underline{p}_i^T\,\underline{p}_i}} \le \varepsilon_{rel.disp}$$

$$\sqrt{\frac{(\underline{q} - \underline{f}(\underline{p}_{i-1}))^T\,(\underline{q} - \underline{f}(\underline{p}_{i-1}))}{\underline{f}(\underline{p}_i)^T\,\underline{f}(\underline{p}_i)}} \le \varepsilon_{rel.force} \tag{4.17}$$

$$\sqrt{\frac{\Delta\underline{p}_i^T\,(\underline{q} - \underline{f}(\underline{p}_{i-1}))}{\underline{p}_i^T\,\underline{f}(\underline{p}_i)}} \le \varepsilon_{rel.energy}$$

$$\sqrt{\frac{\max((\underline{q}^k - \underline{f}^k(\underline{p}_{i-1})))\max((\underline{q}^k - \underline{f}^k(\underline{p}_{i-1})))}{\max(\underline{f}^k(\underline{p}_i))\max(\underline{f}^k(\underline{p}_i))}} \le \varepsilon_{abs.force}$$

The first one checks the norm of deformation changes during the last iteration whereas the second one checks the norm of the out-of-balance forces. The third one checks out-of-balance energy, and the fourth condition checks out-of-balanced forces in terms of maximum components (rather than Euclid norms). The values of the convergence limits ε are set by default to 0.01 or can be changed by the input command SET.

The concept of solving nonlinear equation set by Full Newton-Raphson method is depicted in Fig. 4-1:

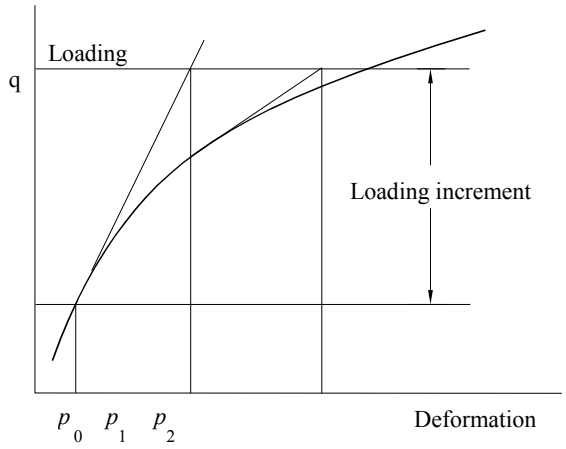


Fig. 4-1 Full Newton-Raphson method.

4.3 Modified Newton-Raphson Method

The most time-consuming part of solution (4.15) is the re-calculation of the stiffness matrix $\boldsymbol{K}(\underline{p}_{i-1})$ at each iteration. In many cases this is not necessary and we can use matrix $\boldsymbol{K}(\underline{p}_0)$ from the first iteration of the step. This is the basic idea of the so-called Modified Newton-Raphson method. It produces very significant time saving, but on the other hand, it also exhibits worse convergence of the solution procedure.

The simplification adopted in the Modified Newton-Raphson method can be mathematically expressed by:

$$\boldsymbol{K}(\underline{p}_{i-1}) \simeq \boldsymbol{K}(\underline{p}_0) \tag{4.18}$$

The modified Newton-Raphson method is shown in Fig. 4-2. Comparing Fig. 4-1 and Fig. 4-2 it is apparent that the Modified Newton-Raphson method converges more slowly than the original Full Newton-Raphson method. On the other hand, a single iteration costs less computing time, because it is necessary to assemble and eliminate the stiffness matrix only once. In practice, a careful balance of the two methods is usually adopted in order to produce the best performance for a particular case. Usually, it is recommended to start a solution with the original Newton-Raphson method and later, i.e., near extreme points, switch to the modified procedure to avoid divergence.

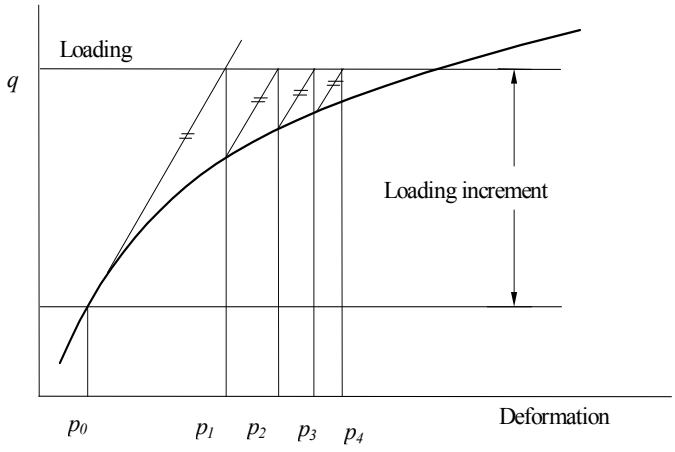


Fig. 4-2 Modified Newton-Raphson method

4.4 Arc-Length Method

Next to the Modified Newton-Raphson method, the most widely used method is the Arc-length method. This method was first employed about fifteen years ago to solve geometrically nonlinear structures. Because of its excellent performance, it is now quite well established for geometric nonlinearity and for material nonlinearity as well. Many workers have been interested in using and improving Arc-length procedures. In Atena, it can be used within CCStructures module, i.e. for static analysis.

The main reason for the popularity of this method is its robustness and computational efficiency which assures good results even in cases where traditional Newton-Raphson methods fail. Using an Arc-length method stability problems such as snap back and snap through phenomena can be studied as well as materially nonlinear problems with non-smooth or discontinuous stress-strain diagrams. This is possible due to the changing load conditions during iterations within an increment.

The main idea of this method is well explained by its name, arc-length. The primary task is to observe complete load-displacement relationship rather than applying a constant loading increment as it is in the Newton-Raphson method. Hence this method fixes not only the loading but also the displacement conditions at the end of a step. There are many ways of fixing these, but one of the most common is to establish the length of the loading vector and displacement changes within the step.

From the mathematical point of view, it means that we must introduce an additional degree of freedom associated with the loading level (i.e., a problem has n displacement degrees of freedom and one for loading) and in addition, a constraint for the new unknown variable must be introduced. The new degree of freedom is usually named λ. There are many possibilities for defining constraints on λ and those implemented in ATENA are briefly reviewed in the following sections.

To derive the Arc-length method, we re-write the set of equations (4.12) in the form of (4.19), where λ defines the new loading factor:

$$\boldsymbol{K}(\underline{p})\,\Delta\underline{p} = \lambda\underline{q} - f(\underline{p}) \tag{4.19}$$

Now re-writing (4.19) in a form suitable for iterative solution:

$$\boldsymbol{K}(\underline{p_{i-1}})\,\Delta\underline{p}_i = \lambda\underline{q} - f(\underline{p}_{i-1}) = \lambda\underline{q} - f_{i-1} \tag{4.20}$$

$$\underline{p}_i = \underline{p}_{i-1} + \Delta\underline{p}_i = \underline{p}_{i-1} + \eta_{i-1}\underline{\delta}_{i-1} \tag{4.21}$$

$$\Delta\underline{p}_i = \Delta\underline{p}_{i-1} + \eta_{i-1}\underline{\delta}_{i-1} \tag{4.22}$$

$$\lambda_i = \lambda_{i-1} + \Delta\lambda_{i-1} \tag{4.23}$$

The notation is explained in Fig. 4-3. The matrix $\boldsymbol{K}$ can be recomputed for every iteration (similar to the Full Newton-Raphson method) or it can be fixed based on the 1st iteration for all subsequent iterations (Modified Newton Raphson method). The vector $\underline{q}$ does not mean in this case the total loading at the end of the step but only a reference loading "type". The actual loading level is a multiple of this.

The scalar η is an additional variable introduced by the Line-search method, which will be discussed later. The scalar η is used to accelerate solutions in cases of well-behaved load-deformation relationships or to damp possible oscillations if some convergence problems arose, e.g., near bifurcation and extreme points.

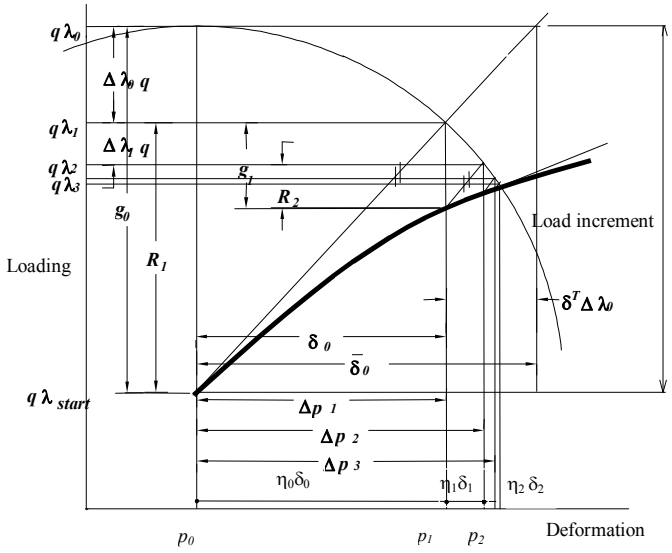


Fig. 4-3 The Arc-length method

Additional notation is defined as follows:

Out-of-balance forces in i-th iteration:

$$g(\underline{p}_i) = \underline{g}_i = \underline{f}_i - \lambda_i \underline{q} = \underline{f} - (\lambda_{i-1} + \Delta\lambda_{i-1})\underline{q}_i \tag{4.24}$$

R.H.S vector in i-th iteration:

$$\underline{RHS}_i = \lambda_i \underline{q} - \underline{f}_{i-1} = \Delta\lambda_{i-1}\underline{q} - \underline{g}_{i-1} \tag{4.25}$$

Substituting (4.21) through (4.25) into (4.20), the deformation increment $\underline{\delta}_{i-1}$ can be calculated from:

$$\boldsymbol{K}\underline{\delta}_{i-1} = \underline{RHS}_{i-1} = \Delta\lambda_{i-1}\underline{q} - \underline{g}_{i-1} \tag{4.26}$$

Hence:

$$\underline{\delta}_{i-1} = \overline{\underline{\delta}}_{i-1} + \Delta\lambda_{i-1}\underline{\delta}_T \tag{4.27}$$

where

$$\begin{aligned} \overline{\underline{\delta}}_{i-1} &= -\boldsymbol{K}^{-1}\underline{g}_{i-1} \\ \underline{\delta}_T &= \boldsymbol{K}^{-1}\underline{q} \end{aligned} \tag{4.28}$$

It remains only to set the additional constraint for $\Delta\lambda_{i-1}$ and η_{i-1} and the whole algorithm is defined. Thus compared to the Newton-Raphson methods in which we solve n dimensional nonlinear problem, the Arc-length method need to solve a (n + 2) dimensional problem, where the first n unknowns correspond to deformations and the last two are $\Delta\lambda_{i-1}$ and η_{i-1}.

If we set $\eta_{i-1} = 1$, then we deal with an (n + *1*) dimensional problem that corresponds to the pure Arc-length method, otherwise, a combination of Arc-length and Line search must be employed. The Line search method is discussed later in this chapter. Note that all vectors including $\underline{\delta}_{i-1}$, $\underline{\delta}_I$ are of order (n + *1*). Their (n + *1*)-th coordinate corresponds to the loading dimension λ and it is set to zero.

Now, introduce two new vectors $\underline{t}_{i-1}$ and $\underline{n}_{i-1}$ as shown in Fig. 4-4. There are defined by:

$$\underline{t}_{i-1} = \Delta\underline{p}_{i-1} + \beta(\underline{\lambda}_{i-1} - \underline{\lambda}_{start}) \tag{4.29}$$

$$\underline{n}_{i-1} = \eta\underline{\delta}_{i-1} + \beta\Delta\underline{\lambda}_{i-1} \tag{4.30}$$

where:

β is scalar that relates dimensions of λ to size of deformation space,

$\underline{\lambda}_{i-1}$ is a (n + *1*) dimensional vector with its firth n coordinates set to zero (deformation space) and its (n + *1*)-th coordinate equal to λ_{i-1}.

$\underline{\lambda}_{start}$ is a (n+*1*) dimensional vector similar to $\underline{\lambda}_{i-1}$, however its ($n$ + *1*)-th coordinate equal to λ_{start}.

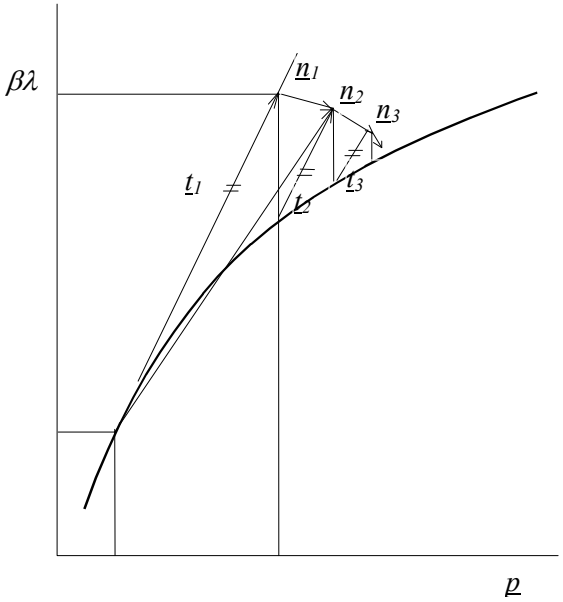


Fig. 4-4 The vectors $\underline{t}_i$ and $\underline{n}_i$ and scalar β.

It is then obvious that

$$\underline{t}_i = \underline{t}_{i-1} + \underline{n}_{n-1} \tag{4.31}$$

Defining the residual R:

$$\underline{R}_{i-1} = \underline{t}_{i-1}\ \underline{n}_{i-1} \tag{4.32}$$

equations (4.20) through (4.32) lead to the final expression for the unknown $\Delta\lambda_{i-1}$ (noting that $\Delta\underline{p}_{i-1}^T\,\Delta\underline{\lambda}_{i-1} = \underline{p}_{i-1}^T\,\underline{\lambda}_{i-1} = 0$):

$$\Delta\lambda_{i-1} = \frac{R_{i-1} - \Delta\underline{p}_{i-1}^T \overline{\underline{\delta}}_{i-1}}{\eta\Delta\underline{p}_{i-1}^T \underline{\delta}_T + \beta^2(\lambda_{i-1} - \lambda_{start})} \tag{4.33}$$

To obtain $\Delta\lambda_{i-1}$ by (4.33) the residual R_{i-1} must be defined. In fact, it also defines the type of Arc-length constrain being used. The types supported in ATENA are described below.

4.4.1 Normal Update Method

Vector t_{i-1} and n_{i-1} are normals in this case, hence residual $R_{i-1} = 0$, see Fig. 4.4-3.

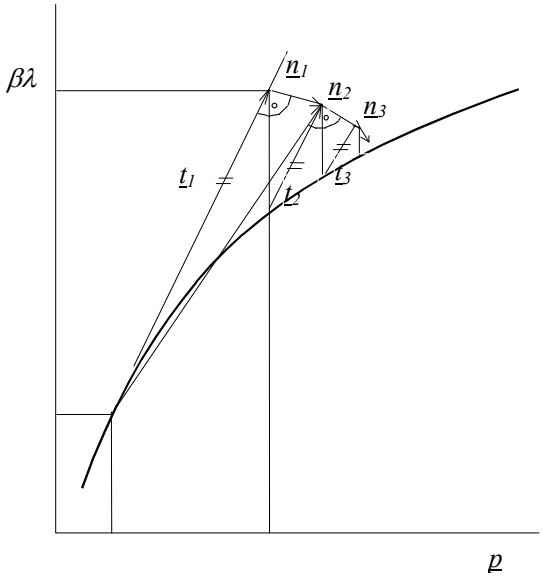


Fig. 4.4-3 Normal update method.

The main advantage of this method is its simplicity. The Normal update plane is relatively reliable, but it can fail if the l-p diagram suddenly changes its slope or turns back or down (snap back and snap through). Nevertheless, if these special conditions are treated by this method, then a very significant reduction in step length is unavoidable.

4.4.2 Consistently Linearized Method

The residual R_{i-1} is defined in this case by

$$R_{i-1} = \underline{t}_{i-1}^T \underline{n}_{i-1} = \left\|\underline{t}_{i-1}\right\| \left\|\underline{n}_{i-1}\right\| \cos(\alpha) = -\left\|\underline{t}_{i-1}\right\| (\left\|\underline{t}_{i-1}\right\| - s) \tag{4.34}$$

The step length s and angle α are depicted in Fig. 4.3-4. The norm of the vector $\left\|\underline{t}_{i-1}\right\|$ is calculated using (4.29):

$$\left\|\underline{t}_{i-1}\right\|^2 = \Delta\underline{p}_{i-1}^T \Delta\underline{p}_{i-1} + \beta^2(\underline{\lambda}_{i-1} - \underline{\lambda}_{start})^2 \tag{4.35}$$

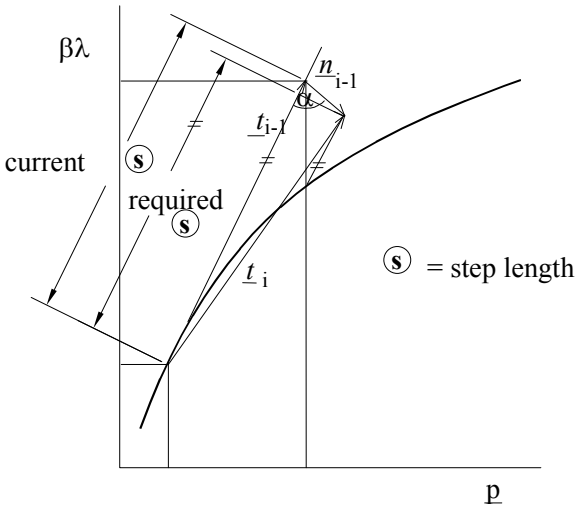


Fig. 4.4-4 Consistently linearized method.

Substituting (4.34) and (4.35) in (4.33) we obtain the final expression for $\Delta\lambda_{i-1}$. It should be noted that the scalar s is set 'a priori' and governs the actual step length. Of course, the proper choice of this parameter is essential for the solution and therefore it will be discussed later in more detail.

This method is especially suitable for solutions that embrace $\lambda - \underline{p}$ diagrams with sudden breaks and discontinuities, e.g. for materially nonlinear problems.

4.4.3 Explicit Orthogonal Method

The basic constraint for $\Delta\lambda_{i-1}$ in this case is that $\|\underline{t}_{i-1}\| = \|\underline{t}_i\| = s$, where s is some distinct 'a priori' set step length. Similar to the previous method, we also have to evaluate the residual R_{i-1}:

$$R_{i-1} = \underline{t}_{i-1}^T \underline{n}_{i-1} = \|\underline{t}_{i-1}\| \, \|\underline{n}_{i-1}\| \cos(\alpha) = -\|\underline{t}_{i-1}\| \, \|\underline{r}_{i-1}\| \tag{4.36}$$

Based on the similar triangles (see Fig. 4.4-), the following can be derived:

$$\frac{\|\underline{r}_{i-1}\|}{\|\underline{t}'_{l-1}\| - s} = \frac{\|\underline{t}_{i-1}\|}{\|\underline{t}'_{l}\|} \tag{4.37}$$

$$R_{i-1} = \frac{-s^2(\|\underline{t}'_i\| - s)}{\|\underline{t}'_i\|} \tag{4.38}$$

$$\underline{t}'_i = \underline{t}_{i-1} + \underline{n}'_{i-1} \tag{4.39}$$

$$\|\underline{t}'_i\|^2 = \|\underline{t}'_{i-1}\|^2 + \beta^2 \Delta\lambda_{i-1}^2 + \eta^2 \|\underline{\delta}_{i-1}\|^2 \tag{4.40}$$

The vector $\|\underline{t}'_{i-1}\|$ is calculated using (4.35). By substituting the above equations into (4.33) the final expression for $\Delta\lambda_{i-1}$ is obtained.

From the above derivation, it is clear that in practice we at first employ Normal Update Method (Chapter 4.4.1) to solve for $\|\underline{t}'_i\|$ and $\|\underline{n}'_{i-1}\|$ and thereafter, we correct the $\Delta\lambda_{i-1}$ in order to satisfy the constraint $\|\underline{t}_{i-1}\| = \|\underline{t}_i\| = s$.

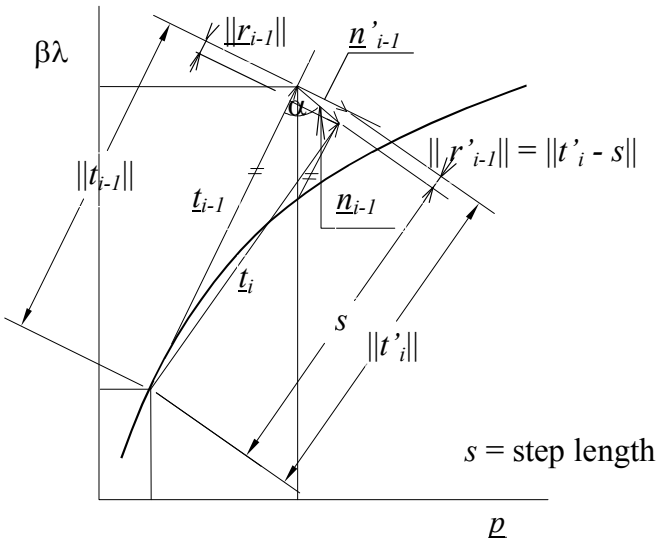


Fig. 4.4-5 Explicit orthogonal method.

This method is usually utilized to analyze geometrically nonlinear structures, particularly stability problems. Its main feature is robustness and compared with the "classical" Crisfield cylinder method (see below) it avoids the problem of the choice of the proper $\Delta\lambda_{i-1}$ root (the condition $\|\underline{t}_{i-1}\| = \|\underline{t}_i\| = s$ while expressing vector length analytically). As for convergence, the method is comparable to the method 4.4.3, but has the advantage that it preserves the step length.

4.4.4 The Crisfield Method.

The Crisfield method is derived directly from the constraint of constant step length $\|\underline{t}_{i-1}\| = \|\underline{t}_i\| = s$ The residual R_{i-1} is not used in this case and we substitute equations (4.20) through (4.31) straight into the above constraint. It leads to the following equation for $\Delta\lambda_{i-1}$:

$$a_1 \Delta\lambda_{i-1}^2 + a_2 \Delta\lambda_{i-1} + a_3 = 0 \tag{4.41}$$

where:

$$a_1 = \eta^2 \underline{\delta}_T^T \underline{\delta}_T + \beta^2$$

$$a_2 = 2\beta^2(\lambda_{i-1} - \lambda_{start}) + 2\underline{\delta}_T^T \overline{\underline{\delta}}_{i-1} \eta^2 \tag{4.42}$$

$$a_3 = \beta^2(\lambda_{i-1} - \lambda_{start})^2 + \eta^2 \overline{\underline{\delta}}_{i-1}^T \overline{\underline{\delta}}_{i-1} - s^2$$

Equation (4.41) has generally two roots $\Delta\lambda_{i-1}$ and hence we must decide which of them to use. There exist several strategies but ATENA chooses that root $\Delta\lambda_{i-1}$, for which $\cos(\underline{t}_{i-1}, \underline{t}_i) \geq 0$ (or higher of them), i.e., direction of new increment as close as possible to direction of the previous increment (within the same step).

4.4.5 Arc Length Step

The proper step length is of essential importance for good execution performance. It directly influences the convergence radius on the one hand and the number of required steps on the other. ATENA uses the following procedure to set (or optimize) s :

(1) Set loading vector $\underline{q}$ and thus define a reference loading level (within one load increment).

(2) Structural response to this load in the 1st execution step, the 1st iteration defines step length s_1 in the 1st step. In the subsequent steps, the step length is kept fixed or optimized (based on SET ATENA input command, subcommand &ARC_LENGTH_OPTIMISATION:

$$s_i = \sqrt{\frac{n_{i-1}}{n}} s_{i-1} \tag{4.43}$$

$$s_i = \sqrt[4]{\frac{n_{i-1}}{n}} s_{i-1} \tag{4.44}$$

$$s_i = \sqrt{\frac{n}{n_{i-1}}} s_{i-1} \tag{4.45}$$

where

s_i and s_{i-1} is Arc length step length in the current and the previous load increment, respectively.

n and n_{i-1} is desired number of iterations and number of iterations in the previous step. n is typically 5-6.

4.5 Line Search Method

The objective of this method is to calculate the parameter η that was already introduced in the Chapter 4.4 Arc-Length Method The method can be used either independently or in combination with Arc length method. The primary reason for introducing a new parameter (i.e. a new degree of freedom to the set of equations) is to accelerate or to damp the speed of analysis of the load-displacement relationship.

The basic idea behind η is to minimize work of current out-of-balance forces on displacement increment.

Let us assume that we have already solved already two points $\underline{p}_0$ and $\underline{p}_0 + \eta' \underline{\delta}$ p and thus we have also calculated out-of-balance forces $\underline{g}(\underline{p}_0)$ and $\underline{g}(\underline{p}_0 + \eta' \underline{\delta})$ at these points. The aim of this method is to set the parameter η so that the work being done by out-of-balance forces at point $\underline{p}_0 + \eta \underline{\delta}$ is minimum.

The work of out-of-balance forces is:

$$\Phi(\underline{p}) = \Phi(\underline{p}_0) + \int_{\underline{p}_o}^{\underline{p}} \underline{g}(\underline{p})^T \, d\underline{p} = minimum \tag{4.46}$$

Hence:

$$\frac{\partial \Phi(\underline{p})}{d\eta} = 0 + \frac{\partial}{\partial \underline{p}}\left(\int_{\underline{p}_0}^{\underline{p}} \underline{g}(\underline{p})^T\right)\frac{\partial \underline{p}}{\partial \eta} = \underline{g}(\underline{p})^T \frac{\partial \underline{p}}{\partial \eta} = 0 \tag{4.47}$$

Interpolating linearly out-of-balance forces between points $\underline{p}_0$ and $\underline{p}_0 + \eta' \underline{\delta}$

$$\underline{g}(\underline{p}_0 + \eta\underline{\delta}) = \underline{g}(\underline{p}_0) + \left(\frac{\underline{g}(\underline{p}_0 + \eta'\underline{\delta}) - \underline{g}(\underline{p}_0)}{\| \underline{p}_0 + \eta'\underline{\delta} - \underline{p}_0 \|}\right) \| \underline{p}_0 + \eta\underline{\delta} - \underline{p}_0 \| = \underline{g}(\underline{p}_0) + \frac{\underline{g}(\underline{p}_0 + \eta'\underline{\delta}) - \underline{g}(\underline{p}_0)}{\eta'}\eta \tag{4.48}$$

and using :

$$\begin{aligned} \underline{p} &= \underline{p}_0 + \eta\underline{\delta} \\ \frac{\partial \underline{p}}{\partial \eta} &= \underline{\delta} \end{aligned} \tag{4.49}$$

The final expression for η' can be derived:

$$\eta = \eta' \frac{\underline{g}(\underline{p}_0)^T \underline{\delta}}{\underline{g}(\underline{p}_0)^T \underline{\delta} - \underline{g}(\underline{p}_0 + \eta'\underline{\delta})^T \underline{\delta}} \tag{4.50}$$

Thus, the Line search method can be summarized:

Use any method to calculate displacement increment $\underline{\delta}$, (see Fig. 4-3 and (4.28)). The parameter η' can be set from the last load increment or simply to unity.

Calculate out-of-balance forces for both $\underline{g}(\underline{p}_0)$ and $\underline{g}(\underline{p}_0 + \eta'\underline{\delta})$.

Use (4.50) to calculate new value for η.

As all the above equations are nonlinear, the parameter η must be solved by iterations until $\frac{\left\|\underline{g}(\underline{p}_0 + \eta\underline{\delta}\right\|}{\underline{g}(\underline{p}_0)} \leq$ a specified energy drop, typically < 0.6 – 0.8 >.

Practical experience suggests that the value of parameter η should be kept in interval < 0.1 – 5>.

4.6 Parameter β

The parameter β scales the deformation space $\underline{p}$ to the loading dimension λ. If $\beta = 0$, the solution for $\Delta\lambda_{i-1}$ is searched on an area of a cylindrical shape of radius equal to step length s (Crisfield method) and the axis normal to the $\underline{p}$ (deformation) space. The solution is the point of intersection of this area and the line, defined by the energy gradients of structure and by the applied load at point $\underline{p}$. If $\beta > 0$, the solution is carried out in the same way on ellipsoidal or spherical space.

The higher value of β, the higher "weight factor" for changes in loading space compared to displacement increments.

ATENA currently supports the following formulae for setting and optimization of β (for current step j). They are reviewed below.

The first strategy requires the "load to displacement" increment ratio (4.51) is constant throughout all steps, (e.g., input value B_{req})

$$\mathrm{B} = \frac{\beta \ \Delta\lambda}{\left\|\Delta(\underline{p})\right\|} = \mathrm{B}_{req} \tag{4.51}$$

Then, at the end step *j-1* we can calculate

$${}_{j-1}\mathrm{B} = \frac{{}_{j-1}\beta \ \Delta_{j-1}\lambda}{\left\|\Delta({}_{j-1}\underline{p})\right\|} \tag{4.52}$$

This value (due to nonlinearities) will not match B_{req}. Therefore, for step *j* we will modify ${}_{j}\beta$ as follows:

$$\begin{aligned} {}_{j}\mathrm{B} &= \mathrm{B}_{req} = {}_{j-1}\mathrm{B}\,\chi \\ \chi &= \frac{\mathrm{B}_{req}}{{}_{j-1}\mathrm{B}} \\ {}_{j}\beta &= {}_{j-1}\beta\,\chi = {}_{j-1}\beta\frac{\mathrm{B}_{req}}{{}_{j-1}\mathrm{B}} = {}_{j-1}\beta\frac{\mathrm{B}_{req}}{\frac{{}_{j-1}\beta \ \Delta_{j-1}\lambda}{\left\|\Delta({}_{j-1}\underline{p})\right\|}} = \mathrm{B}_{req}\frac{\left\|\Delta({}_{j-1}\underline{p})\right\|}{\Delta_{j-1}\lambda} \end{aligned} \tag{4.53}$$

The above optimization process is initialized in the first step by assuming that ${}_{0}\beta = 1,\ {}_{0}\Delta\lambda = 1, \left\|\Delta({}_{j-1}\underline{p})\right\| = \left\|\underline{\delta}_{T}\right\|$, where $\underline{\delta}_{T}$ is displacement corresponding to master Arc-length load increment defined earlier in this chapter. Hence

$${}_{1}\beta = {}_{0}\beta\,\chi = \frac{\mathrm{B}_{req}}{{}_{j-1}\mathrm{B}} = \frac{\mathrm{B}_{req}}{\frac{{}_{j-1}\beta \ \Delta_{j-1}\lambda}{\left\|\Delta({}_{j-1}\underline{p})\right\|}} = \frac{\mathrm{B}_{req}}{\frac{1}{\left\|\underline{\delta}_{T}\right\|}} = \mathrm{B}_{req}\left\|\underline{\delta}_{T}\right\| \tag{4.54}$$

The parameters ${}_{j}\mathrm{B}$ in all subsequent steps are calculated using (4.53). If the ratio of displacements changes $\left\|\Delta({}_{j}\underline{p})\right\|$ to load changes $\Delta({}_{j}\lambda)$ in the last load step increase, then the equation (4.54)(4.55) increases β in the current step, thereby puts higher „weight factor" on loads compared to displacements. Hence, the equation (4.54) tends to keep constant importance of loading space irrespective of displacements. Note that the equation (4.54) corresponds to BETA_FORCES_DISPLS_RATIO_CONSTANT.

The second supported strategy is different. In ATENA, it is referred to as BETA_RATIO_CONSTANT method and it tries to keep constant β coefficients, whilst managing the coefficients B. Thus, it works in the opposite way as compared to the first strategy described above.

From (4.52) we can write for steps *(j-1)* and *j*

$$ {}_{j-1}\beta = \frac{{}_{j-1}\mathrm{B}\left\|\Delta({}_{j-1}\underline{p})\right\|}{\Delta_{j-1}\lambda} $$

$$ {}_{j}\beta = \frac{{}_{j}\mathrm{B}\left\|\Delta({}_{j}\underline{p})\right\|}{\Delta_{j}\lambda} $$

Now requiring ${}_{j-1}\beta = {}_{j}\beta$ we have

$$ \frac{{}_{j}\mathrm{B}\left\|\Delta({}_{j}\underline{p})\right\|}{\Delta_{j}\lambda} = \frac{{}_{j-1}\mathrm{B}\left\|\Delta({}_{j-1}\underline{p})\right\|}{\Delta_{j-1}\lambda} $$

$$ \frac{{}_{j}\mathrm{B}}{{}_{j-1}\mathrm{B}} = \frac{\dfrac{\left\|\Delta({}_{j-1}\underline{p})\right\|}{\Delta_{j-1}\lambda}}{\dfrac{\left\|\Delta({}_{j}\underline{p})\right\|}{\Delta_{j}\lambda}} \tag{4.56} $$

and if we assume $\dfrac{\Delta_{j-1}\lambda}{\left\|\Delta({}_{j-1}\underline{p})\right\|} = \dfrac{\Delta_{j}\lambda}{\left\|\Delta({}_{j}\underline{p})\right\|}$, then $\dfrac{{}_{j}\mathrm{B}}{{}_{j-1}\mathrm{B}} = \dfrac{{}_{j}\beta}{{}_{j-1}\beta}$ and the above equation yields

$$ {}_{j}\beta = {}_{j-1}\beta \frac{\dfrac{\left\|\Delta({}_{j-1}\underline{p})\right\|}{\Delta_{j-1}\lambda}}{\dfrac{\left\|\Delta({}_{j}\underline{p})\right\|}{\Delta_{j}\lambda}} \tag{4.57} $$

If $\dfrac{\dfrac{\left\|\Delta({}_{j-1}\underline{p})\right\|}{\Delta_{j-1}\lambda}}{\dfrac{\left\|\Delta({}_{j}\underline{p})\right\|}{\Delta_{j}\lambda}}$ in subsequent steps changes, the procedure is trying to compensate for that by re-adjusting the coefficients β. In other words, this strategy is trying to keep $\dfrac{\left\|\Delta(\underline{p})\right\|}{\Delta\lambda}$ constant, (i.e., the relative importance of load vs. displacement spaces).

4.7 Band Width Optimization

The way in which individual structural degrees of freedom (dofs) are mapped into the global structural matrices has a significant impact on their size and cost of the solution in terms of required CPU and RAM resources.

Let us assume the 2D example of the 3 bars element from Fig. 4-5. The structure consists of three beam elements 1,2,3. It has four global nodes with three degrees of freedom in each of them, i.e., two displacements and one rotation. Suppose the structure is solved by a direct solver, i.e., we use half-band skyline storage scheme (4.4).

By default, i.e., without any optimization, the structural degrees of freedom are allocated sequentially starting from the node 1 up to the last node *n,* i.e., 4. Hence, the *j*th degree of freedom at the node i has number $ndof\,(i-1)+j$, where *ndof* is number of dofs per node.

If the structural nodes are numbered as indicated, then the beam 1,2 and 3 have nodal incidences 1-3, 3-4 and 4-2, respectively and the final stiffness matrix **K** has the pattern from the left-bottom part of Fig. 4-5. Note that the matrix **K** must also store the entries depicted as circles without filling. Although they are initially zero, they may turn nonzero during the matrix decomposition needed to solve the problem, i.e., we must store the matrix with 69 entries and maximum half-band width 9.

On the other hand, if nodal degrees of freedom are numbered as shown in the right-bottom part of Fig. 4-5, then the matrix **K** must store only 51 entries and has maximum half bandwidth only 6.

The two examples document, how important efficient numbering of the degrees of freedom of the structure is. If the structure (to be solved) is simple, then a suitable dofs' numbering can be done manually by appropriate numbering of the structural nodes. However, in the more complex cases (and in particular if a model of the structure is generated automatically), an optimal dofs mapping must be calculated.

There are number of algorithms that deliver more or less efficient dofs mapping. Probably the best established algorithm of that kind is Cuthill-McKee algorithm (Cuthill, McKee 1969). This is not due to its superior property, but due it has been developed as first. The algorithm produces an ordered n-tuple R of vertices which is the new order of the structural vertices. It numbers the vertices according to a particular breadth-first traversal, where neighboring vertices are visited in order from lowest to highest vertex order.

The reverse Cuthill–McKee algorithm (RCM) is the alternative of the Cuthill-McKee algorithm, in which the vertices are visited in reverse order, i.e. form the highest to the lowest vertex.

ATENA implements Gibbs and Sloan dofs optimization algorithms:

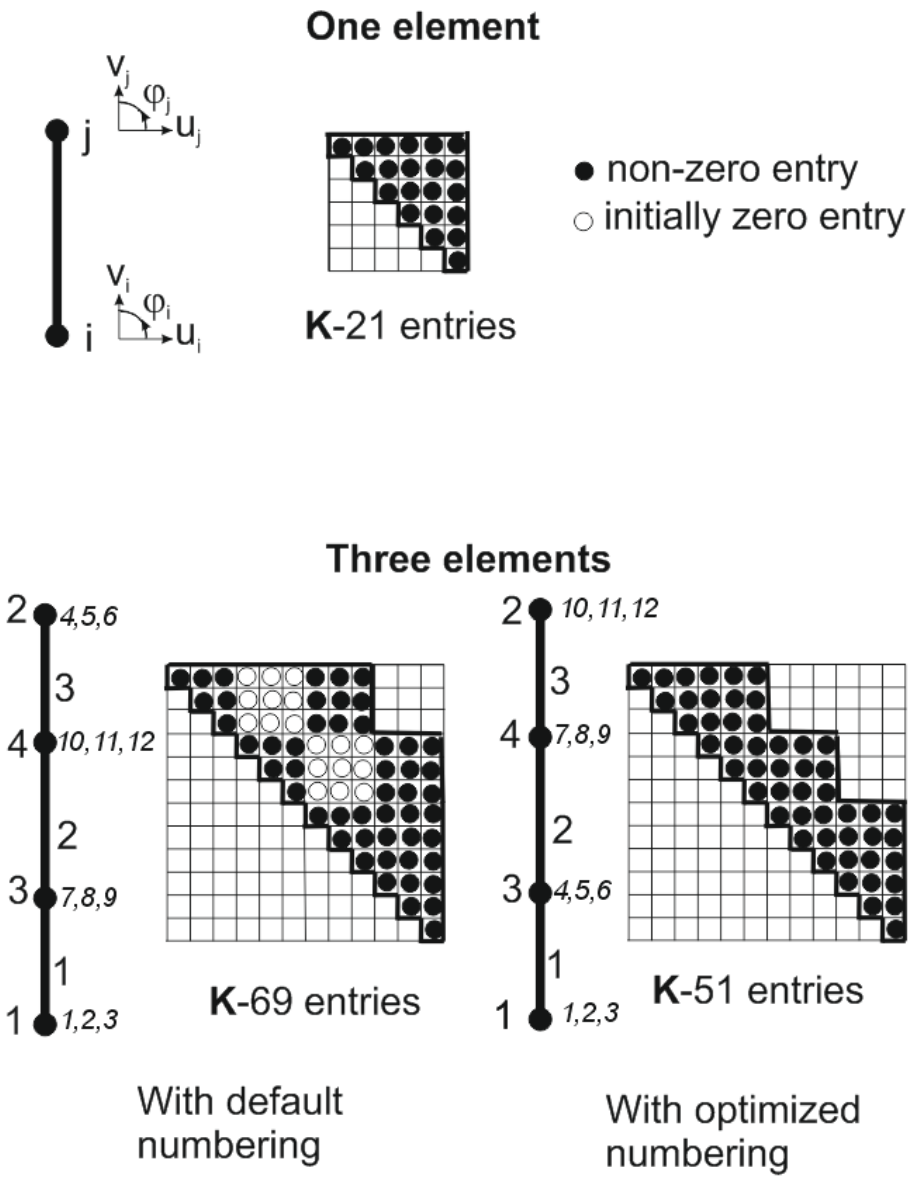


Fig. 4-5 Optimization of dofs numbering

The Sloan algorithm (Sloan, Randolf (1983)

In an effort to obtain an optimum elimination order, the algorithm first renumbers the nodes, and then uses this result to resequence the elements. This intermediate step is necessary because of the nature of the frontal solution procedure, which assembles variables on an element-by-element basis but eliminates them node by node. To renumber the nodes, a modified version of the King' algorithm is used. In order to minimize the number of nodal numbering schemes that need to be considered, the starting nodes are selected automatically by using some concepts from graph theory. Once the optimum numbering sequence has been ascertained, the elements are then reordered in an ascending sequence of their lowest-numbered nodes. This ensures that the new elimination order is preserved as closely as possible. For meshes that are composed of a single type of high-order element, it is only necessary to consider the vertex nodes in the renumbering process. This follows from the fact that mesh numberings which are optimal for low-order elements are also optimal for high-order elements. Significant economies in the reordering strategy may thus be achieved.

The Gibbs et. al. algorithm (Gibbs et. al. 1976)

This algorithm typically produces bandwidth and profile, which are comparable to those of the commonly-used reverse Cuthill–McKee algorithm, yet it requires significantly less computation time. Nevertheless, it delivers dofs mapping that is usually slightly less efficient than that by the Sloan algorithm and therefore, it is less preferred option the optimization.

Note that the above algorithms optimize dofs numbering by reordering the structural nodes. They do not account for possible different number of dofs within a particular node. Note also that in order to minimize cost of the dofs remapping, the optimization is carried out before assembling the structural global matrices and vectors. Thus, they are assembled directly into their final, optimized location.

Iterative solvers use data storage scheme (4.3). As the storage scheme stores only nonzero elements, the solution is less sensitive to a bad dofs mapping. For huge analyses it is nevertheless suggested to carry out a dofs mapping optimization, as it typically yields individual elements entries stored closer to each other with positive effect on solution convergence and RAM data management.

A detailed description of the above algorithms is above scope of the publication. For more information the reader is suggested to study the given references.

4.8 References

BATHE, K.J.(1982), Finite Element Procedures In Engineering Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632, ISBN 0-13-317305-4.

CRISFIELD, M.A. (1983) - An Arc-Length Method Including Line Search and Accelerations, International Journal for Numerical Methods in Engineering, Vol.19, pp.1269-1289.

CUTHILL, E. and J. MCKEE (1969). Reducing the Bandwidth of Sparse Symmetric Matrices. Proc. 24th Nat. Conf. ACM.

DAVIS, T., AMESTOY, P., DUFF, I.S (1995) - An Aproximate Minimum Degree Ordering Algorithm, Comp. and Information Science Dept., University of Florida, Tech. Report TR-94-039.

DUFF, I. S. and KOSTER, J. (1999) - The Design and Use of Algorithms for Permuting Large Entries to the Diagonal of Sparse Matrices. SIAM J. Matrix Analysis and Applications, 20(4):889-901.

FELIPPA, C. (1966) - Refined Finite Element Analysis of Linear and Nonlinear Two-Dimensional Structures, Ph.D. Dissertation, University of California, Engineering, pp.41-50.

GIBBS, N. E., W. G. POOLE, et al. (1976). "An Algorithm for Reducing the Bandwidth and Prole of a Sparse Matrix." SIAM Journal of Numerical Analysis 13(2).

KARYPIS, G. and KUMAR, V. (1998) - A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. SIAM Journal on Scientific Computing, 20(1):359-392.

LIU, J.W.H. (1985) - Modification of the Minimum-Degree Algorithm by Multiple Elimination. ACM Transactions on Mathematical Software, 11(2):141-153.

LI, X.S., DEMMEL, J, W. (1999) - A Scalable Sparse Direct Solver Using Static Pivoting. In Proceeding of the 9th SIAM conference on Parallel Processing for Scientific Computing, San Antonio, Texas, March 22-34.

RAMM, E. (1981) - Strategies for Tracing Non- linear Responses Near Limit Points, Non- linear Finite Element Analysis in Structural Mechanics, (Eds. W.Wunderlich,E.Stein, K.J.Bathe)

REKTORYS, K. (1995). Přehled užité matematiky. Prague, Prometheus.

SLOAN, S. W. and M. F. RANDOLF (1983). "Automatic Element Reordening for Finite Element Analysis with Frontal Solution Schemes." Int. Journal for Numerical Methods in Eng. 19: 1153-1181.

SEAGER, M. K. and A. GREENBAUM (1988). A SLAP for the Masses, Lawrence Livermore National Laboratory

SCHENK, O., GARTNER, K. and FICHTNER, W. (2000) - Efficient Sparse LU Factorization with Left-right Looking Strategy on Shared Memory Multiprocessors. BIT, 40(1):158-176.

SCHENK, O. and GARTNER, K. (2004) - On Fast Factorization Pivoting Methods for Sparse Symmetric Indefinite Systems. Technical Report, Department of Computer Science, University of Basel, submitted.

SONNEVELD, P. (1989) - CGS, a Fast Lanczos-Type Solver for Nonsymmetric Linear Systems. SIAM Journal on Scientific and Statistical Computing, 10:36-52.

VONDRACEK, R. (2006) - The Use Of The Sparse Direct Solver In The Egineering Applications Of The Finite Element Method. Theses for Ph.D. Czech Technical University, Prague.

5 CREEP AND SHRINKAGE ANALYSIS

Creep and shrinkage are undoubtedly features that have a significant influence on concrete behaviour. Although creep and shrinkage analysis can be neglected in the design of most civil structures, there exist cases when these phenomena have to be accounted for. The Ref. (Bazant and Baweja 1999) provides a five levels classification of structures that can serve as simple guidelines for making a decision, when creep and shrinkage analysis is needed and when it is not needed. The recognized levels of structures are as follows:

Level 1: Reinforced concrete beams, frames, and slabs with span under 20m and heights of up to 30m, plain concrete footings, retaining walls.

Level 2. Prestressed beams or slabs of spans up to 20m, high-rise building frames up to 100m high.

Level 3. Medium-span box girder, cable-stayed or arch bridges with spans of up to 80m, ordinary tanks, silos, pavements.

Level 4. Long-span prestressed box-girder, cable-stayed or arched bridges; large bridges built sequentially in stages by joining parts, large gravity, arch or buttress dams, cooling towers, large roof shells, very tall buildings.

Level 5. Record span bridges, nuclear containments and vessels, large offshore structures, large cooling towers, record-span thin roof shells, record-span slender arch bridges.

Full creep and shrinkage analysis is mandatory for the design of structures level 4 and 5 and it is recommended also for the level 3 structures.

5.1 Implementation of Creep and Shrinkage Analysis in ATENA

ATENA software provides a powerful method for creep and shrinkage analysis for most problems from engineering practice. It is based on the so-called cross-sectional approach, meaning that the analysis builds upon creep and shrinkage behavior of the whole cross-section rather than the behavior of individual material points only. The reason for choosing this method is that at this moment, there are available numerous models for predicting creep and shrinkage behavior of a concrete cross-section, whereas there is very low evidence about the same behavior at the material point level. The second reason is that its accuracy suffices for most analyses from engineering practice, and it is much less expansive in terms of computational cost.

5.1.1 Basic Theoretical Assumptions

The implemented creep and shrinkage analysis is based on the assumption of linear creep, which in other words means that the material compliance function $\Phi(t,t')$ and accompanying function for shrinkage $\varepsilon^0(t)$ depends only on material composition, temperature, shape, and time at observation t and at loading t'. It does not depend on stress-strain conditions. In spite of the simplifications, the provided analysis is sufficiently accurate in most practical cases and it is fast and efficient. On the other hand, it is applicable only for structures, where the stress value does not exceed about 60% of the ultimate strength of concrete. For higher load levels, the material

nonlinearity becomes significant and a more elaborate solution has to be employed. The above simplification applies to time-dependent (i.e., long-term) material behavior only. For short-term behavior of the material, model retains its nonlinearity, i.e., it accounts for phenomena such as cracks, plasticity.

The creep and shrinkage analysis is based on the assumption of Stieltjes integral, which is written for the case of 1D analysis in the following form:

$$\varepsilon(t) = \int_{t'}^{t} \Phi(t,t') \frac{\partial \sigma}{\partial \tau} d\tau + \varepsilon^0(t) \tag{4.58}$$

where:

t = observation time,

t' = loading time,

$\sigma(t)$ =stress at the time t,

$\varepsilon^0(t)$ = initial stress-independent strain such as concrete shrinkage,

$\Phi(t,t')$ = compliance function of concrete.

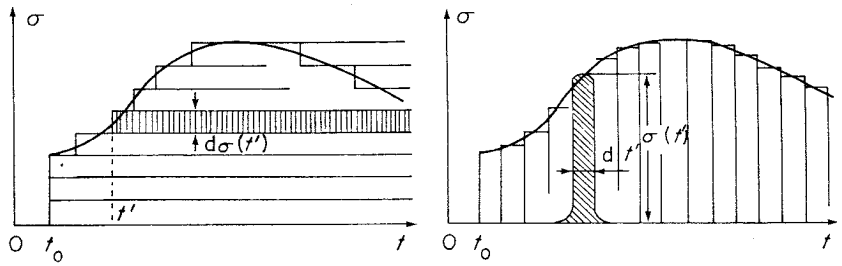


Fig. 5-1 Decomposition of stress history into stress steps (left) or impulses (right).

The sense of Stieltjes integral is given in the above figure.

Equation (4.58) has to be modified for the case of 2 and 3D analyses for practical analyses. This is done below. It is important to note that (4.58) applies to any stress and strain history, and it is defined in incremental form. It means that at a particular time t , stress at $t + \Delta t$ depends only on the current material state at time *t* and stress increment at a time $t + \Delta t$, i.e. $\Delta \bar{\sigma} = \frac{\partial \bar{\sigma}}{\partial \tau} d\tau$.

The final form of the above equations reads:

$$\bar{\varepsilon}(t) = \int_{t'}^{t} \Phi(t,\tau) \left(\mathbf{B}(\bar{\sigma}(\tau)) \frac{\partial \bar{\sigma}}{\partial \tau} + \frac{\partial \mathbf{B}(\bar{\sigma}(\tau))}{\partial \tau} \bar{\sigma}(\tau) \right) d\tau + \bar{\varepsilon}^0(t) \tag{4.59}$$

where:

$\bar{\sigma}(t)$ = is stress vector at a time t, (note the bar atop of a symbol indicates vector),

$\bar{\varepsilon}^0(t)$ = vector of initial strains, such as shrinkage,

$\mathbf{B}(\bar{\sigma}(\tau))$ = matrix accounting for multiaxial stress-strain conditions, including all material short-term nonlinearities.

Notice the way the equation (4.59) is written. Long-term and short-term material behavior is separated. The former is encapsulated in the compliance function $\Phi(t,t')$, whereas the short-term behavior is comprised in the matrix $\mathbf{B}(\bar{\sigma}(\tau))$. This assumption brings significant simplification of the creep and shrinkage analysis, and it is believed that for most practical analysis, the induced inaccuracy is acceptable.

Substituting $t = t' + \delta t,\ \delta t \to 0$ into (4.59) and applying load increment $\Delta\sigma(t') = \sigma(t')$ (i.e., loading from the zero level) at a time t', it can be derived

$$\bar{\varepsilon}(t'+\delta t) = \Phi(t'+\delta t, t')\mathbf{B}(\bar{\sigma}(t'))\bar{\sigma}(t') + \bar{\varepsilon}^{0}(t'+\delta t) \tag{4.60}$$

Comparison of (4.60) with similar equations for constitutive relations for short-term loading conditions, i.e. $t' + \delta t \doteq t'$, yields instantaneous secant material rigidity matrix:

$$\mathbf{D}(t') = \left(\mathbf{B}(\bar{\sigma}(t'))\Phi(t',t')\right)^{-1} \tag{4.61}$$

The matrix $\mathbf{D}(t')$ corresponds to the reciprocal value of the well-known secant Young modulus $E(t')$ in the case of 1D stress-strain conditions. In the case of plane stress conditions, the matrix $\mathbf{B}(\bar{\sigma}(\tau))$ reads (4.62), etc.

$$\mathbf{B} = \begin{bmatrix} 1 & -\nu & 0 \\ & 1 & 0 \\ sym. & & 2(1+\nu) \end{bmatrix} \tag{4.62}$$

5.2 Approximation of Compliance Functions $\Phi(t,t')$ by Dirichlet Series.

Ref. (Bazant and Spencer 1973) and others show that significant improvement of computational efficiency can be obtained if the original material compliance function $\Phi(t,t')$ is during the creep solution approximated by Dirichlet series $\Phi'(t,t')$ as follows:

$$\Phi'(t,t') = \frac{1}{E(t')} + \sum_{\mu=1}^{n} \frac{1}{E_{\mu}(t')}\left(1 - e^{\left(-\frac{t-t'}{\tau_{\mu}}\right)}\right) \tag{4.63}$$

where :

τ_{μ} = are so-called retardation times,

n = number of approximation functions, i.e., this parameter is related to the input parameter number of retardation times.

$E(t')$ = instant Young modulus at the time t',

$E_{\mu}(t')$ =coefficients for the approximation functions.

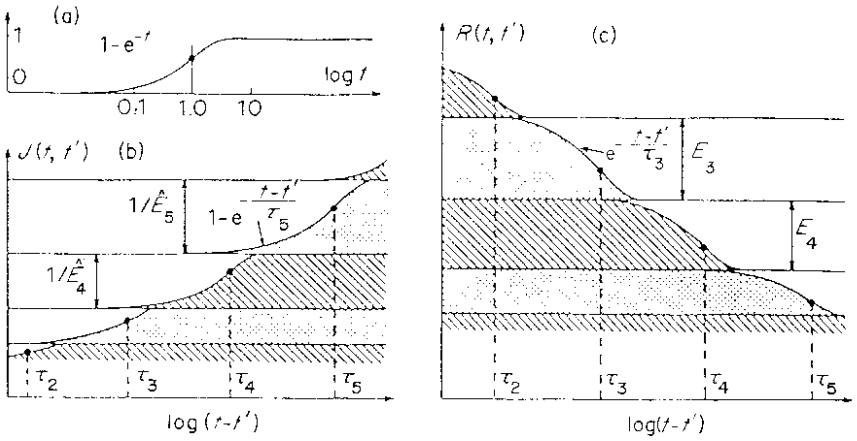


Fig. 5-2 Approximation of compliance (or retardation) function curve at age *t'* at loading by a sum of exponentials used as shape functions of Dirichlet series

The effect of the use of Dirichlet series approximation is depicted in the above figure. A single approximation exponential is drawn in sub-figure (a), while the whole process of decomposition of compliance and retardation curves is depicted in the sub-figures (b), (c), respectively.

The incorporation of the Dirichlet series $\Phi'(t,t')$ brings the following benefits:

- Creep analysis is independent of the material creep prediction model.
- Time integration is exact; hence, fewer temporal increments are necessary.
- Less demand of computer storage needed for storing data from the previous temporal steps of the analysis. It suffices to store data from the previous analysis step only, rather than the complete stresses-strain history of the analyzed structure.

5.3 Step by Step Method

Equation (4.59) (upon substitution (4.63) is solved numerically. The structure is discretized in space by the finite element method (described elsewhere in this document). As for time, the solution is carried out by the Step-by-step method (SBS) (Bazant 1988). The structural behavior is analyzed in several time steps, i.e. in time increments, as it corresponds to (4.59). After some mathematical manipulations (Jendele and Phillips 1992), the final solution equations read:

$$\Delta\bar{\sigma}_r = \tilde{E}_{r-1/2}\left(\mathbf{B}_{r-1/2}\right)^{-1}\left(\Delta\bar{\varepsilon}_r - \Delta\tilde{\bar{\varepsilon}}_r\right) \tag{4.64}$$

$$\bar{\varepsilon}(t_r) = \bar{\varepsilon}_r = \bar{\varepsilon}_{r-1} + \Delta\bar{\varepsilon}_r \tag{4.65}$$

$$\bar{\sigma}(t_r) = \bar{\sigma}_r = \bar{\sigma}_{r-1} + \Delta\bar{\sigma}_r \tag{4.66}$$

$$\frac{1}{\tilde{E}_{r-1/2}} = \frac{1}{E_{r-1/2}} + \sum_{\mu=1}^{n}\left(1-\lambda_{\mu,r}\right)\frac{1}{E_{\mu,r-1/2}} \tag{4.67}$$

$$\lambda_{\mu_r} = \left(1 - e^{-\frac{\Delta t_r}{\tau_\mu}}\right)\frac{\tau_\mu}{\Delta t_r} \tag{4.68}$$

$$\Delta\overline{\overline{\varepsilon}}_r = \sum_{\mu=1}^{n}\left[1 - e^{-\frac{\Delta t_i}{\tau_\mu}}\right]\overline{\varepsilon}^*_{\mu_{r-1}} + \Delta\overline{\varepsilon}^0_r \tag{4.69}$$

$$\overline{\varepsilon}(t_r) = \overline{\varepsilon}_r = \overline{\varepsilon}_{r-1} + \Delta\overline{\varepsilon}_r \tag{4.70}$$

$$\overline{\varepsilon}^*_{\mu_r} = e^{-\frac{\Delta t_r}{\tau_\mu}}\overline{\varepsilon}^*_{\mu_{r-1}} + \frac{\lambda_{\mu_r}\widetilde{E}_{r-1/2}\left(\Delta\overline{\varepsilon}_r - \Delta\overline{\overline{\varepsilon}}\right)}{E_{\mu_{r-1/2}}} \tag{4.71}$$

In the above the following notation is used:

r = identification of temporal increments, $r \in \langle 1..N \rangle$, where N is number of time increments for the analysis,

$\Delta t_r = t_r - t_{r-1}$ = time increment,

$\Delta\overline{\sigma}_r = \overline{\sigma}_r - \overline{\sigma}_{r-1}$ = stress increment in time t_r,

$\overline{\varepsilon}^*_{\mu} = \overline{\varepsilon}^*_{\mu}(t_r)$ =internal variables at time t_r,

$\overline{\varepsilon}^0_r = \overline{\varepsilon}^0(t_r)$ = shrinkage at time t_r,

$E_{r-1/2} = \frac{1}{2}\left(E(t_r) + E(t_{r-1})\right)$ = constant average secant Young modulus at time incremenent Δt_r,

$E_{\mu r-1/2} = \frac{1}{2}\left(E_\mu(t_r) + E_\mu(t_{r-1})\right) = \frac{1}{2}\left(E_{\mu_r} + E_{\mu_{r-1}}\right)$ = constant average value of Dirichlet coefficient E_μ at Δt_r,

$\mathbf{B}_{r-1/2} = \frac{1}{2}\left(\mathbf{B}(t_r) + \mathbf{B}(t_{r-1})\right)$ = average value of the matrix B at Δt_r.

Equation (4.64) thru (4.71) defines all necessary relations to complete the creep and shrinkage analysis in ATENA. Of course, they are supplemented by relations used by the short-term material constitutive model, i.e., equations for calculating the matrix **B.**

At each time increment, a typical short-term alike analysis is carried. The difference between the short-term analysis and the described analysis of one step of the creep and shrinkage is that the latter one uses especially adjusted Young modulus $\widetilde{E}_{r-1/2}$ and initial strain increments $\Delta\overline{\overline{\varepsilon}}_r$ to account for creep and shrinkage. After each step, these have to be updated. It involves mainly update of λ_{μ_r} and $\Delta\overline{\overline{\varepsilon}}_r$. With these values, a new $\widetilde{E}_{r-1/2}$ is calculated and the next temporal analysis step is carried out.

5.4 Integration and Retardation Times

Appropriate selection of retardation and integration times is of crucial importance for accurate and efficient creep and shrinkage analysis. The choice of retardation times has a direct impact on the accuracy of approximation of an original compliance function by Dirichlet series, see

Equation (4.63) and Fig. 5-2, whilst the choice of integration times affects the accuracy of the approximation of loading function of the structure, see Equation (4.58) and Fig. 5-1. If the number of times is too low, some important features of concrete behavior can be disregarded. The opposite extreme, i.e., using too many retardation or integration times results in worthless lengthy solution of the problem.

The ATENA software respects recommendation in (Bazant and Whittman 1982). Retardation times are spread uniformly in $\log(t)$ space and they are automatically calculated as follows:

$$\begin{aligned} &\tau_\mu = \tau_i + 10^{\frac{\mu-i-1}{m}}(\tau_{i+1}-\tau_i), \quad \mu = 1,2..n,\ i = 0,\ \tau_i = 0 \\ &\tau_\mu = 10^{\frac{\mu-1}{m}}(\tau_1), \quad \mu = 1,2..n \end{aligned} \tag{4.72}$$

In the above m is the number of retardation times per $\log(t)$ unit, $m \geq 1$. By default, this constant is in ATENA set to 1. If required, a more detailed approximation is possible, i.e., any value $m > 1$ can be used. In the program, this parameter is input as a number of retardation times per time unit in logarithmic scale. For a typical concrete creep law, a certain optimal value can be determined, and it is independent of a structure being analyzed. Note, however, that the value depends on the choice of time units.

Example: If the retardation times parameter is set to 2, the creep law will be approximated by two approximation points for the time interval between 0 - 1 day, two points for the interval 1 - 10 days, then two points for 10 - 100 days, etc.

Therefore, the proper values will depend on the choice of time units. If the time unit is a day, the recommended value is 1 - 2.

Start time τ_1 must be chosen sufficiently low, so that Dirichlet series can account for processes in very young concrete right after its loading has been applied. As a default, ATENA uses the expression $\tau_1 = 0.1\,t'$.

As for the upper limit for τ_μ , it is required:

$$\tau_n \geq \frac{t}{2} \tag{4.73}$$

The above limits are applicable for the case when the coefficients $E_\mu(t')$ of the Dirichlet series in (4.63) are calculated by the Least-square method (Jendele and Phillips 1992).

ATENA also supports an alternative way of calculation of the coefficients $E_\mu(t')$ of the Dirichlet series in (4.63). In this case, Inverse Laplace transformation (Bazant and Xi 1995) is used instead. This method requires $\tau_1 \to 0$, typically 1E-3 and

$$\tau_n \geq t \tag{4.74}$$

Comparing the above two approaches, it can be said that the Least-square method yields approximation of the compliance function at discrete times, whereby Inverse transformation is based on continuous approach. In some cases, the Least-square method results in better convergence behavior; however it sometimes suffers from numerical problems during

calculation due to an ill-posed problem for solution of $E_\mu(t')$. It is left to experience and engineering judgment to decide, which of the method is more appropriate for a particular solution.

Integration times or sample times t_r are calculated in a similar way. In this case, the times are uniformly spread in $\log(t-t')$ time scale. They are generated starting from the 1st loading time t'. Hence, we can write

$$t_r = t_i + 10^{\frac{r-i-1}{l}}\left(t_{i+1} - t_i\right), \qquad t' = t_i \tag{4.75}$$

where $l \geq 2$ is the number of time increments per unit of $\log(t-t')$ and $t_{i+1} \doteq t' + 0.1 = t_i + 0.1$ days. Each new major load increment or decrement causes the generation procedure (4.75) must start again from small time increments. This parameter defines the number of time steps that the program will use to integrate the structural behavior. Creep or other nonlinear effects will cause a redistribution of stresses inside the structure. In order to properly capture such processes, a sufficiently small time steps are needed. Its definition depends on the type of the analyzed structure as well as on the choice of time units. For typical reinforced concrete structures and for the time unit being a day, it is recommended to set this parameter to 2. This will mean that for each load interval longer then 1 day, two sub-steps will be added. For a load that is interval longer than 10 days, 4 sub-steps will be added. For an interval longer than 100 days, it will be 6 sub-steps, etc.

The creep and shrinkage analysis in ATENA requires that the user set number of retardation times m and the number of time increments l per unit of log time, (unless the default values are OK). He/she also specifies time span, i.e., τ_1 and τ_n. Then, retardation times are generated, i.e., an appropriate command is issued. It follows to set stop time of the analysis. Usual input data describing structural shape, material etc. are given thereafter; however, there are three important differences from the time-independent analysis:

1. Material model for concrete contains data for long-term as well as for short-term material model.
2. Step data must include information about the time at which the step is applied.
3. It is recommended to input data for all intended load time steps prior to the steps are executed. It helps the generation of integration (intermediate) times

Intermediate time steps, i.e., times t_r as well retardation times are generated automatically. The analysis proceeds until the stop time is reached. If no stop time is specified, it is assumed to be the time of the last load step. If the time span for retardation times does not cover step load times, the solution is aborted, giving an appropriate error message.

5.5 Creep and Shrinkage Constitutive Model

In the above sections, it was silently assumed that the long-term part of the material model, i.e., compliance function $\Phi(t,t')$ and shrinkage function $\bar{\varepsilon}_r^0$ for concrete, is known and it was shown how it is utilized within creep and shrinkage analysis. It is the primary intention of this section to

describe what long-term creep and shrinkage prediction models are implemented in ATENA and how they should be used.

Generally speaking, ATENA applies no restriction on the kind and shape of both $\Phi(t,t')$ and $\bar{\varepsilon}_r^0$, as it adopts the SBS method solution algorithm, in which compliance function is approximated by Dirichlet series. Hence, the most widely recognized creep prediction models could be implemented.

The CCStructureCreep module currently supports the following models:

1. CCModelACI78 (ACI_Committee_209 1978), recommended by ACI,
2. CCModelCEB_FIP78 (Beton 1984), recommended by CEB committee, by now already obsolete,
3. CCModelB3 (Bazant and Baweja 1999), developed by Bazant and Al Manaseer in 1996, very efficient model recognized world-wide,
4. CCModelB3Improved, same as the above, improved to account for temperature history, probably the best model available in ATENA,
5. CCModelCSN731202, model developed by CSN 731202 Code of practice in Czech Republic,
6. CCModelBP1_DATA (Bazant and Panula 1978; Bazant and Panula 1978; Bazant and Panula 1978; Bazant and Panula 1978), relatively efficient and complex model; now it is superseded by CCModelBP_KX or CCModelB3,
7. CCModelBP2_DATA (Bazant and Panula 1978), simplified version of the above model,
8. CCModelBP_KX (Bazant and Kim 1991; Bazant and Kim 1991; Bazant and Kim 1991; Bazant and Kim 1991), a powerful model with accounts for humidity and temperature history etc., for practical use it may-be too advanced,
9. CCModelGeneral general model into which experimentally obtained $\Phi(t,t')$ and $\bar{\varepsilon}_r^0$ function can be input.
10. CCModelEN1992- Eurocode model for creep, (EN1992),
11. CCModelFIB_MC2010- creep model based on CEB-FIP FIB Model Code 2010.

The following data summarized input parameters for the supported models. Note that some models allow improved prediction based on laboratory data. If it is the case, the model input the corresponding experimentally measured values. Also, some models can account for material point history of humidity $h(t)$ and temperature $T(t)$. Again, a model supports this feature if it can input adequate data.

Table 5.5-1 : List of material parameters for creep and shrinkage prediction – definition and description

Parameter name	Description	Units	Default
Concrete. type	Type of concrete according to ACI. Type 1 is Portland cement etc. Types 1,3 accepted for static analysis, types 1-4 accepted for transport analysis.		1
Cement class	Type of cement, see e.g. http://www.cis.org.rs/en/cms/about-cement/standardization-of-cement : Strength classes of cement Cements are according to standard strength grouped into three classes, they being: • Class 32,5 • Class 42,5 • Class 52,5 Three classes of early strength are defined for each class of standard strength: • Class with ordinary early strength – N • Class with high early strength – R • Class with low early strength – L Class L can be applied only on CEM III cements.		42,5
Aggregate	Type of aggregate. One of BASALTDENSELIMESTONE, QUARTZITE, LIMESTONE, SANDSTONE , LIGHTWEIGHTSANDSTONE		QUARTZITE
Thickness V / S	Cross section thickness defined as ration of section's volume to surface	length	0.0767m
Strength f_{cyl28}	Material cylindrical strength in compression at time 28 days	stress	35.1MPa
Strength $f_{cyl0,28}$	Strength at onset of nonlinear behaviour in compression at time 28 days	stress	Constant from the base material

Fracture energy $G_{f,28}$	Fracture energy at time 28 days	stress	Constant from the base material
Strength f_{t28}	Material tensile strength at time 28 days	stress	Constant from the base material
Young m. E_{28}	Short-term material Young modulus at 28 days, i.e. inverse compliance at 28.01 days loaded at 28 days	stress	$F(f_{cyl28})$
Ambient humid. h	Ambient relative humidity. Accepted range (0.4..1).		0.78
Ratio a_c	Total aggregate/cement weight ratio.		7.04
Ratio w_c	Water/cement weight ratio.		0.63
Ratio a_s	Total aggregate/find sand weight ratio. $a_s = s_a^{-1}$.		2.8
Ratio s_a	Fine/total aggregate weight ratio. $s_a = a_s^{-1}$		0.4
Ratio g_s	Coarse gravel/fine aggregate weight ratio.		1.3
Ratio s_c	Fine aggregate/cement weight ratio.		1.8
Shape factor	Cross section shape factor. It should be 1, 1.15, 1.25, 1.3, 1.55 for slab, cylinder, square prism, sphere, cube, respectively.		1.25
Slump	Result of material slump test.	length	0.1 m
Air content	Material volumetric air content.	%	5
Cement mass	Weight of cement per volume of concrete	mass/ length^3	320kg/ m^3
Concr. density	Material density used to evaluate strength and Young modulus at 28 days..	mass/ length^3	2125kg/ m^3
Curing type	Curing conditions. It can be either in water (i.e. WATER) or air under normal temperature (i.e. WATER) or steamed curing (i.e. STEAM).		AIR

Thermal expansion coefficient α_T		Thermal expansion coefficient α_T	1/temperature	Constant from he base material
End of curing		Time at beginning of drying, i.e. end of curing.	days	7
$\varepsilon_{a,\infty}$		Autogenous shrinkage at infinity time, (typically negative!) $\varepsilon_a = \varepsilon_{a,\infty}(0.99 - \min(0.99, h_{a,\infty}) \tanh\left(\frac{t - t_s}{\tau_a}\right)$	-	0
τ_a		Half-time of autogenous shrinkage.	days	30
t_s		Time of final set of cement	days	5
$h_{a,\infty}$		Final self-desiccation relatibe humidity	-	0.8
I/D	Current time t	Current time	days	0
	Load time t'	Load time	days	0
Tot.water loss w		Total water loss (up to zero humidity and infinite time). It is measured in an oven in a laboratory and it is used to enhance prediction of shrinkage infinite $t_{sh2\infty}$ (Bazant and Baweja 1999). This value is in turn used to elaborate drying creep and shrinkage prediction of the model. If it is not specified, the model prediction enhancement is not activated. It can be used, if water loss *w(t)* are input as well.	kg	N/A
Improvem.	Water loss *w(t)*	Water losses at time *t*; measured at a laboratory. It is used to enhance drying creep and shrinkage prediction. See also description of total water loss *w*.	kg	N/A
	Shrink. $\bar{\varepsilon}^0(t)$	Measured shrinkage at time *t*. It is used to enhance drying creep and shrinkage prediction. See also description of total water loss *w*.		N/A
	Compl. $\Phi(t,t')$	Measured material compliance at time *t*. It is used to improve overall creep and shrinkage prediction of the model.	1 /stress	N/A

Hist.	Humidity $h(t)$	History of humidity in a material point. Value at time *t*. Some material models can use these values to account for real temporal humidity and temperature conditions. Although the data can be input manually, i.e. to group material points with similar humidity and temperature history into a group and dedicate a distinct material for that group, it is prepared for full automatic processing being currently in development. It will automatically link heat and humidity transport analysis with the static analysis using one of available creep and shrinkage prediction model. Applicable range (0.4..1).		N/A
	Temperat. $T(t)$	History of temperature in a material point. See also description of $h(t)$	Celsia	
Direct	Compl. $\Phi(t,t')$	Measured compliance at time t loaded at time t'. This and the next two parameters should be used, if known (measured) compliance functions are to be employed in ATENA creep and shrinkage analysis. Hence, no prediction is done and the given data are only used to calculate the parameters of Dirichlet series approximation.	1/ stress	
	Shrink. $\bar{\varepsilon}^0(t)$	Measured shrinkage at time t . See the parameter above.		N/A
	Strength $f_{cyl}(t)$	Measured shrinkage at time t . See the parameter above		

Table 5.5-2: Input parameters needed by individual creep and shrinkage prediction models

Model name	B3	B3-impr	BP-KX	CEB	ACI	CSN	BP1	BP2	General	EN 1992	MC 2010
Model No.	3	4	8	2	1	5	6	7	9	10	11
Concrete. Type	x	x	x		x	x	x	x			
Cement class										x	x
Aggregate										x	x
Thickness S / V	x	x	x	x	x	x	x	x		x	x
Strength f_{cyl28}	x	x	x	x	x	x	x	x		x	x
Strength $f_{cyl0,28}$		x								x	x
Fracture energy $G_{f,28}$		x								x	x
Strength f_{t28}		x								x	x
Young m. E_{28}	x	x	x	x		x				x	x
Ambient humid. h	x	x	x	x	x	x	x	x		x	x
Ratio a_c	x	x	x		x		x	x			
Ratio w_c	x	x	x		x		x	x			
Ratio a_s											
Ratio s_a							x	x			
Ratio g_s							x	x			
Ratio s_c							x	x			
Shape factor	x	x	x				x	x			
Slump					x						
Air content					x						
Cement mass							x				
Concr. density	x	x	x		x					x	x

Curing type		x	x	x		x		x	x			
End of curing		x	x	x	x	x	x	x	x		x	x
Thermal expansion coefficient α_T			x								x	x
$\varepsilon_{a,\infty}$			x									
τ_a			x									
t_s			x									
$h_{a,\infty}$			x									
	Current time t	x	x	x	x	x	x	x	x	x	xx	x
	Load time t'	x	x	x	x	x	x	x	x	x	xx	x
Tot.water loss w		x	x			x						
	Water loss $w(t)$	x	x									
	Shrink. $\bar{\varepsilon}^0(t)$	x	x	x	x	x	x	x	x		xx	x
	Compl. $\Phi(t,t')$	x	x	x								
	Humidity $h(t)$		x	x			x				xx	x
	Temperat. $T(t)$		x	x			x				xx	x
	Compl. $\Phi(t,t')$									x		
	Shrink. $\bar{\varepsilon}^0(t)$									x		
	Strength $f_{cyl}(t)$									x		

The above parameter "Concrete type" actually referes to a cement type according to the ACI classification. It used in the creep analysis. The following table brings description of widely recognized cement types. Note that only types 1,3 are supported in Atena static analysis. The

transport analysis in Atena recognizes types 1-4. The remaining types are described just for information.

Table 5.5-3: Cement types according to ACI classification

ATENA Concrete type	Cement type	Description
1	I and Type IA[6]	General purpose cements suitable for all uses where the special properties of other types are not required.
2	II and Type IIA[6]	Type II cements contain no more than 8% tricalcium aluminate (C_3A) for moderate sulfate resistance. Some Type II cements meet the moderate heat of hydration option of ASTM C 150.
3	III and Type IIIA[6]	Chemically and physically similar to Type I cements except they are ground finer to produce higher early strengths.
4	IV	Used in massive concrete structures where the rate and amount of heat generated from hydration must be minimized. It develops strength slower than other cement types.
5	V	Contains no more than 5% C_3A for high sulfate resistance.
6	IS (X)[7]	Portland blast furnace slag cement
7	IP (X)[7]	Portland-pozzolan cement.
8	GU[8]	General use
9	HE[8]	High early strength
10	MS[8]	Moderate sulfate resistance
11	HS[8]	High sulfate resistance
12	MH[8]	Moderate heat of hydration

[6] Air-entraining cements

[7] Blended hydraulic cements produced by intimately and uniformly intergrinding or blending two or more types of fine materials. The primary materials are portland cement, ground granulated blast furnace slag, fly ash, silica fume, calcined clay, other pozzolans, hydrated lime, and pre-blended combinations of these materials. The letter "X" stands for the percentage of supplementary cementitious material included in the blended cement. Type IS(X), can include up to 95% ground granulated blast-furnace slag. Type IP(X) can include up to 40% pozzolans.

[8] All portland and blended cements are hydraulic cements. "Hydraulic cement" is merely a broader term. ASTM C 1157, *Performance Specification for Hydraulic Cements*, is a performance specification that includes portland cement, modified portland cement, and blended cements. ASTM C 1157 recognizes six types of hydraulic cements.

13	LH[8]	low heat of hydration

5.6 References

ACI_COMMITTEE_209 (2008). Prediction of Creep, Shrinkage and Temperature Effects in Concrete Structures. Detroit, 209R-92, ACI.

BATHE, K. J. (1982). Finite Element Procedures in Engineering Analysis. Englewood Cliffs, New Jersey 07632, Prentice Hall, Inc.

BAZANT, Z. AND T. SPENCER (1973). "Dirichlet Series Creep Function for Aging Concrete." ASCE Journal of Engineering and Mechanical Division: 367-387.

BAZANT, Z. P. (1988). Mathematical Modeling of Creep and Shrinkage of Concrete. New York, John Wiley & Sons.

BAZANT, Z. P. AND S. BAWEJA, EDS. (1999). Creep and Shrinkage Predicition Model for Analysis and design of Concrete Structures: Model B3. Creep and Shrinkage of Concrete, ACI Special Publicatino.

BAZANT, Z. P. AND J. K. KIM (1991). "Improved Prediction Model for Time-Dependent Deformation of Concrete: Part 1- Shrinkage." Materials and Structures **24**: 327-345.

BAZANT, Z. P. AND J. K. KIM (1991). "Improved Prediction Model for Time-Dependent Deformation of Concrete: Part 2- Basic Creep." Materials and Structures **24**: 409-421.

BAZANT, Z. P. AND J. K. Kim (1991). "Improved Prediction Model for Time-Dependent Deformation of Concrete: Part 3- Creep at Drying." Material and Structures **25**: 21-28.

BAZANT, Z. P. AND J. K. KIM (1991). "Improved Prediction Model for Time-Dependent Deformation of Concrete: Part 4- Temperature Effects." Material and Structures **25**: 84-94.

BAZANT, Z. P. AND L. PANULA (1978). "Practical Prediction of Time-dependent Deformations of Concrete; Part 1: Shrinkage." Material and Structures **11 (65)**: 301-316.

BAZANT, Z. P. AND L. PANULA (1978). "Practical Prediction of Time-dependent Deformations of Concrete; Part 3: Drying Creep." Material and Structures **11 (65)**: 415-423.

BAZANT, Z. P. AND L. PANULA (1978). "Practical Prediction of Time-dependent Deformations of Concrete; Part 4: Temperature Effect on Basic Creep." Material and Structures **11 (66)**: 424-434.

BAZANT, Z. P. AND L. PANULA (1978). "Practical Prediction of Time-dependent Deformations of Time-dependent Deformation of Concrete; Part 2: Basic Creep." Material and Structures **11 (65)**: 317-328.

BAZANT, Z. P. AND L. PANULA (1978). Simplified Prediction of Concrete Creep and Shrinkage from Strength and Mix. Struct. Engng. Report No. 78-10/6405. Evanston, Illinoins, Northwestern University, Dep. of Civ. Engng.

BAZANT, Z. P. AND F. H. WHITTMAN (1982). Creep and shrinkage in Concrete Structures. New York, John Wiley & Sons.

BAZANT, Z. P. AND Y. XI (1995). "Continous Retardation Spectrum for Solidification Theory of Concrete Creep." Journal of Engineering Mechanics **121**(2): 281-287.

BETON, C. E.-I. D. (1984). CEB Design Manual on Structural Effects on Time Dependent Behaviour of Concrete. Saint Saphorin, Switzerland, Georgi Publishing Company.

HAGEMAN, L. AND D. YOUNG (1981). Applied Iterative Methods. New York, Academic Press.

JENDELE, L. AND D. V. PHILLIPS (1992). "Finite Element Software for Creep and Shrinkage in Concrete." Computer and Structures **45 (1)**: 113-126.

REKTORYS, K. (1995). Přehled užité matematiky. Prague, Prometheus.

SEAGER, M. K. AND A. GREENBAUM (1988). A SLAP for the Masses, Lawrence Livermore National Laboratory.

6 DURABILITY ANALYSIS[9]

The durability analysis in ATENA can currently assess the deterioration of structures due to carbonation and chlorides ingress. It is available for static and creep analyses. At each time step, an appropriate 1D transport analysis is carried out to investigate how far the pollution (i.e., carbonation and/or chlorides) penetrate from loaded surfaces inside the structure. The main results of the analyses are induction times, i.e., times at which the pollution concentration reaches critical values that are already for the structure unacceptable (e.g., the reinforcement corrosion begins etc.). They are always given with respect to time $t_0 = 0$. In addition, pollution concentration at times (corresponding to the individual steps) is also computed.

Note that static analysis in ATENA typically does not care about time (or more precisely, each analysis step increments the structural age by unit time). At each step, it yields a sort of artificial age of the structure. Hence, if the durability analysis is carried out, this artificial age must be somehow mapped onto real structural age. It is done in ATENA with the help of a multilinear function. Such a function corresponds to loading functions used to define variable BCs and it is input in exactly the same way.

The following text describes the theory behind the 1D transport analysis of the carbonation and chlorides pollution, and, in the end, some information regarding the transport parameters is given.

The service life of a structure t_l usually has the form of

$$t_l = t_c + t_i + t_p + t_r \tag{4.76}$$

where t_c is the construction phase, t_i initiation (induction) period, t_p propagation period, and t_r post-repair period.

We aim at predicting the initiation period without going into propagation or post-repair phases. Carbonation and chloride ingress are two leading mechanisms contributing to reinforcement corrosion. Both of them are described further. The initiation phase ends with the beginning of reinforcement corrosion. Fig. 6-1 brings a more detailed description of initiation and propagation phases and their relationship to concrete events. Prediction of the initiation period represents a preventive measure that is affected above all by concrete cover thickness, concrete composition, and environment. It makes sense to change the design at the beginning rather than mitigating reinforcement corrosion later. Acceleration of carbonation and chloride ingress on crack appearance is taken into account.

[9] Not available in ATENA version 5.1 and older. Development/testing implementation of CARBONATION, CHLORIDES, and ASR in version 5.3.

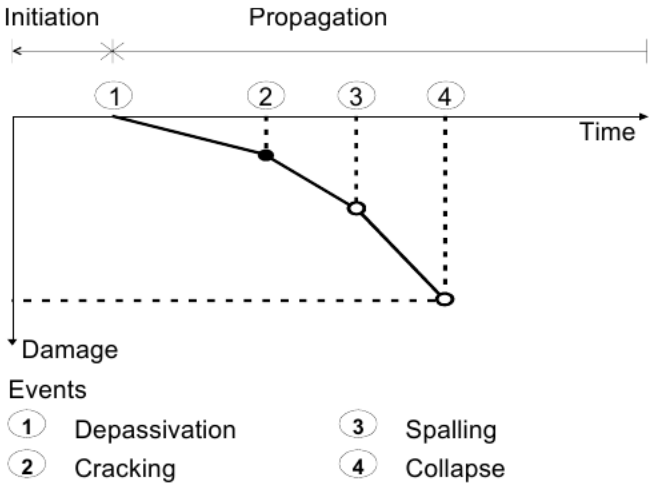


Fig. 6-1 Important events in service life (III 2000).

6.1 Carbonation

Carbonation depth of a sound (uncracked) concrete reads (Papadakis and Tsimas 2002)

$$x_c = \sqrt{\frac{2D_{e,CO2}CO_2}{0.218(C+kP)}}\sqrt{t} = A_1\sqrt{t} \tag{4.77}$$

where x_c is the carbonation depth, $D_{e,CO2}$ is the effective diffusivity for CO_2, C is the Portland cement content in kgm^{-3}, $k\in$<0.3,1.0> is the efficiency factor of supplementary cementitious material (SCM-slag, silica, fly ash), P is the amount of SCM in kgm^{-3}, CO_2 is the volume fraction of CO_2 in the atmosphere taken as 3.6e-4 and t is the time of exposure. The effective diffusivity in m^{2}s^{-1} is given by the empirical equation (Papadakis and Tsimas 2002)

$$D_{e,CO2} = 6.1\cdot 10^{-6}\left(\frac{(W-0.267(C+kP))/1000)}{\frac{C+kP}{\rho_c}+\frac{W}{1000}}\right)^3 (1-RH)^{2.2} \tag{4.78}$$

where W is the water content in concrete in kgm^{-3}, ρ_c is the cement density in kgm^{-3} assumed as 3150 kgm^{-3} and RH is the relative humidity of ambient air. Eqs. (4.77)(4.78) allow predicting either carbonation depth or induction time of uncracked concrete. Relative humidity must be higher than 0.50 for carbonation to proceed.

Cracked concrete leads to faster carbonation. This acceleration is given in the form (Kwon and Na 2011)

$$x_c(t) = (2.816\sqrt{w}+1)A_1\sqrt{t} \tag{4.79}$$

where w is the crack width in mm, A_1 is the carbonation velocity according to Eq.(4.77). Eq. (4.79) allows computing carbonation depth and induction time. Note that crack 0.3 mm increases carbonation depth by a factor of 2.54. This also means that induction time is 6.46 times shorter compared to a sound concrete.

In reality, cracks may grow during any service time. Thus, Eq. (4.79) needs to be recast to incremental form. An increment of carbonation depth in a given time step Δt is evaluated from the total derivative by differentiating Eq. (4.79)

$$\Delta x_c(t) = \frac{\left(2.816\sqrt{w_{i+1}}+1\right)A_1}{2\sqrt{t_{i+0.5}}}\Delta t + \frac{2.816 A_1 \sqrt{t}}{2\sqrt{w_{i+0.5}}}\Delta w \tag{4.80}$$

where w_{i+1} is the crack width at the end of the time step, $t_{i+0.5}$ is the mid-time. It is assumed that nonzero Δw at a frozen time t has no effect on carbonation depth; thus the term Δw can be left out. Eq. (4.80) allows predicting either carbonation depth or induction time of gradually cracking concrete.

6.1.1 Example of Carbonation

Let us consider first a regular concrete made from ordinary Portland cement, w/b=0.45, C=400 kgm^{-3}, W=202.5 kgm^{-3}, P=50 kgm^{-3}. The supplementary cementitious material is fly ash with almost zero calcium content hence k=0.5. Concrete is exposed to relative humidity 0.60. Consider a concrete cover of 30 mm. A crack is always introduced at the beginning of the exposure.

The second concrete is made from ordinary Portland cement, w/b=0.45, C=200 kgm^{-3}, W=90 kgm^{-3}, P=0 kgm^{-3}. Table 6.1-1 compares both concretes in terms of induction time.

Crack width (mm)	Induction time for concrete w/b=0.45, C=400 kgm^{-3}, P=50 kgm^{-3} (years)	Induction time for concrete w/b=0.45, C=200 kgm^{-3}, P=0 kgm^{-3} (years)
0	246	157
0.1	69.9	44.5
0.2	49.2	31.4
0.3	39.1	24.9

Table 6.1-1. Induction time for carbonation, two concretes, cover thickness 30 mm.

6.2 Chlorides

Implemented model for chloride ingress is based on (Kwon, Na et al. 2009). Let us consider 1D transient problem of chloride ingress in concrete with initially free chloride content

$$C(x,t) = C_S\left[1 - erf\left(\frac{x}{2\sqrt{D_m(t)\,f(w)t}}\right)\right] \tag{4.81}$$

where C_S is the chloride content at surface in kgm^{-3}, D_m is the averaged diffusion coefficient at time t in mm^2 s^{-1}, x is the position from the surface in mm, and $f(w)$ gives acceleration by cracking and equals to one for a crack-free concrete. C_s and C can be related to concrete volume or to binder volume; however, the units must be kept consistently through the computation.

The diffusion coefficient $D(t)$ is assumed to decrease over time t according to the power-law

$$D(t) = D_{ref}\left(\frac{t_{ref}}{t}\right)^m \tag{4.82}$$

where m is a decay rate (sometimes called an age factor). If m=0, a constant value of $D(t)=D_{ref}$ is recovered. This model was proposed by (Collepardi, Marcialis, et al. 1972). Nowadays, it became clear that this assumption is too conservative and is not generally recommended. The mean diffusion coefficient D_m is obtained by averaging $D(t)$ over time of interest

$$D_m(t) = \frac{1}{t}\int_0^t D_{ref}\left(\frac{t_{ref}}{\tau}\right)^m d\tau = \frac{D_{ref}}{1-m}\left(\frac{t_{ref}}{t}\right)^m, t < t_R \tag{4.83}$$

$$D_m(t) = D_{ref}\left[1+\frac{t_R}{t}\left(\frac{m}{1-m}\right)\right]\left(\frac{t_{ref}}{t_R}\right)^m, t \geq t_R \tag{4.84}$$

where t_R is the time when diffusion coefficient is assumed to be constant and is generally taken as 30 years. t_{ref} corresponds to the time when the diffusion coefficient was measured. Fig. 6-2 shows the characteristic evolution of diffusion coefficients over time.

The mean diffusion coefficient increases when cracks are present in the concrete. Based on recent results, the following scaling function is proposed (Kwon, Na et al. 2009)

$$f(w) = 31.61w^2 + 4.73w + 1 \tag{4.85}$$

where w stands for crack width in mm. The crack width 0.3 mm increases the mean diffusion coefficient by a factor of 5.26. In reality, crack width evolves, and incremental solution needs to be formulated. The mean coefficient $D_{m,w}(t)$ incorporating crack width is evaluated from a crack increment

$$D_{m,w}(t) = D_m(t)\int_0^{\bar{w}} 63.22w + 4.73\,dw \approx D_m(t)\sum_{i=0}^{n}\left[63.22\left(w(t_i) + \frac{w(t_{i+1}) - w(t_i)}{2}\right) + 4.73\right]\{w(t_{i+1}) - w(t_i)\} \tag{4.86}$$

If last values of $f(w)$ and w are stored, Eq. (4.86) can be evaluated only in the actual time step. This speeds up the solution.

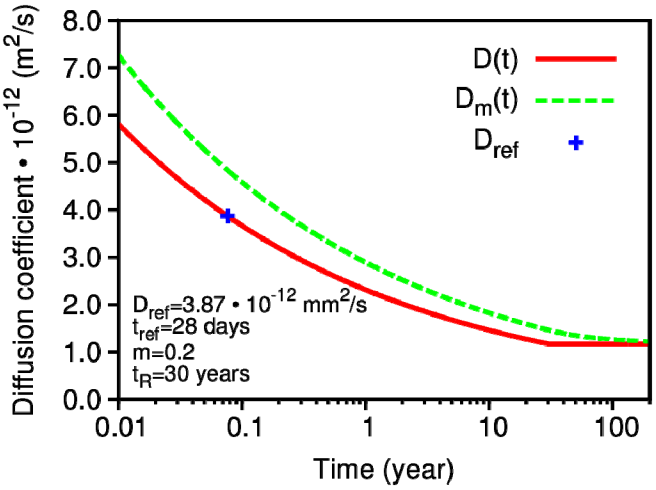


Fig. 6-2. Evolution of actual and mean diffusion coefficients for standard concrete, based on data from (Kwon, Na et al. 2009).

6.3 Diffusion coefficient for chlorides

Proper determination of diffusion coefficient is not a trivial subject, considering various concretes, cement types, models, and exposure conditions. (Papadakis 2000) presented a model for estimating intrinsic effective diffusivity for concretes made from blended cements; however, recalculation to D_a is not straightforward. DuraCrete model (III 2000) provides useful data for estimating apparent diffusion coefficient in the form

$$D_a(\mathrm{t}) = k_e k_c D_{Cl}(t_0)\left(\frac{t_0}{t}\right)^m \gamma_{Da} \tag{4.87}$$

where $k_e \in <0.27,3.88>$ is the environment factor, $k_c \in <0.79,2.08>$ is the curing factor, $D_{cl}(t_0)$ is the measured diffusion coefficient determined at time t_0, $m \in <0.2,0.93>$ is the age factor and $\gamma_{Da} \in <1.25,3.25>$ is the partial factor. In our notation, $D_a(t)=D_m(t)$ and $t_0=t_{ref}$.

To our opinion, the most relevant and well-documented field data come from 10 years exposure tests (Luping, Tang et al. 2007). Fig. 6-3 shows the apparent diffusion coefficient in dependence of water-binder ratio. In this particular case, t_{ref}=10 years, m is unknown, $D_{ref}=(1-m)D_a$, t_R can be assumed as 30 years.

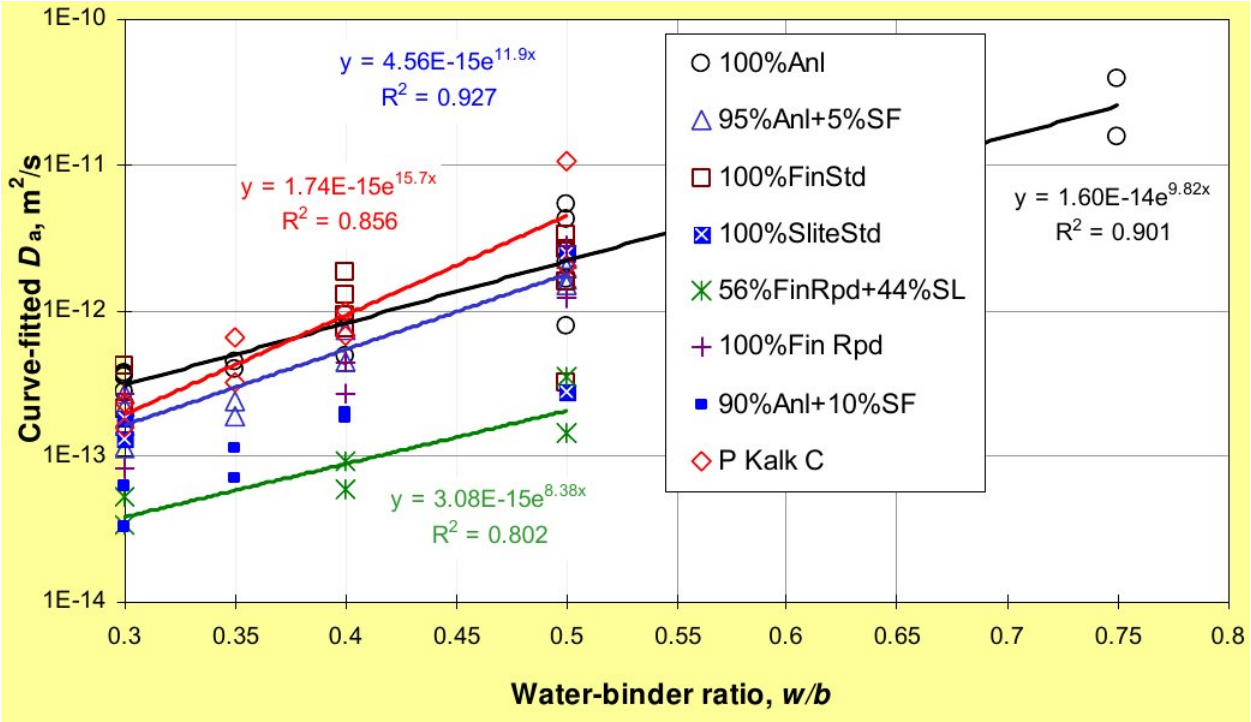


Fig. 6-3. Fitted apparent diffusion coefficients from 10-years exposure of concrete (Luping, Tang et al. 2007).

The next figure shows the apparent diffusivity coefficient at 10 years from Fig. 6-3. They can be used as a starting point for estimating D_{ref}.

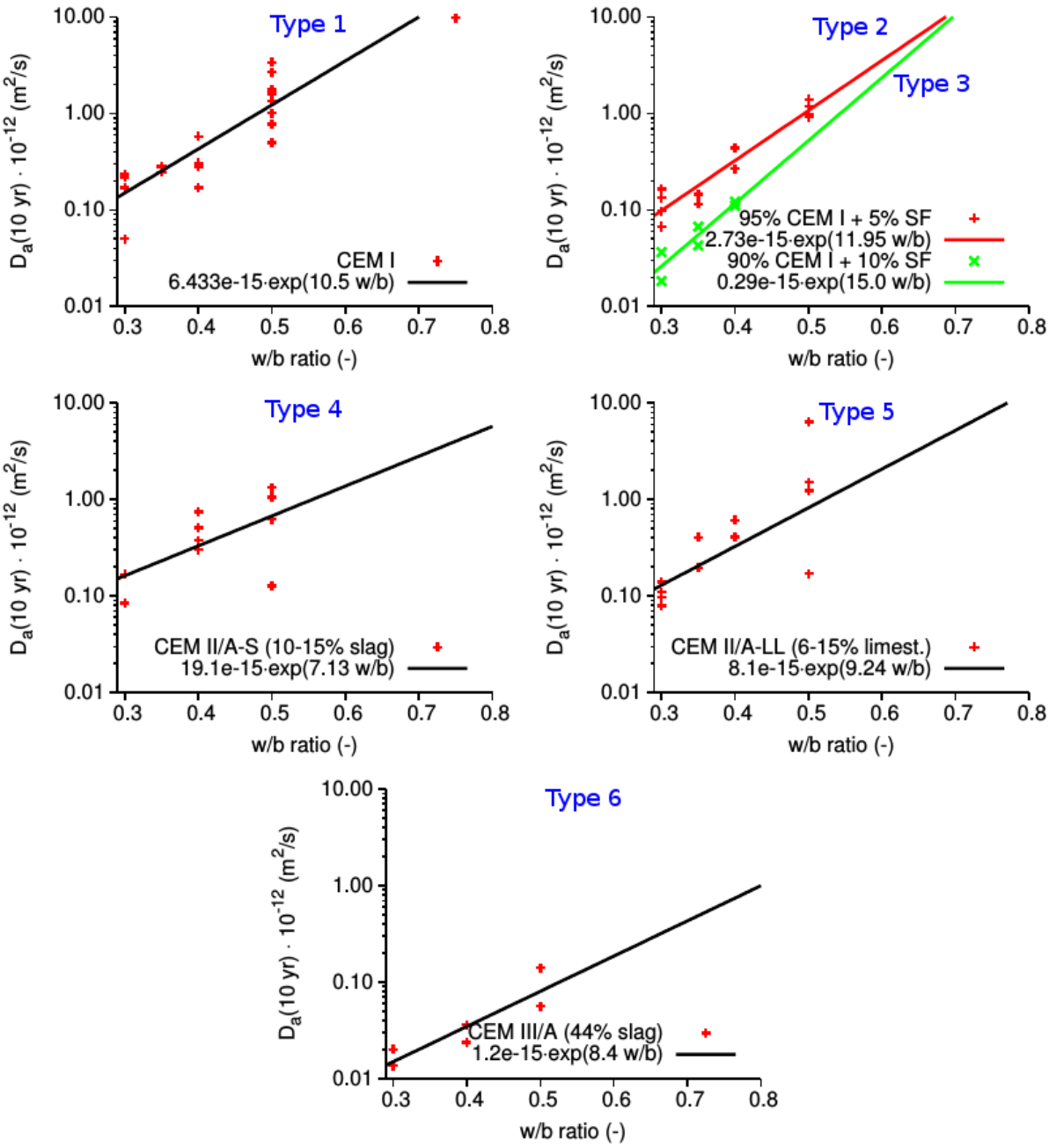


Fig. 6-4. Apparent diffusion coefficients from 10-years exposure of concrete (Luping, Tang et al. 2007).

3.2. Example of chloride ingress

Let us consider regular concrete made from ordinary Portland cement, w/b=0.45. According to Fig. 6-3, D_a is about 2e-12 m^2s^{-1} at t_{ref}=10 years. According to the Duracrete model, the age factor for concrete submerged in salt water corresponds to m=0.30 (Table 8.6 in DuraCrete). In such case, $D_{ref}=(1-m)D_a$=1.4e-12 m^2s^{-1}. Fig. 6-5 shows the evolution of diffusion coefficients for this particular case.

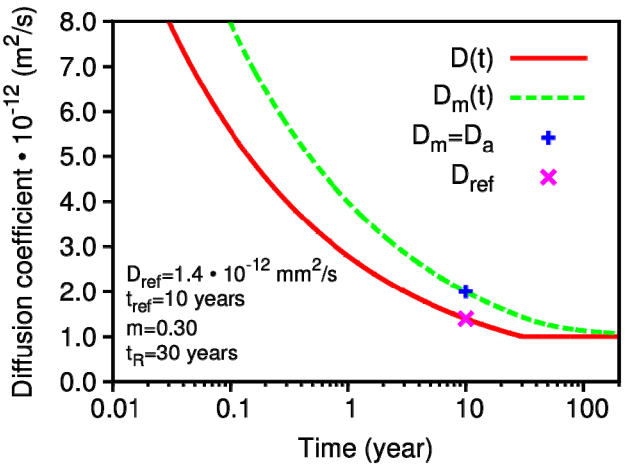


Fig. 6-5. Evolution of diffusion coefficients for chlorides in an example.

Let us assume characteristic value C_s 10.3% of chlorides per binder for submerged concrete without further reductions (Table 8.5 in DuraCrete). The critical level for corrosion is 1.85 % per binder (Table 8.7 in DuraCrete). The concrete cover is taken as 100 mm. Computed induction time according to Eq. (4.81) is summarized in Table 6.3-1. Crack width is considered since the beginning of the exposure.

Table 6.3-1 Induction time for chloride corrosion of submerged concrete, in dependence on original crack width.

Crack width (mm)	Induction time (years)
0	74.58
0.1	36.02
0.2	15.70
0.3	7.76

6.4 MODELS for PROPAGATION PHASE

6.4.1 Carbonation during propagation phase

The corrosion rate for the carbonation depends on the corrosion current density i_{corr} [μA/cm^2], which ranges between 0.1-10 (passive corrosion-high corrosion) and depends on the quality and the relative humidity of the concrete (Page CL, 1992). This model predicts the amount of corroded steel during the whole propagation period t_p. The corrosion rate is based on Faraday's law (Rodriguez, 1996), determined as follows:

$$\dot{x}_{\text{corr}}(t) = 0.0116 i_{\text{corr}}(t) \tag{4.88}$$

where $\dot{x}_{\text{corr}}$ is the average corrosion rate in the radial direction [μm/year], i_{corr} is corrosion current density [μA/cm^2], and t is the calculated time after the end of the induction period [years].

By integration of Eq. (1), it is obtained the corroded depth for 1D propagation:

$$x_{\text{corr}}(t) = \int_{t_{ini}}^{t} 0.0116 i_{\text{corr}}(t) R_{\text{corr}} \, \mathrm{d}t \tag{4.89}$$

where x_{corr} is the total amount of corroded steel in radial direction [mm] and R_{corr} is parameter, depends on the type of corrosion [-]. For uniform corrosion (carbonation) R_{corr} = 1, for pitting corrosion (chlorides) R_{corr} = <2; 4> according to (Gonzales at.al., 1995) or R_{corr} = <4; 5.5> according to (Darmawan &, 2007).

Effective bar diameter for both types of corrosion is obtained from:

$$d(t) = d_{ini} - \psi 2 x_{corr}(t) \tag{4.90}$$

where $d(t)$ is the evolution of bar diameter in time t, d_{ini} is initial bar diameter [mm], ψ is uncertainty factor of the model [-], mean value ψ = 1 and x_{corr} is the total amount of corroded steel according to (2).

6.4.2 Chloride ingress during propagation phase

The corrosion rate for chlorides is more complicated because it is affected by the concentration of chlorides in the concrete. Calculation of corrosion current density was formulated by Liu and Weyer's model (Liu, Weyers, 1998):

$$i_{corr} = 0.926 * \exp\left[7.98 + 0.7771 \ln(1.69 C_t) - \frac{3006}{T} - 0.000116 R_C + 2.24 t^{-0.215} \right] \tag{4.91}$$

where i_{corr} is corrosion current density [μA/cm^2], C_t is total chloride content [kg/m^3 of concrete] on reinforcement which is determined from 1D nonstationary transport, T is temperature at the depth of reinforcement [K] and R_c is ohmic resistance of the cover concrete [Ω] (Liu, 1996) and t is time after initiation [years]:

$$R_C = \exp\left[8.03 - 0.549 \ln(1 + 1.69 C_t) \right] \tag{4.92}$$

The average corrosion rate in radial direction is determined further when plugging(4.93),(4.94) to (1). The total amount of corroded steel in radial direction stems from (2) and the effective bar diameter from (3).

6.4.3 Cracking of concrete cover

The cracking of concrete cover for both carbonation and chlorides can be estimated from DuraCrete model, which provides realistic results (DuraCrete, 2000). The critical penetration depth of corroded steel $x_{corr,cr}$ is formulated as:

$$x_{corr,cr} = a_1 + a_2 \frac{C}{d_{ini}} + a_3 f_{t,ch} \tag{4.95}$$

where parameter a_1 is equal 7.44e-5 [m], parameter a_2 is equal 7.30e-6 [m], a_3 is [-1.74e-5 m/MPa], C is cover thickness of concrete [m], d_{ini} initial bar diameter [m], $f_{t,ch}$ is characteristic splitting tensile strength of concrete [MPa].

6.4.4 Spalling of concrete cover

The critical penetration depth of corroded steel $x_{corr,sp}$ for both carbonation and chlorides is calculated from (DueaCrete, 2000) as:

$$x_{corr,sp} = \frac{w^d - w_0}{b} + x_{corr,cr} \tag{4.96}$$

where parameter b depends on the position of the bar (for top reinforcement 8.6 μm/μm and bottom 10.4 μm/ μm), w^d is critical crack width for spalling (characteristic value 1 mm), w_0 is the width of initial crack (known from previous ATENA computation) and $x_{corr,cr}$ depth of corroded steel at the time of cracking [m].

After spalling of concrete cover, corrosion of reinforcement takes place in direct contact with the environment. To determine the rate of corrosion of reinforcement after spalling, (Spec-net, 2015) gives rates of reinforcement corrosion.

Table 2: Corrosion rates of steel under atmospheric exposition

Corrosivity zone (ISO 9223)		Typical environment	Corrosion rate for first year (μm/yr)	
Category	Description		Mild steel	Zinc
C1	Very low	Dry indoors	≤1,3	≤0,1
C2	Low	Arid/Urban inland	>1,3 a ≤25	>0,1 a ≤0,7
C3	Medium	Coastal and industrial	>25 a ≤50	>0,7 a ≤2,1
C4	High	Calm sea-shore	>50 a ≤80	>2,1 a ≤4,2
C5	Very High	Surf sea-shore	>80 a ≤200	>4,2 a ≤8,4
CX	Extreme	Ocean/Off-shore	>200 a ≤700	>8,4 a ≤25

6.5 Alkali-Aggregate Reaction

6.5.1 Introduction of alkali-aggregatea model for concrete

In most concrete, aggregates are more or less chemically inert. However, some aggregates react with the alkali hydroxides in concrete, causing expansion and cracking over a period of many years. This alkali-aggregate reaction has two forms: alkali-silica reaction (ASR) and alkali-carbonate reaction (ACR).

Alkali–silica reaction (ASR), one of those common deleterious mechanisms, consists of a chemical reaction between "unstable" silica mineral forms within the aggregate materials and the alkali hydroxides (Na, K–OH) dissolved in the concrete pore solution. It generates a secondary alkali-silica gel that induces expansive pressures within the reacting aggregate material(s) and the adjacent cement paste upon moisture uptake from its surrounding environment, thus causing micro cracking, loss of material's integrity (mechanical/durability), and, in some cases, functionality in the affected structure.

Several aggregate types in common use, particularly those with a siliceous composition, may be attacked by the alkaline pore fluid in concrete. This attack, essentially a dissolution reaction, requires a certain level of moisture and alkalis (leading to high pH) within the concrete to take place. During the reaction, a hygroscopic gel is produced. When imbibing water, the gel will swell and thus cause expansion, cracking, and in the worst case, disruption of the concrete (Lindgart 2012).

Thus, the degree of reaction of an aggregate is a function of the alkalinity of the pore solution. For a given aggregate, a critical lower pH-value exists below which the aggregate will not react. Consequently, ASR will be prevented by lowering pH of the pore solution beneath this critical level where the dissolution of alkali-reactive constituents (silica) in the aggregates will be strongly reduced or even prevented, as discussed in (Rodriguez at.al, 1996). No "absolute" limit is defined because the critical alkali content largely depends on the aggregate reactivity [3], but from many experimental tests we can estimate threshold value (Lindgart 2012), (Poyet , 2003).

Many studies carried out over the past few decades have shown that ASR can affect the mechanical properties of concrete as a "material." Usually, ASR generates a significant reduction in tensile strength and modulus of elasticity of concrete. These two properties are much more affected than compressive strength, which begins to decrease significantly only at high levels of expansion.

Several ASR models were developed over the years to predict expansion and damage on both ASR affected materials (microscopic models) (Multon at.al., 2009), (Bazant, Steffens, 2009), (Comby-Perot, 2009) and ASR affected structures/structural elements (macroscopic models) (Ulm at.al., 1999), (Saouma, Perotti,2006), (Comi, Fedele, Perego, 2009). The first group has a goal of modeling both the chemical reactions and the mechanical distress caused by ASR or even the coupling of the two phenomena. The second group aims at understanding the overall distress of structures/structural concrete elements in a real context, simulating their likely in situ behavior (Farage et al.,2000) seems to have finally bridged the gap between scientific rigor and practical applicability to real structures.

In terms of mechanical effects, it is known that ASR expansions occur over long time periods. During this process, ASR-affected concretes are subjected to a progressive stress built up that is very likely to cause creep on the distressed materials.

AAR depends on the availability of three factors: alkalis liberated from cement during hydration, siliceous minerals present in certain kinds of aggregates, and water. Several microscopic and random factors are involved in AAR expansion, such as concrete porosity, amount and location of reactive regions in the material, and permeability (Farage et al.,2000). These parameters, added to concrete's intrinsic heterogeneity, turn simulating the AAR expansion into a rather complex task.

Even though the AAR process has not been well explained so far, the commonly accepted theory for describing it is two distinct phases that need to be considered: gel formation and water absorption by the gel, causing expansion. According to this mechanism, the reaction does not always lead to expansion. As long as there is enough void space to be filled by the gel, i.e., pores and cracks, concrete volume remains unchanged.

Due to the lack of a model, which is able to incorporate effects of relative humidity, alkali/silica content in the mixture, ambient temperature, authors suggest to combine ASR kinetics proposed by (Ulm et. al., 1999) with the influence of moisture, published by (Léger et al., 1996) and influence of alkali/silica content proposed by Multon et al.

Implementation of modeling expansion due to ASR consists of modeling engeinstrains in time-steps *t* on the entire structure. Function for volumetric eigenstrain reads

$$\varepsilon_{ASR}(t) = \varepsilon_{cal}^{\infty} \xi(t) \mathrm{F_M} \tag{4.97}$$

where $\varepsilon_{cal}^{\infty}$ is the volumetric strain of ASR swelling at infinity time, $\xi(t) \in \langle 0,1 \rangle$ is the chemical extent of ASR, and $\mathrm{F_M}$ is the coefficient reflecting moisture influence. It is described later in the text. In the case of varying the relative humidity, eq. (4.97) changes to the incremental form, for time t_i

$$\varepsilon_{ASR}(t_i) = \varepsilon_{ASR}(t_{i-1}) + \varepsilon_{cal}^{\infty} \left(\xi(t_i) - \xi(t_{i-1}) \right) \mathrm{F_M} \left(\frac{t_i + t_{i-1}}{2} \right) \tag{4.98}$$

6.5.2 Model for ASR kinetics

For the complete 3D constitutive model, we consider the first-order reaction

$$1 - \xi = t_{\mathrm{c}}(\theta, \xi)\, \dot{\xi} \tag{4.99}$$

where $t_{\mathrm{c}}(\theta, \xi) = k_d / A_0$ is the characteristic time. It has been found that t_{c} depends on temperature $\theta[K]$ and the ASR extent ξ. Referring to (4.99) the implementation of the chemoelastic material law in the constitutive laws is relatively straightforward and a suitable integration scheme is given in (Ulm ea., 1999).

Consider an isothermal stress-free ASR expansion test carried out at constant temperature $\theta = \theta_0$. In this test, the volumetric strain ε_{ASR} is recorded as a function of time that and ASR extent is calculated as

$$\xi(t) = \frac{\varepsilon_{ASR}(t)}{\varepsilon_{ASR}(\infty)} \tag{4.100}$$

For macroscopically stress-free sample, (4.99) in (4.100) yields

$$\begin{aligned}\varepsilon_{ASR}(\infty)(1-\xi) &= \varepsilon_{ASR}(\infty)t_c(\theta,\xi)\frac{\dot{\varepsilon}_{ASR}(t)}{\varepsilon_{ASR}(\infty)}\\ \varepsilon_{ASR}(\infty)-\xi\varepsilon_{ASR}(\infty) &= t_c(\theta,\xi)\dot{\varepsilon}_{ASR}(t)\\ \varepsilon_{ASR}(\infty)-\varepsilon_{ASR}(t) &= t_c(\theta,\xi)\dot{\varepsilon}_{ASR}(t)\end{aligned} \tag{4.101}$$

With $\varepsilon_{ASR}(t)$ and $\dot{\varepsilon}_{ASR}(t)$ being measurable functions of time, the characteristic time t_c can be determined from a stress-free expansion test. In a recent extensive series of stress-free expansion tests carried out at different constant temperatures(Larive, 1998), t_c has been found to depend on both temperature $\theta[K]$ and reaction extent ξ[-] in the form

$$t_c = \tau_c(\theta)\lambda(\xi,\theta) \tag{4.102}$$

$$\lambda(\xi,\theta) = \frac{1+\exp\left[-\tau_L(\theta)/\tau_C(\theta)\right]}{\xi+\exp\left[-\tau_L(\theta)/\tau_C(\theta)\right]} \tag{4.103}$$

In this experimentally determined kinetics function, $\tau_c(\theta)$ is a characteristic time [day] and $\tau_L(\theta)$ is a latency time [day]. The use of (4.103),(4.102) in (4.101) yields after integration

$$\xi(t) = \frac{1-\exp(-t/\tau_C)}{1+\exp(-t/\tau_C+\tau_L/\tau_C)} \tag{4.104}$$

For variable temperature, cracking etc., it is difficult to solve for $\xi(t)$ analytically and numerical integration is needed. A suitable solution scheme is derived in (Ulm at.al., 2006), which is implemented in in ATENA. Fig. 6-6 shows the shape of (4.100), together with the time constants, τ_c τ_C and τ_L, which stand for the characteristic time and the latency time of ASR swelling, respectively. Furthermore, proceeding as in physical chemistry (Atkins, 1994), we explore the temperature dependence of the time constants τ_c τ_C and τ_L from stress-free expansion tests carried out at different constant temperatures. The plots of $\ln(\tau_c)$ τ_C nd $\ln(\tau_L)$ against $1/\theta$ τ_C are given in Fig. 6-7. It is remarkable that the experimental values align (almost) perfectly along a straight line, matching the Arrhenius concept.

$$\tau_C(\theta) = \tau_C(\theta_0)\exp\left[U_c\left(\frac{1}{\theta}-\frac{1}{\theta_0}\right)\right] \tag{4.105}$$

$$\tau_L(\theta) = \tau_L(\theta_0)\exp\left[U_L\left(\frac{1}{\theta}-\frac{1}{\theta_0}\right)\right] \tag{4.106}$$

where

$$U_C = 5400 \pm 500\,K;\; U_L = 9400 \pm 500\,K \tag{4.107}$$

It is explored (Atkins, 1994) that the temperature dependence of the time constants τ_c τ_C nd τ_L carried out at different constant temperatures (23, 33, 38, and 58 °C), see Fig. 6-6. Default values are $\tau_C(311{,}15\text{K})$ ays and $\tau_L(311.15°\text{C}) = 145$ days [20], see Fig. 6-8. , Fig. 6-9. According to Larive's experimental data from water-saturated tests [14] $\tau_C(288{,}15\text{K})$ days and $\tau_L(288{,}15\text{K})$ ays, $\tau_C(281{,}15\text{K})$ ys and $\tau_L(288{,}15\text{K})$ ays. Under drying conditions, the values for $\tau_L\tau_L$ roughly increase by a factor of 4; and τ_C by 2.5 (Larive , 1998), (Ulm at.al, 1999)

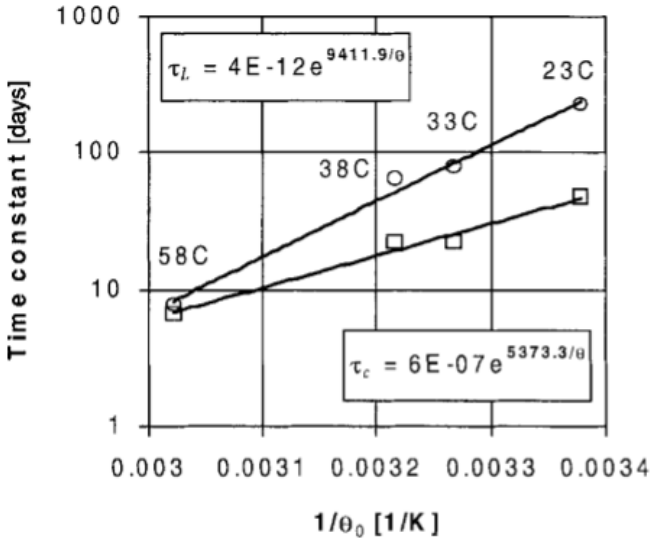


Fig. 6-6. Larive's test data of temperature dependency of ASR time constants τ_C nd τ_L Slope of trendlines represents activation energy constants Uc = 5,400 K and UL = 9,700 K.

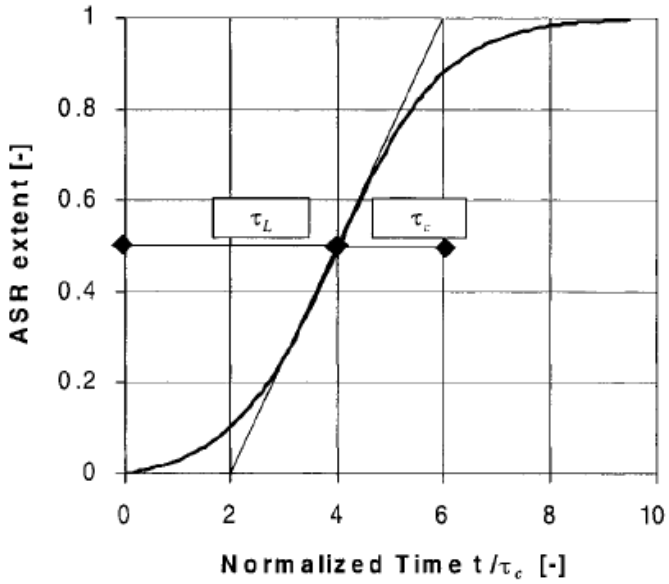
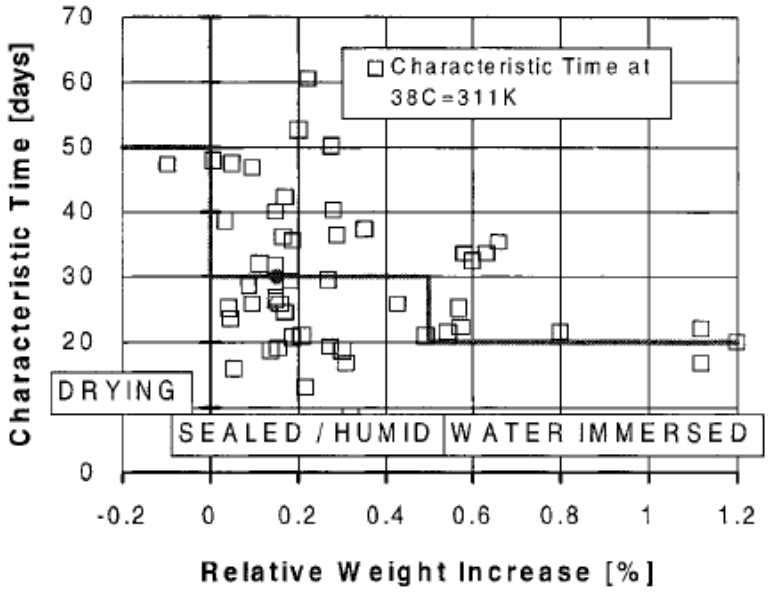


Fig. 6-7. Definition of Latency Time τ_L d Characteristic Time τ_C Normalized Isothermal Expansion



Curve $\xi\ \epsilon(t)/\epsilon(\infty)$

Fig. 6-8. Parameter Analysis of Characteristic Time τ_C (311 K) of ASR Swelling with Regard to Hydral Ambient Conditions, reproduced from 0.

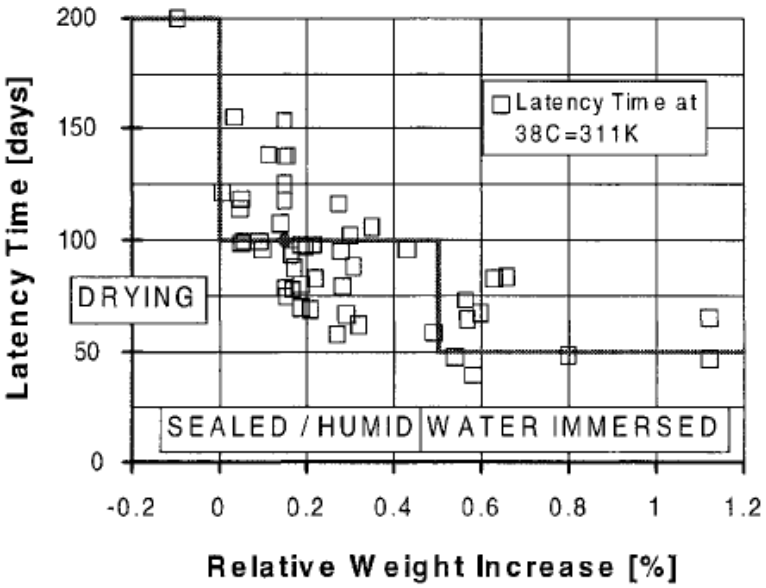


Fig. 6-9. Parameter Analysis of Latency Time τ_L (311 K) of ASR Swelling with Regard to Hydral Ambient Conditions, reproduced from 0.

6.5.3 Prediction of ASR swelling $\varepsilon_{cal}^{\infty}$

$\varepsilon_{cal}^{\infty}$ [-] is the predicted volumetric expansion at infinity time obtained by model proposed by (Multon et al., 2008). It is calculated based on reactive aggregates, amount of reactive silica in the aggregates, and value of measured stress-free expansion test done in Poyet's study (Lindgart , 2012) on samples containing reactive particles only. $\varepsilon_{cal}^{\infty}$ is defined as follows

$$\varepsilon_{cal}^{\infty}(t) = s \cdot p \cdot AC \cdot \varepsilon_F \frac{A_C}{A_R} \tag{4.108}$$

where ε_F [m^3/kg] is measured ASR strain expansion per kg of aggregate in m^3 of the concrete mixture on samples containing reactive particles only with enough sufficiency of alkali. Typically it ranges in 8.93e-7 ... 1.34e-5 [m3/kg]. See Table 3 for more details. A_C kg/m^3 Na_2O_{eq}] and A_R kg/m^3 Na_2O_{eq}] are amounts of consumed and required alkali, respectively. AC is total aggregate content in [kg/m^3]. One of the main assumptions of the model is that the maximum expansion of mortar is achieved if there is enough alkali to react with all the reactive silica of the mixture. This amount of required alkali content A_R kg/m^3 Na_2O_{eq}] is defined as

$$A_R = r \cdot s \cdot p \cdot AC \tag{4.109}$$

where *s* is the proportion of quantity of soluble silica [-], *p* is the proportion of reactive aggregate [-]. *r* states for the amount of required alkali per kg of reactive silica, and it is a constant value *r* = 15.4 %. Value A_C s defined as min A_R (, A_A s the available amount of alkali for ASR reaction. A_A s defined as the difference between the initial amount of available alkali A_T kg/m^3 Na_2O_{eq}] and alkali content threshold A_0 kg/m^3 Na_2O_{eq}] when ASR reaction starts.

$$A_A = A_T - A_0 \tag{4.110}$$

It should be noted that this model does not consider any alkali flow through boundaries inside the structure during the service life. By default, A_0 s equal to 3.7 kg/m^3 Na_2O_{eq} (Poyet, 2003), but other values in the range of 3 – 5 kg/m^3 Na_2O_{eq} can be found in the literature (Lindgart, 2012)

Table 3: Mixtures and ASR expansions of mortars studied by (Poyet, 2003) and (Multon, 2008). F1-F3 are size fractions 80 µm-3.15 mm.

Non-reactive sand (%)			Reactive sand (%)			Measured ASR expansion
F1	F2	F3	F1	F2	F3	(%)
0	50	25	25	0	0	0.003
25	25	25	0	25	0	0.06
25	50	0	0	0	25	0.06
0	25	25	25	25	0	0.045
25	25	0	0	25	25	0.08

Value of *p* depends on the mix ratio of reactive aggregate. Value *s* depends on amount of reactive silica in aggregates, moreover common values are: *p* = 11,1% (Multon, 2008) 0or 9,4% and 12,4% (Multon, 2009).

6.5.4 Influence of moisture F_M

Approximately 75% relative humidity (RH) within concrete is necessary to initiate significant expansion, which is assumed to vary linearly between 75% RH and 100% RH as shown in Fig. 6-10.

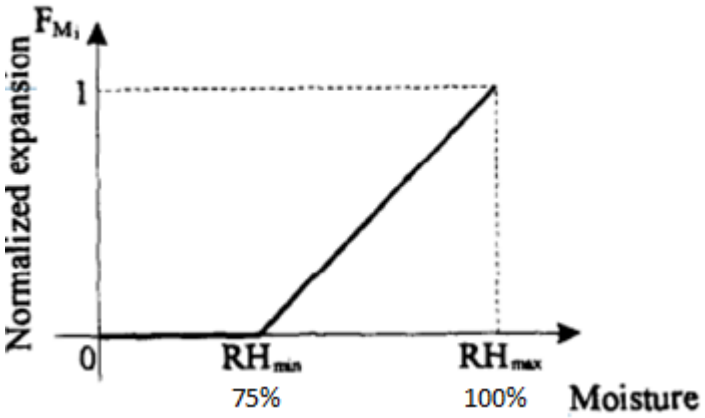


Fig. 6-10. Parameter Factor of RH influencing ASR concrete expansion, reproduced from (Multon, Toutlemonde, 2010).

The coefficient F_M eflects influence of moisture h. The function for FM is approximated as

$$F_M(h) = \frac{1}{1-h_{\min}}(h - h_{\min}) \tag{4.111}$$

where h_{min} is relative humidity threshold where ASR begins to appear, 0.75 by default. Other variables will be explained in further text.

6.5.5 ASR for 3D conditiions

Expansion of free concrete specimens due to ASR has been summarized in (Červenka, Jendele, Šmilauer, 2016). It predicts ASR under unrestrained conditions, i.e., under free expansion. The expansion model takes into account reaction kinetics, alkali content, reactive amount of aggregates, relative humidity, and temperature. The model has been validated on 4 examples found in the literature.

Degradation of material due to ASR reaction (Saouma, 2016, eqs. 18,19)

$$E(t,\theta) = E_0\left[1-(1-\beta_E)\xi(t,\theta)\right] \tag{4.112}$$

$$f_t(t,\theta) = f_{t,0}\left[1-(1-\beta_f)\xi(t,\theta)\right] \tag{4.113}$$

$$G_f(t,\theta) = G_{f,0}\left[1-(1-\beta_G)\xi(t,\theta)\right] \tag{4.114}$$

where $\beta_{E,f,G}$ are residual values of E/E_0, f/f_0, G_f/G_{f0}. Default values are $\beta_E = 0.1$, $\beta_f = 0.6$ (Esposito, Hendriks, 2012) and $\beta_G = 0.6$ is estimated.

The general equation for the incremental volumetric AAR strain is given by (Saouma, 2016, (5))

$$\dot{\varepsilon}_V(t) = \dot{\varepsilon}_I(t) + \dot{\varepsilon}_{II}(t) + \dot{\varepsilon}_{III}(t) = \underbrace{\Gamma_t \Gamma_c\left(\overline{\sigma}, f_c'\right) F_M(h)\dot{\xi}(t)\varepsilon_{cal}^{\infty}}_{\text{Only considered in implementation}} + \Gamma_t \dot{\Gamma}_c\left(\overline{\sigma}, f_c'\right) F_M(h)\xi(t)\varepsilon_{cal}^{\infty} + \ldots \dot{\Gamma}_t, \dot{\overline{\sigma}}, \dot{f}_c', \dot{F}_M, \ldots \quad (4.115)$$

where Γ_c reflects the effect of compressive stresses (Saouma, 2016, eq. 10), Γ_t accounts for the influence of tensile cracking (assumed here as 1), F_M is the effect of relative humidity, which is already accounted for in (4.111) and equals to one. (4.115) considers further only the most relevant first term and is rewritten in incremental form as

$$\begin{aligned} &\Delta\varepsilon_V(t_i) = \Gamma_t \cdot \Gamma_c\left(\overline{\sigma}_{i-1}, f'_{c_{i-1}}\right) \cdot F_M\left(h_{\overline{i}}\right) \cdot \varepsilon_{cal}^{\infty} \cdot \left(\xi(t_i) - \xi(t_{i-1})\right) \\ &\overline{i} = (i + (i-1))/2 \end{aligned} \quad (4.116)$$

Reduction Γ_c due to compressive stress is considered as follows:

$$\Gamma_c = \begin{cases} 1 & if\ \overline{\sigma} \geq 0 \quad Tension \\ 1 + \dfrac{e^{\beta}\overline{\sigma}}{1 + \left(1 - e^{\beta}\right)\overline{\sigma}} & if\ \overline{\sigma} < 0 \quad Compresion \end{cases} \quad (4.117)$$

$$\overline{\sigma} = \frac{\sigma_I + \sigma_{II} + \sigma_{III}}{3\left|f_c'\right|}$$

where the shape factor β is -2 by default (Saouma, 2016, Tab.2) and f_c' is the compressive strength.

Under constrained conditions, ASR expansion develops depending on the stress state. It is known that compressive stress beyond approximately -10 MPa stops ASR expansions, which needs to be reflected for strain redistribution into the principal directions. Similarly to (Saouma, 2016, Fig. 5), weight factors are assigned to three directions. Let us assume that directions of principal stresses σ_I, σ_{II}, σ_{III} are known. Expansion is then assigned to each principal stress direction according to the weight factors W_1', W_2', W_3'. When compressive stress reaches -0.3 MPa, the weight factor decreases until maximum stress -10 MPa is reached in that direction. This situation is depicted in Fig. 6-11.

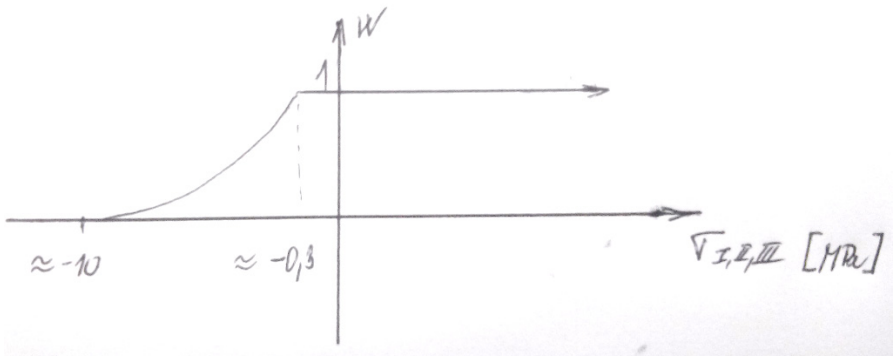


Fig. 6-11. Weight factor for ASR expansion

For compressive stress σ_i under -0.3 MPa, the following decay function is used, according to (Leger, Coté, Tinawi, 1995), where $\sigma_L \approx$ -0.3 MPa and $\sigma_u \approx$ -10 MPa, see Fig. 6-11. :

$$W_i(\sigma_i) = \begin{cases} \left[1 - \dfrac{1}{\log\left(\dfrac{\sigma_u}{\sigma_L}\right)} \log\left(\sigma_i / \sigma_L\right)\right] & for\ \sigma_i < -0.3MPa \\ 1 & for\ \sigma_i \geq -0.3MPa \end{cases} \tag{4.118}$$

Weight factors need to be normalized as

$$W_i = \frac{W_i'}{\sum_{i=1}^{3} W_i'} \tag{4.119}$$

Three principal strains from ASR are assigned as

$$\Delta\varepsilon_{ASR,i} = W_i \cdot \Delta\varepsilon_V(t_i) \tag{4.120}$$

This new approach simplifies the procedure outlined by (Saouma, 2016, Fig. 5) where several stress state cases were treated individually.

6.5.6 Validation on free expansion

The following Fig. 2-12 and Fig. 6-13 validate experimental data for free expansion. The following material parameters were used, summarized in Table 6.5-4.

Variable	*Symbol*	*Value*	*Source*
REQUIRED ALKALI PER REACTIVE SILICA	r	15.4 %	(Multon, Cyr, Sellier, Leklou, & Petit, 2008)

PROPORTION REACTIVE SILICA	s	21.8 %	(Multon, Cyr, Sellier, Leklou, & Petit, 2008)
PROPORTION REACTIVE PARTICLES IN SAND	p	30 %	(Multon, Cyr, Sellier, Leklou, & Petit, 2008)
SAND MASS	AC	833 kg/m^3	(Kagimoto, Yasuda, & Kawamura, 2014)
ASR MEASSURED ASR STRAIN	ε_F	0.0525 %/kg	(Poyet, 2003)
AMOUNT OF REQUIRED ALKALI	A_R	8.39 kg/m^3	(Poyet, 2003)
TOTAL ALKALI IN MORTAR for Ca-5.4 (for Ca-9.0)	A_T	5.4 (9) kg/m^3	(Kagimoto, Yasuda, & Kawamura, 2014)
THRESHOLD ALKALI IN CONCRETE	A_0	3.7 kg/m^3	(Poyet, 2003)
CHARACTERISTIC TIME	τ_C	20 day	
LATENCY TIME for Ca-5.4 (for Ca-9.0)	τ_L	55 (45) day	
ELASTIC MODULUS	E	27 GPa	(Kagimoto, Yasuda, & Kawamura, 2014)
COMPRESSIVE STRENGTH	fc	26 MPa	(Kagimoto, Yasuda, & Kawamura, 2014)

Table 6.5-4. Summarized parameters for validation.

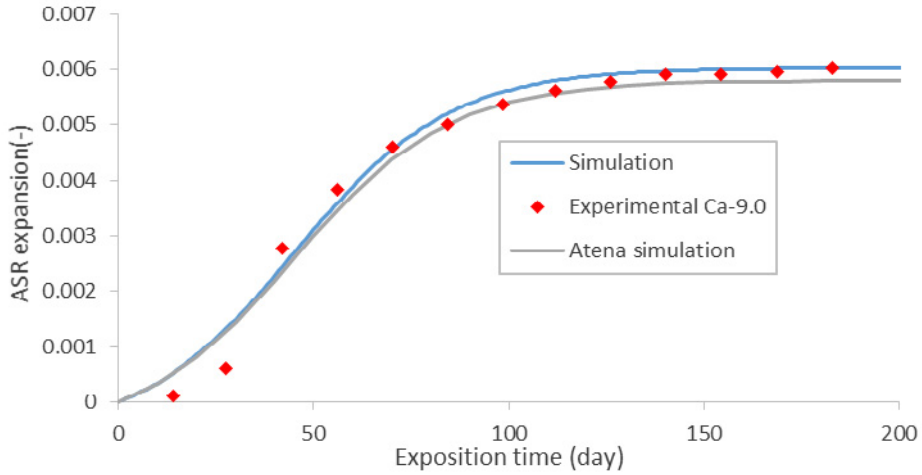


Fig. 6-12. Validation of free expansion (Kagimoto, Yasuda, & Kawamura, 2014)

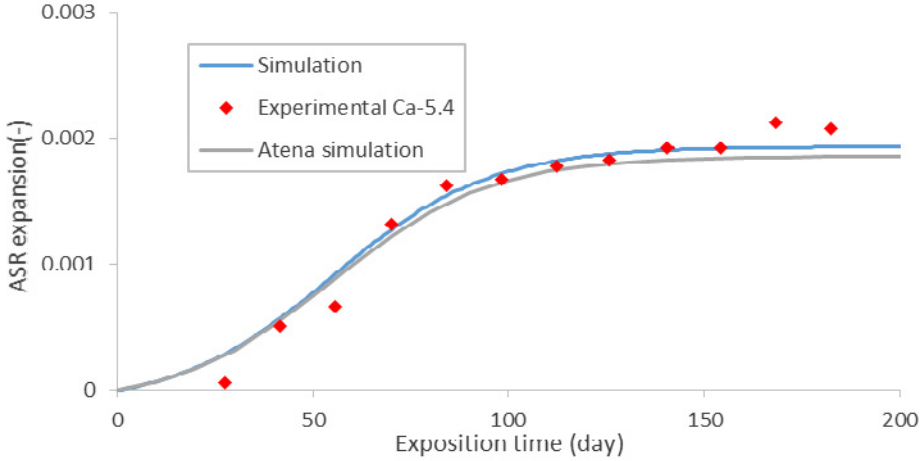


Fig. 6-13. Validation of free expansion, (Kagimoto, Yasuda, & Kawamura, 2014)

6.5.7 Implementation in Atena

Differential Equation (4.99) represents kinetics of development of ASR extent ξ. In the case of constant temperature θ in the structure, it can be solved analytically, see (4.104). Otherwise, it must be solved numerically. The following lines and equations describe the procedure to solve ξ that is implemented in ATENA.

Let's start from (4.99) and rewrite the equation into its differential form. We expect to now all at the time i and solve for time $i+1$. We do it in an iterative manner, i.e., we know all at iteration k and compute ξ at iteration $k+1$:

$$t_{c,i+1}^{k}\ \frac{\xi_{i+1}^{k+1}-\xi_i}{t_{i+1}-t_i}-\left(1-\xi_{i+1}^{k+1}\right)=0 \tag{4.121}$$

The unknown ξ_{i+1}^{k+1} ASR extent is searched for in the form $\xi_{i+1}^{k+1}=\xi_{i+1}^{k}+\delta\xi$, where $\delta\xi$ is the correction of ξ resulting from the k -th iteration. Denoting $t_{i+1}-t_i=\Delta t$ and $\xi_{i+1}^{k}-\xi_i=\Delta\xi$ the above equation can be written

$$t_{c,i+1}^{k}\ \left(\delta\xi+\Delta\xi\right)-\Delta t\left(1-\xi_{i+1}^{k}-\delta\xi\right)=0 \tag{4.122}$$

from which, after some mathematical manipulation, we can calculate $\delta\xi$

$$\delta\xi=-\frac{\Delta\xi\ t_{c,i+1}\ -\Delta t\left(1-\xi_{i+1}^{k}\right)}{t_{c,i+1}^{k}+\Delta t} \tag{4.123}$$

and $\xi_{i+1}^{k+1}=\xi_{i+1}^{k}+\delta\xi$. Note that in (4.121) thru (4.123) we used $t_{c,i+1}^{k}$ although $t_{c,i+1}^{k+1}$ should be employed, as $t_c=t_c(\xi,\theta)$ is a nonlinear function. Therefore, after each iteration, $k+1$ we update $t_{c,i+1}^{k}$ to $t_{c,i+1}^{k+1}$ and recalculate (4.122)

$$t_{c,i+1}^{k+1}\ \left(\delta\xi+\Delta\xi\right)-\Delta t\left(1-\xi_{i+1}^{k}-\delta\xi\right)=E^{k+1} \tag{4.124}$$

It yields an error E^{k+1} that is further compared against some maximum acceptable error. If it is too high, the next iteration is carried out; otherwise, the iteration process is finshed.

Note, however, that for the sake of convergency speed, the third and further iterations are in ATENA computed in a different way. Using linear interpolation between iteration k and $k+1$ requiring error $E^{k+2}=0$ in iteration $k+2$ value ξ_{i+1}^{k+2} is calculated by

$$\begin{aligned}
&E^{k+2}=\frac{1}{\xi_{i+1}^{k+1}-\xi_{i+1}^{k}}\left[E^{k+1}\left(\xi_{i+1}^{k+2}-\xi_{i+1}^{k}\right)+E^{k}\left(\xi_{i+1}^{k+2}-\xi_{i+1}^{k+1}\right)\right]\to 0\\
&E^{k+1}\left(\xi_{i+1}^{k+2}-\xi_{i+1}^{k}\right)+E^{k}\left(\xi_{i+1}^{k+2}-\xi_{i+1}^{k+1}\right)=0\\
&\xi_{i+1}^{k+2}=\frac{\xi_{i+1}^{k}E^{k+1}+\xi_{i+1}^{k+1}E^{k}}{E^{k+1}+E^{k}}
\end{aligned} \tag{4.125}$$

and checked by (4.124) written for iteration $k+2$. The iterating process continues this (latter) way until a sufficient accuracy is obtained.

The time step Δt is input by the user, but it is automatically limited by $\Delta t<0.01t_c$ requirement to ensure reasonable accuracy and convergence of the solution.

ASR loading results in the development of ASR strain and deterioration of material properties like Young modulus E, tension strength f_t, and fracture energy G_f. For each step i, we can write

$$\sigma_i=\sigma_{i-1}+E_{i-1}(\varepsilon_i-\varepsilon_{i-1})+\varepsilon_{i-1}(E_i-E_{i-1}) \tag{4.126}$$

The above equation calculates stress σ_i at (current) time step i based on stress σ_{i-1} from the previous time step and current changes of Young modulus E and strains ε. The strains ε represents "mechanical strains," i.e., strains producing stresses in an unrestrained material. They are total geometrical strains minus initial strain that corresponds to ASR expansion strains $\varepsilon_{ASR,i}$.

The differential formulation corresponds to the incremental solution used in Atena and the case of linear elastic material law. (More advanced materials are treated in a similar way). Using $E_k = E_0 \; cf_E^{ASR}(\xi_k)$, (4.126) can be written

$$\begin{aligned}
\sigma_i &\doteq \sigma_{i-1} + E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})\Delta\varepsilon_i + \varepsilon_{i-1} E_0 \;\Delta\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1},\xi_i) \\
\sigma_i &\doteq \sigma_{i-1} + E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})(\varepsilon_i - \varepsilon_{i-1}) + \varepsilon_{i-1} E_0 (\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_i) - \mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})) \\
\sigma_i &\doteq \sigma_{i-1} + E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})(\varepsilon_i - \varepsilon_{i-1}) + \varepsilon_{i-1} E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})\left[\frac{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_i)}{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})} - 1\right] \\
\sigma_i &\doteq \sigma_{i-1} + E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})(\varepsilon_i - \varepsilon_{i-1}) + \sigma_{i-1}\left[\frac{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_i)}{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})} - 1\right]
\end{aligned} \tag{4.127}$$

Note that strains ε are strains that are facilitated in material law, i.e., geometrical strains after subtracting ASR swelling strains. The ASR strains are implemented by initial element strains, and the term $\sigma_{i-1}\left(\frac{\xi_i}{\xi_{i-1}} - 1\right)$ is incorporated in the solution in the form of element initial stresses. Also, at each step, we update f_t and G_f.

An alternative solution to (4.127) is

$$\begin{aligned}
\sigma_i &= \sigma_{i-1} + E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})(\varepsilon_i - \varepsilon_{i-1}) + \varepsilon_{i-1} E_0 (\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_i) - \mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})) \\
\sigma_i &= \sigma_{i-1} + E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})(\varepsilon_i - \varepsilon_{i-1}) + \varepsilon_{i-1} E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})\left[\frac{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_i)}{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})} - 1\right] \\
\sigma_i &= \sigma_{i-1} + E_0 \,\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})\left[(\varepsilon_i - \varepsilon_{i-1} - \varepsilon_{i-1}\left(1 - \frac{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_i)}{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})}\right)\right]
\end{aligned} \tag{4.128}$$

The term $\varepsilon_{i-1}\left(1 - \frac{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_i)}{\mathrm{cf}_{\mathrm{E}}^{\mathrm{ASR}}(\xi_{i-1})}\right)$ is then added to ASR swelling initial element strains (4.120) calculated earlier. For linear material law, both equations (4.127) and (4.128) are equivalent. For the case of nonlinear law, they can slightly differ. By default, Atena prefers approach according to (4.128).

For the sake of simplicity, the above derivation has been presented for uniaxial stress-strain conditions. Its extension to 3D conditions is obvious.

6.5.8 Comments

The proposed model is derived from free expansion tests. The model works in 2D and 3D stress state by limiting expansion when a compressive load is present in a principal direction. In the

case of hydrostatic compression above -10 MPa, no ASR expansion occurs, and no reduction of mechanical properties happens (E, ft, Gf). This is justified by the fact that ASR gel grows into cracks and no macroscopic cracks occur.

The majority of structures are exposed to the thermal field; hence ASR usually proceeds faster close to the surface due to higher average temperature. Since the surface is often unloaded, the main expansion happens perpendicular to the surface, which induces a small compressive load and delamination of layers.

6.6 References

Atkins, P. W. (1994). Physical chemistry, 5th Ed., Oxford University Press, Oxford, U.K.

Berra, M. et al., Influence of stress restraint on the expansive behavior of concrete affected by alkali-silica reaction, Cement and concrete research 40, 1403-1409, 2010

Bazant, Z.P., Steffens, A. Mathematical model for kinetics of alkali–silica reaction in concrete. Cem Concr Res 2000;30:419–28.M. S. Darmawan & M. G. Stewart, Effect of Pitting Corrosion on Capacity of Prestressing Wires, Magazine of Concrete Research, 59(2), 131-139, 2007.

Collepardi, M., A. Marcialis, et al. (1972). "Penetration of Chloride Ions into Cement Pastes and Concrete." Journal of the American Ceramic Society 55: 534-535.

Comby-Perot, I., Bernard F., Bouchard P.O., Bay, F., Garcia-Diaz, E. Development and validation of a 3D computational tool to describe concrete behaviour at mesoscale. Application to the alkali–silica reaction. Comput Mater Science 2009;46(4):1163–77.

Červenka, J, Jendele, L., Šmilauer, V: Report I-05-01-2016 TAČR - TA04031458, 2016

Comi, C., Fedele, R., Perego, U. A chemo–thermo-damage model for the analysis of concrete dams affected by alkali–silica reaction. Mech Mater 2009;41:210–30.

III, T. E. U. B. E. (2000). DuraCrete Final Technical Report. General Guidelines for Durability Design and Redesign. Doc. BE95-1347/R17, 2000

Esposito, R., Hendriks, M.A.N., Degradation of the mechanical properties in ASR-affected concrete: overview and modeling, Strategies for Sustainable Concrete Structures, 2012

Farage, M. C. R. Modelagem e implementaca˜o nume´rica da expansa˜o por reaca˜o a´lcali–agregado do concreto, DSc thesis, Department of Civil Engineering, COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2000 (in Portuguese).

J. A. Gonzales, C. Andrade, C. Alonso & S. Feliu, Comparison of Rates of General Corrosion and Maximum Pitting Penetration on Concrete Embedded Steel Reinforcement, Cement and Concrete Research, 25(2), 257-264, 1995.

Kwon, S.-J. and U.-J. Na (2011). "Prediction of Durability for RC Columns with Crack and Joint under Carbonation Based on Probabilistic Approach." Int. Journal of Concrete Structures and Materials **5**(1): 11-18.

Kwon, S. J., U. J. Na, et al. (2009). "Service Life Prediction of Concrete Wharves with Early-aged Crack: Probabilistic Approach for Chloride Diffusion." Structural Safety **31**(1): 75-83.

Larive, C. (1998). ‘‘Apports combinés de l’expérimentation et de la modélisation á la comprehension de l’alcali-reaction et de ses effets mécaniques.’’ Monograph LPC, OA 28, Laboratoires des Ponts et Chaussées, Paris (partially translated into English).

Leger, P., Coté, P., Tinawi, R., Finite element analysis of concrete swelling due to alkali-aggregate reactions in dams, Computer and structures, 1995

Léger, P., Coté, P. and Tinawi, R. (1996) Finite Element Analysis of Concrete Swelling due to Alkali-aggregate Reactions in Dams, Comparers & Structures Vol. 60. No. 4. pp. 601-611.

Lindgart, J., (2012), Alkali–silica reactions (ASR): Literature review on parameters influencing laboratory performance testing. Cement and Concrete Research, 42, 223-243.

Liu Y, Weyers R.E., Modeling the Dynamic Corrosion Process in Chloride Contaminated Concrete Structures, Cement and Concrete Research, 28(3), 365-367, 1998.

Liu Y., Modelling the Time-to-corrosion Cracking of the Cover Concrete in Chloride Contaminated Reinforced Concrete Structures, Ph.D. dissertation, Virginia Polytechnic Institute, 1996.

Luping, Tang, et al. (2007). Chloride Ingress and Reinforcement Corrosion in Concrete under De-Icing Highway Environment – A Study after 10 Years' Field Exposure, SP Sveriges Tekniska Forskningsinstitut. **Vol. SP Report 2007:76**.

Multon, S., Cyr, M., Sellier, A., Leklou, N., Petit L. (2008) Coupled effects of aggregate size and alkali content on ASR expansion, Cement and Concrete Research 38, 350-359.

Multon, S., Toutlemonde, F. (2010) Effect of moisture conditions and transfers on alkali silica reaction damaged structures, Cement and Concrete Research 40, 924–934.

Multon, S., Sellier, A., Cyr, M. Chemo–mechanical modeling for prediction of alkali silica reaction (ASR) expansion. Cement Concrete Research 2009;39:490–500.

Lindgart, J., (2012), Alkali–silica reactions (ASR): Literature review on parameters influencing laboratory performance testing. Cement and Concrete Research, 42, 223-243.

Papadakis, V. G. (2000). "Effect of Supplementary Cementing Materials on Concrete Resistance Against Carbonation and Chloride Ingress." Cement Concrete Research **30**(2): 291-299.

Papadakis, V. G. and S. Tsimas (2002). "Supplementary Cementing Materials in Concrete. Part I: Efficiency and Design." Cement and Concrete Research **32**(10): 1525–1532.

Page CL, Nature and properties of concrete in relation to environment corrosion, Corrosion of Steel in Concrete, Aachen, 1992.

S. Poyet, Etude de la dégradation des ouvrages en béton atteints par la réaction alcali-silice: Approche expérimentale et modélisation numérique multi-échelles des dégradations dans un environnement hydro-chemomécanique variable, Ph.D. Thesis (in French), Université de Marne-La-Vallée, 2003

J. Rodriguez, L. M. Ortega, J. Casal & J. M. Diez, Corrosion of Reinforcement and Service Life of Concrete Structures. In Proc. of Int. Conf. on Durability of Building Materials and Components, 7, Stockholm, 1:117-126, 1996.

Saouma, V., Perotti, L., 2006, Constitutive model for alkali-aggregate reaction. ACI Material Journal 103, 194-202.

Spec-net, Corrosivity zones for steel construction [online] available from: http://www.spec-net.com.au/press/1014/gaa_081014/Corrosivity-Zones-for-Steel-Construction-Galvanizers-Association, 2015.

Ulm, F. J., Coussy, O., Li, K., Larive, C. Thermo–chemo–mechanics of ASR expansion in concrete structures. J Eng Mech 1999;126(3):233–42. Saouma, V., Perotti, L. Constitutive model for alkali-aggregate reactions. ACI Mater J 2006;103(3):194–202.

7 TRANSPORT ANALYSIS

As pointed out in the previous section, creep material behavior of concrete strongly depends on moisture and temperature conditions. Some constitutive models for creep in ATENA can pay regards to these factors and based on previously computed moisture and temperature histories within the structure they can predict concrete behavior more accurately. This section describes a module called CCStructuresTransport that is used to calculate the histories. A more accurate creep analysis then typically consists of two steps: firstly execute CCStructuresTransport module and calculate the moisture and humidity histories of the structure and secondly execute CCStructuresCreep module to carry out the actual static analysis. Of course, for both analyses, we have to prepare an appropriate model. Export/Import of the results between the modules is already done by ATENA automatically.

To be exact, both the transport and static analysis should be executed simultaneously, but as moisture and temperature transport does not depend significantly on structural deformations, i.e., coupling of the analyses is low, the implemented "staggered" solution yields sufficiently accurate results.

The governing equations for moisture transport read (for representative volume REV] :

$$\frac{\partial w}{\partial t} = \frac{\partial (w_e + w_n)}{\partial t} = -div(J_w) \tag{5.1}$$

where:

w is total water content defined as a ratio of weight of water at current time t to weight of water and cement at time $t_0 = 0$ in REV, [mass/mass], e.g. [kg/kg]

w_e, w_n = stands for the amounts of free and fixed (i.e. bound) water contents, [mass/mass],

J_w = moisture flux, [length*mass/ (time*mass)]. e.g. [m/day],

t =time, [time], e.g., [day].

The moisture flux is computed by

$$J_w = -D_w \nabla w_e \tag{5.2}$$

where

D_w is moisture diffusivity tensor of concrete [m^2/day],

∇ is gradient operator.

Note that in (5.2) only diffusion of water vapor is considered. Moisture advection is negligible.

The equations (5.1) and (5.2) can also be written as being dependent on w or relative moisture h. A relationship between h and w is given by

$$w = w(h) \tag{5.3}$$

Using (5.3) Equation (5.2) can be written as follows

$$J_w = -D_h \nabla h \tag{5.4}$$

A special attention must be paid to the calculation of the above time derivatives and integration of the governing equations. For example, in the case of usual Gauss integration and use of exact time derivatives the solution may suffer from mass losses. To remedy the problem the CCStructuresTransport module integrates the structure, i.e., all the individual finite elements in nodes and time derivatives are calculated numerically (Jendele 2001). This integration is similar to use of finite volume method, which is also known to be robust against the mass losses.

Heat transfer is governed by similar equation

$$\frac{\partial Q}{\partial t} = \frac{\partial}{\partial t}\left(C_T\,(T - T_{ref})\right) = C_T\,\frac{\partial T}{\partial t} = -div(J_T) \tag{5.5}$$

where

Q is total amount of energy in a unit volume [J/m^3]

C_T is heat capacity [J/(K.m^3)],

J_T is heat flux [J/(day.m^2)].

If hydration we want to add heat $Q_h(t)$, which expr$\frac{\partial}{\partial t}\frac{\partial}{\partial t}\frac{\partial}{\partial t}\frac{\partial}{\partial t}$esses amount of hydration heat within unit volume i.e $Q_h, \left[\frac{J}{m^3}\right]$, Equation (5.5) changes to

$$\frac{\partial}{\partial t}\left(C_T\,(T - T_{ref}) + Q_h\right) = C_T\,\frac{\partial T}{\partial t} + \frac{\partial Q_h}{\partial t} = -div(J_T) \tag{5.6}$$

Heat flux $J_T, \left[\frac{J}{m^2\,s}\right]$ is calculated by

$$J_T = -K_T\,grad(T) \tag{5.7}$$

and K_T stands for heat conductivity, e.g. [J/(day.m.K)].

Note that Equation (5.5) accounts for heat transport due to conduction only. Heat advection is negligible. In (5.5) we can also neglect hydration heat because in large times, its impact for heat transfer is small. On the other hand, we cannot neglect concrete moisture consumption due to the hydration process. According to (Bazant and Thonguthai 1978; Bazant 1986) hydration water content w_h can be calculated by:

$$w_n = w_h \approx 0.21\,c\left(\frac{t_e}{\tau_e + t_e}\right)^{\frac{1}{3}} \tag{5.8}$$

where

τ_e = 23 days, t_e is equivalent hydration time in water at temperature 25 ^0C that corresponds to the same degree of hydration subject to real age, moisture and temperature conditions of the material. The parameter c relates to the amount of cement and is calculated by(5.53). If temperature ranges from 0 to 100 ^0C, t_e is computed by

$$t_e = \int \beta_h \, \beta_T \, dt \tag{5.9}$$

where dt is time increment after the mold has been removed and coefficients β_T, β_h are calculated by

$$\beta_h = \frac{1}{1+(3.5-3.5h)^4} \tag{5.10}$$

$$\beta_T = \exp\left[\frac{U_h}{R}\left(\frac{1}{\widehat{T}_0} - \frac{1}{\widehat{T}}\right)\right] \tag{5.11}$$

In the fraction $\frac{U_h}{R}$ the symbol U_h stands for the activation energy of hydration and R is gas constant. According to (Bazant 1986) $\frac{U_h}{R} = 2700\,^0K$. $\widehat{T}, \widehat{T}_0$ are real and reference concrete temperature is expressed in ^0K. The reference temperature is given by

$$\widehat{T}_0 = 273.15 + 25 \tag{5.12}$$

The following figure depicts the relationship between real t and equivalent time t_e for the case of constant temperature $T = 15\,^0C$ and moisture $h = 0.8$. In practice, this relationship is rarely linear because with increase of time the amount of fixed water (due to hydration) w_h is increasing as well and it involves a gradual decrease of relative moisture h.

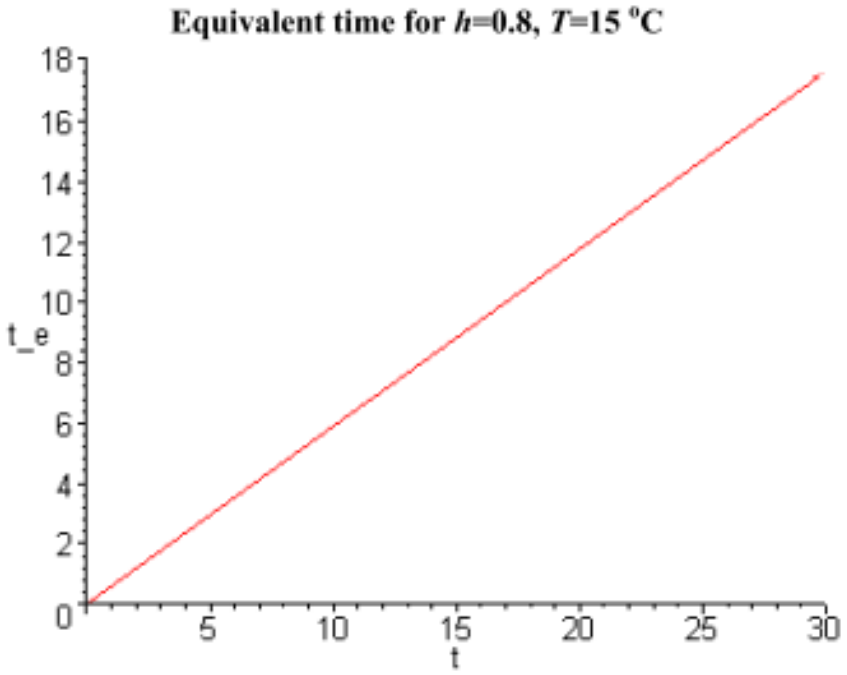


Fig. 7-1 Equivalent vs. real time relationship

The amount of water that was needed for hydration of concrete according to Equation (5.8) for the case of $c = 300$ kg is shown below:

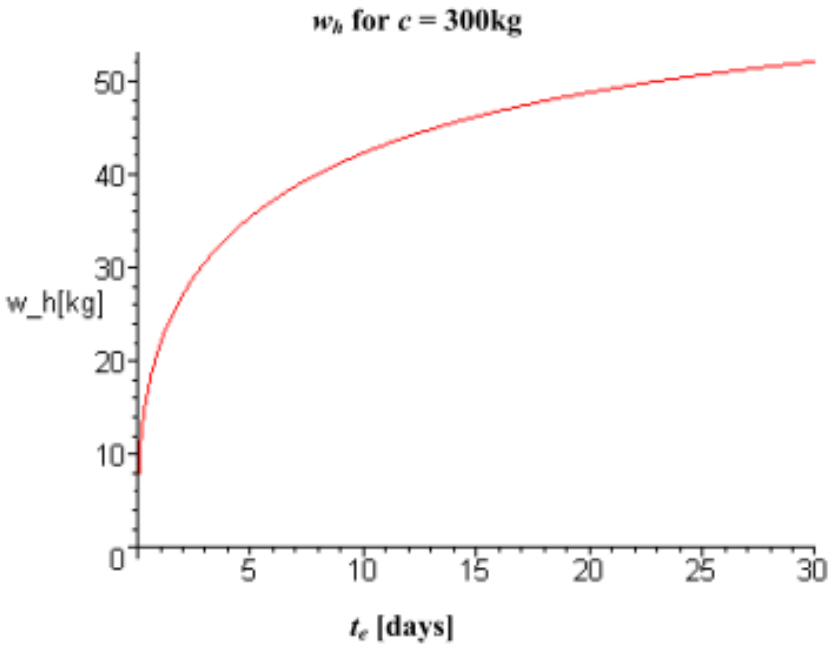


Fig. 7-2 Moisture consumed by hydration as a function of equivalent time

7.1 Numerical Solution of the Transport Problem – Spatial Discretisation

The transport governing equations for a typical engineering problem are too complex for analytical solution. Hence, similar to other ATENA engineering modules, the finite element method is also used for the CCStructuresTransport module. The transport problem gets spatially and temporarily discretized and then the resulting set of nonlinear algebraic equations is solved by a special iterative solver. This section is dedicated to the detailed description of the former type of discretization.

The solution is based on Equations (5.1) thru(5.7). Note that the unknown variables are

$$h = h(t);\ T = T(t); w = w(h,T);\ w_h = w_h(h,T,t) \tag{5.13}$$

and they are to be discretized. Let the left-hand side part of (5.1) and (5.4) is denoted LHS_h, LHS_T, respectively. The subscript h and T indicates moisture and temperature fluxes. Similar subscripts are also used for the right-hand-side of the equations, RHS_h, RHS_T. Notice that *RHS* expressions do not include the divergence operator!

$$LHS_h = \frac{\partial}{\partial t}\left(w + w_h\right) \tag{5.14}$$

$$LHS_T = C_T \frac{\partial T}{\partial t} + \frac{\partial Q_h}{\partial t} \tag{5.15}$$

$$\overline{RHS_h} = -\overline{J_w} = -\overline{J_h} \tag{5.16}$$

$$\overline{RHS_T} = -\overline{J_T} \tag{5.17}$$

The strip over an entity in the above equations means that the entity is vector. (Scalar entities do not have the strip). The fluxes $\overline{J_w} = \overline{J_h}$ are identical, i.e., the subscript *w* indicates also moisture phase. Using the above notation Equations (5.1) and (5.5) can be written as follows

$$\begin{aligned} LHS_h &= div(RHS_h) \\ LHS_T &= div(RHS_T) \end{aligned} \tag{5.18}$$

The LHS_h includes time derivative of moisture. It is computed using the following expressions:

$$\begin{aligned} w_h &= w_h(t_e) \\ \frac{\partial t_e}{\partial t} &= \beta_h \ \beta_T \\ \frac{\partial w_h}{\partial t} &= \frac{\partial w_h}{\partial t_e}\frac{\partial t_e}{\partial t} = \frac{\partial w_h}{\partial t_e}\beta_h \ \beta_T \end{aligned} \tag{5.19}$$

For the next derivation, let us write Equations(5.14), (5.15) in a general form:

$$\begin{aligned} LHS_h &= c_{hh}\frac{\partial h}{\partial t} + c_{hw}\frac{\partial w}{\partial t} + c_{hT}\frac{\partial T}{\partial t} + c_{h0} \\ LHS_T &= c_{Th}\frac{\partial h}{\partial t} + c_{Tw}\frac{\partial w}{\partial t} + c_{TT}\frac{\partial T}{\partial t} + c_{T0} \end{aligned} \tag{5.20}$$

and equations(5.16), (5.17)

$$\begin{aligned} \overline{RHS_h} &= \left[k_{hh}\right]\overline{\nabla}h + \left[k_{hw}\right]\overline{\nabla}w + \left[k_{hT}\right]\overline{\nabla}T + \overline{k}_{h0} \\ \overline{RHS_T} &= \left[k_{Th}\right]\overline{\nabla}h + \left[k_{Tw}\right]\overline{\nabla}w + \left[k_{TT}\right]\overline{\nabla}T + \overline{k}_{T0} \end{aligned} \tag{5.21}$$

where square bracket indicates that the enclosed entity is a matrix [ ].

Comparing (5.20) with (5.1) and (5.5) we find that

$$\begin{aligned} c_{hh} = c_{hT} = 0;\ c_{hw} = 1;\ c_{h0} \neq 0 \\ c_{Th} = c_{Tw} = 0;\ c_{TT} \neq 0;\ c_{T0} = \frac{\partial Q_h}{\partial t} \neq 0 \end{aligned} \tag{5.22}$$

The parameter c_{TT} is in ATENA an input material parameter, c_{h0} is computed from(5.19), i.e. $c_{h0} = \frac{\partial w_h}{\partial t_e} \beta_h \ \beta_T$. The solution also includes expressions $\frac{\partial w}{\partial h} \neq 0; \frac{\partial w}{\partial T}$. Their values depend on a constitutive model being used in the solution. For more information, please refer to Section Material Constitutive Model.

For right-hand sides, we can write in a similar manner:

$$\begin{gathered} \left[k_{hw}\right] = \left[k_{hT}\right] = \left[0\right]; \ \left[k_{hh}\right] \neq 0; \ \bar{k}_{h0} = \bar{0} \\ \left[k_{Th}\right] = \left[k_{Tw}\right] = \left[0\right]; \ \left[k_{TT}\right] \neq 0; \ \bar{k}_{T0} = \bar{0} \end{gathered} \tag{5.23}$$

The parameter $\left[k_{TT}\right]$ is a material input parameter, $\left[k_{hh}\right]$ is calculated from a constitutive model, see the next section.

For the next derivation, let us assume discretization of the unknown variables as follows. Remind that these are in the governing equations integrated in finite nodes, (Celia, Bouloutas et al. 1990; Celia and Binning 1992).

$$\begin{gathered} h = \bar{N}^T \ \bar{h}; \qquad \bar{\nabla} h = \left[\bar{\nabla}\bar{N}\right]^T \ \bar{h} \\ w = \bar{N}^T \ \bar{w}; \qquad \bar{\nabla} w = \left[\bar{\nabla}\bar{N}\right]^T \ \bar{w} \\ T = \bar{N}^T \ \bar{T}; \qquad \bar{\nabla} T = \left[\bar{\nabla}\bar{N}\right]^T \ \bar{T} \end{gathered} \tag{5.24}$$

where

$\bar{h}$, $\bar{w}$, $\bar{T}$ stands for vectors of the corresponding entities. The vectors have dimension *n* equal to number of finite nodes of the problem.

$\bar{N}$ is vector of interpolation, (i.e., shape) functions,

$$\left[\bar{\nabla}\bar{N}\right]^T = \begin{bmatrix} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & \cdots & \frac{\partial N_n}{\partial x} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_2}{\partial y} & \cdots & \frac{\partial N_n}{\partial y} \\ \frac{\partial N_1}{\partial z} & \frac{\partial N_2}{\partial z} & \cdots & \frac{\partial N_n}{\partial z} \end{bmatrix}$$

Using (5.24) Equations (5.20) and (5.21) can be written in the form

$$LHS_h = c_{hh}\bar{N}^T \frac{\partial \bar{h}}{\partial t} + c_{hw}\bar{N}^T \frac{\partial \bar{w}}{\partial t} + c_{hT}\bar{N}^T \frac{\partial \bar{T}}{\partial t} + c_{h0}$$

$$LHS_T = c_{Th}\bar{N}^T \frac{\partial \bar{h}}{\partial t} + c_{Tw}\bar{N}^T \frac{\partial \bar{w}}{\partial t} + c_{TT}\bar{N}^T \frac{\partial \bar{T}}{\partial t} + c_{T0} \tag{5.25}$$

and

$$\overline{RHS_h} = [k_{hh}]\left[\bar{\nabla}\bar{N}\right]^T \bar{h} + [k_{hw}]\left[\bar{\nabla}\bar{N}\right]^T \bar{w} + [k_{hT}]\left[\bar{\nabla}\bar{N}\right]^T \bar{T} + \bar{k}_{h0}$$

$$\overline{RHS_T} = [k_{Th}]\left[\bar{\nabla}\bar{N}\right]^T \bar{h} + [k_{Tw}]\left[\bar{\nabla}\bar{N}\right]^T \bar{w} + [k_{TT}]\left[\bar{\nabla}\bar{N}\right]^T \bar{T} + \bar{k}_{T0} \tag{5.26}$$

The resulting set of equations are solved iteratively using finite element method, see (Zienkiewicz and Taylor 1989), (weak formulation, in which the shape functions $\bar{N}$ are used as weight function):

$$\int_V \bar{N}\left(LHS_h - div(\overline{RHS_h})\right)dV = 0$$

$$\int_V \bar{N}\left(LHS_T - div(\overline{RHS_T})\right)dV = 0 \tag{5.27}$$

where V is volume of the analyzed structure. Each of the above equations represents a set of equations with dimension equal to number of finite nodes *n*. Note that $div(\overline{RHS_h})$ and $div(\overline{RHS_T})$ are scalars !

In the next derivation, the two parts of (5.27) are dealt with separately.

$$\int_V \bar{N}\left(LHS_h\right)dV = \int_V \bar{N}\left(c_{hh}\bar{N}^T \frac{\partial \bar{h}}{\partial t} + c_{hw}\bar{N}^T \frac{\partial \bar{w}}{\partial t} + c_{hT}\bar{N}^T \frac{\partial \bar{T}}{\partial t} + c_{h0}\right)dV =$$

$$\int_V c_{hh}\bar{N}\bar{N}^T dV \frac{\partial \bar{h}}{\partial t} + \int_V c_{hw}\bar{N}\bar{N}^T dV \frac{\partial \bar{w}}{\partial t} + \ldots \int_V c_{h0}\bar{N}dV = \tag{5.28}$$

$$[cc_{hh}]\frac{\partial \bar{h}}{\partial t} + [cc_{hw}]\frac{\partial \bar{w}}{\partial t} + \ldots \overline{cc}_{h0}$$

$$\int_V \bar{N}\left(LHS_T\right)dV = \int_V \bar{N}\left(c_{Th}\bar{N}^T \frac{\partial \bar{h}}{\partial t} + c_{Tw}\bar{N}^T \frac{\partial \bar{w}}{\partial t} + c_{TT}\bar{N}^T \frac{\partial \bar{T}}{\partial t} + c_{T0}\right)dV =$$

$$[cc_{Th}]\frac{\partial \bar{h}}{\partial t} + [cc_{Tw}]\frac{\partial \bar{w}}{\partial t} + \ldots \overline{cc}_{T0} \tag{5.29}$$

and the matrices $[cc]$ are calculated by

$$\left[cc_{hh}\right]=\int_V c_{hh}\bar{\bar{N}}\bar{\bar{N}}^T dV;\ \left[cc_{hw}\right]=\int_V c_{hw}\bar{\bar{N}}\bar{\bar{N}}^T dV;\ \ldots\ \overline{cc}_{h0}=\int_V c_{h0}\bar{\bar{N}}dV$$
$$\left[cc_{Th}\right]=\int_V c_{Th}\bar{\bar{N}}\bar{\bar{N}}^T dV;\ \left[cc_{Tw}\right]=\int_V c_{Tw}\bar{\bar{N}}\bar{\bar{N}}^T dV;\ \ldots\ \overline{cc}_{T0}=\int_V c_{T0}\bar{\bar{N}}dV \tag{5.30}$$

The second part of (5.27) are calculated using Green theorem (5.36):

$$\int_V \bar{\bar{N}}\left(-div(\overline{RHS_h})\right)dV=-\oint_S \bar{\bar{N}}\left(\bar{n}_s^T\ \overline{RHS_h}\right)dS+\int_V\left[\bar{\bar{\nabla}}N\right]\overline{RHS_h}\ dV=$$

$$\begin{aligned}&=-\oint_S \bar{\bar{N}}\ \bar{n}_s^T\left(\left[k_{hh}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T\bar{h}+\left[k_{hw}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T\bar{w}+\left[k_{hT}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T\bar{T}+\bar{k}_{h0}\right)dS+\\&+\int_V\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]\left(\left[k_{hh}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T\bar{h}+\left[k_{hw}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T\bar{w}+\left[k_{hT}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T\bar{T}+\bar{k}_{h0}\right)dV\end{aligned} \tag{5.31}$$

where S is the structural surface (with possibly defined boundary conditions).

In the case of heat transfer, we can derive all the equations in a similar way. In analogy to (5.30) let us introduce matrices $\left[kk\right]$

$$\begin{aligned}&\left[kk_{hh}\right]=\int_V\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]\left[k_{hh}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T dV\\&\left[kk_{hw}\right]=\int_V\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]\left[k_{hw}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T dV\\&\overline{kk}_{h0}=\int_V\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]\bar{k}_{h0}\,dV\\&\ldots\\&\left[kk_{TT}\right]=\int_V\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]\left[k_{TT}\right]\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]^T dV\\&\overline{kk}_{T0}=\int_V\left[\bar{\bar{\nabla}}\bar{\bar{N}}\right]\bar{k}_{T0}\,dV\end{aligned} \tag{5.32}$$

and also

$$\left[J_{hh}\right]=\oint_S \bar{N}\ \bar{n}_s^T\left[k_{hh}\right]\left[\bar{\nabla}\bar{N}\right]^T dS$$

$$\left[J_{hw}\right]=\oint_S \bar{N}\ \bar{n}_s^T\left[k_{hw}\right]\left[\bar{\nabla}\bar{N}\right]^T dS$$

...

$$\left[J_{TT}\right]=\oint_S \bar{N}\ \bar{n}_s^T\left[k_{TT}\right]\left[\bar{\nabla}\bar{N}\right]^T dS$$

$$\bar{J}_{h0}=\oint_S \bar{N}\ \bar{n}_s^T\ \bar{k}_{h0}dS$$

$$\bar{J}_{T0}=\oint_S \bar{N}\ \bar{n}_s^T\ \bar{k}_{T0}dS \tag{5.33}$$

Using (5.28) to (5.33) the original governing equations (5.27) can be written as follows:

$$\begin{aligned}&\left[cc_{hh}\right]\frac{\partial\bar{h}}{\partial t}+\left[cc_{hw}\right]\frac{\partial\bar{w}}{\partial t}+\left[cc_{hT}\right]\frac{\partial\bar{T}}{\partial t}+\overline{cc}_{h0}+\left[kk_{hh}\right]\bar{h}+\left[kk_{hw}\right]\bar{w}+\left[kk_{hT}\right]\bar{T}+\overline{kk}_{h0}=\\&=\left[J_{hh}\right]\bar{h}+\left[J_{hw}\right]\bar{w}+\left[J_{hT}\right]\bar{T}+\bar{J}_{h0}\\&\left[cc_{Th}\right]\frac{\partial\bar{h}}{\partial t}+\left[cc_{Tw}\right]\frac{\partial\bar{w}}{\partial t}+\left[cc_{TT}\right]\frac{\partial\bar{T}}{\partial t}+\overline{cc}_{T0}+\left[kk_{Th}\right]\bar{h}+\left[kk_{Tw}\right]\bar{w}+\left[kk_{TT}\right]\bar{T}+\overline{kk}_{T0}=\\&=\left[J_{Th}\right]\bar{h}+\left[J_{Tw}\right]\bar{w}+\left[J_{TT}\right]\bar{T}+\bar{J}_{T0}\end{aligned} \tag{5.34}$$

After sorting the unknown variables $\bar{h}$, $\bar{T}$ by finite nodes into a single vector $\bar{\psi}$, Equation (5.34) will read

$$\left[cc\right]\frac{\partial\bar{\psi}}{\partial t}+\left|kk\right|\bar{\psi}+\overline{cc}_0+\overline{kk}_0=\left[J\right]\bar{\psi}+\bar{J}_0 \tag{5.35}$$

The right-hand side (5.35) is non-zero only for non-zero prescribed boundary conditions and hence it has character of "load" vector in a static analysis.

In (5.31) we used Green theorem. It states:

$$\begin{aligned}\int_V u\ div(\bar{v})dV&=\oint_S u\ n_s^T\ \bar{v}\ dS-\int_V\left[\bar{\nabla}u\right]\underline{v}\ dV\\\int_V \bar{u}\ div(\bar{v})dV&=\oint_S \bar{u}\ n_s^T\ \bar{v}\ dS-\int_V\left[\bar{\nabla}\bar{u}\right]\underline{v}\ dV\end{aligned} \tag{5.36}$$

where

$$\left[\bar{\nabla} u\right] = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \end{bmatrix}$$

$$\left[\bar{\nabla} \bar{u}\right] = \begin{bmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} & \frac{\partial u_1}{\partial z} \\ \frac{\partial u_2}{\partial x} & \frac{\partial u_2}{\partial y} & \frac{\partial u_2}{\partial z} \\ \cdots & \cdots & \cdots \\ \frac{\partial u_n}{\partial x} & \frac{\partial u_n}{\partial y} & \frac{\partial u_n}{\partial z} \end{bmatrix} \tag{5.37}$$

7.2 Numerical Solution of the Transport Problem – Temporal Discretisation

The heat and moisture transfer governing equations (5.35) can be written in the form:

$$^{t+\Delta t}\mathbf{K}\,^{t+\Delta t}\underline{\psi} + {}^{t+\Delta t}\mathbf{C}\frac{\partial}{\partial t}\left(^{t+\Delta t}\underline{\psi}\right) = {}^{t+\Delta t}\underline{J} \tag{5.38}$$

where $^{t+\Delta t}\mathbf{K}$, $^{t+\Delta t}\mathbf{C}=$ are unsymmetrical problem matrices defined in the previous section, $^{t+\Delta t}\underline{J}=$vector of concentrated nodal fluxes (both moisture and heat) and $^{t+\Delta t}\underline{\psi}$ is the vector of unknown variables. All of these apply for time $t+\Delta t$. Equation (5.38) is solved iteratively. i.e., the vector $^{t+\Delta t}\underline{\psi}$ is searched for in the incremental form:

$$^{t+\Delta t}\underline{\psi} = {}^{t+\Delta t(i)}\underline{\psi} = {}^{t+\Delta t(i-1)}\underline{\psi} + {}^{t+\Delta t(i)}\Delta\underline{\psi} \tag{5.39}$$

where index $^{(i)}$ indicates the number of iteration and $^{t+\Delta t(i)}\Delta\underline{\psi}$ is the increment of the unknowns for time $t+\Delta t$ and iteration $^{(i)}$:

$$^{t+\Delta t(i)}\Delta\underline{\psi} = {}^{t+\Delta t(i-1)}\,\tilde{\mathbf{K}}^{-1}\,{}^{t+\Delta t(i)}\tilde{\underline{J}} \tag{5.40}$$

The matrix and vector $^{t+\Delta t(i-1)}\tilde{\mathbf{K}}^{-1}$ and $^{t+\Delta t(i)}\tilde{\underline{J}}$ is derived from $^{t+\Delta t(i-1)}\mathbf{K}^{-1}$, $^{t+\Delta t(i-1)}\mathbf{C}^{-1}$ and Δt based on temporal integration method being used:

CCStructureTransport module currently supports θ Crank Nicholson (Wood. 1990) and Adams-Bashforth (Diersch and Perrochet 1998) integration scheme. The former scheme is linear, i.e., it's a first-order integration procedure. The latter scheme is a second-order integration procedure. It is supposed to be more accurate; however, it is also more CPU and RAM expensive and it is more difficult to predict its real behavior. Hence, the θ Crank Nicholson scheme is typically preferred. It has been more tested and verified in the CCStructureTransport module, and thereby it is more recommended.

7.2.1 θ-parameter Crank Nicholson Scheme

This scheme comprises a number of well established integration procedures. It depends, what value of the parameter θ is used. The set of equations (5.38) is solved for time $t+\Delta t\ \theta$, whereby the vector of unknown variables is calculated as a linear combination of the corresponding vectors at a time t and $t+\Delta t$. Hence

$$^{t+\Delta t}\underline{\psi} = {}^{t}\underline{\psi}(1-\theta) + {}^{t+\Delta t}\underline{\psi}\theta \tag{5.41}$$

Depending on a particular value of the parameter θ we get the well known Euler implicit integration (for θ=1), trapezoidal Crank Nicholson scheme (for θ=0.5), Galerkin integration method (for θ=2/3) or even Euler explicit scheme (for θ=0), which is only conditionally stable.

Solution predictor:

$$^{t+\Delta t}\psi = {}^{t}\psi + \Delta t\frac{\partial^{t}\psi}{\partial t} \tag{5.42}$$

Solution corrector:

$$\frac{\partial^{t+\Delta t}\psi}{\partial t} = \frac{1}{\Delta t}\left({}^{t+\Delta t}\psi - {}^{t}\psi\right) \tag{5.43}$$

Using the above, after some mathematical manipulation, we derive final expressions for $\tilde{\mathbf{K}}, \tilde{J}$. These read:

$$\tilde{\mathbf{K}} = \left(\mathbf{K}\theta + \frac{1}{\Delta t}\mathbf{C}\right)$$

$$\tilde{J} = \bar{J} - \mathbf{K}\left(\theta\,{}^{t+\Delta t}\psi + (1-\theta)\,{}^{t}\psi\right) - \mathbf{C}\frac{1}{\Delta t}\left({}^{t+\Delta t}\psi - {}^{t}\psi\right) \tag{5.44}$$

$$\Delta\psi = \left(\tilde{\mathbf{K}}\right)^{-1}\tilde{J}$$

7.2.2 Adams-Bashforth Integration Scheme

Solution predictor:

$$^{t+\Delta t}\psi = {}^{t}\psi + \frac{\Delta t}{2}\left[\left(2+\frac{\Delta t}{\Delta t_{prev}}\right)\frac{\partial^{t}\psi}{\partial t} - \frac{\Delta t}{\Delta t_{prev}}\frac{\partial^{t}\psi_{prev}}{\partial t}\right] \tag{5.45}$$

where

index $_{prev}$ indicates that the entity comes from time preceding time t Note that we assume that all entities from time t are already known and we solve for their values at time $t+\Delta t$.

Solution corrector:

$$\frac{\partial^{t+\Delta t}\psi}{\partial t} = \frac{2}{\Delta t}\left({}^{t+\Delta t}\psi - {}^{t}\psi\right) - \frac{\partial^{t}\psi}{\partial t} \tag{5.46}$$

$$\frac{\partial^t \psi}{\partial t} = \frac{\Delta t_{prev}}{\Delta t + \Delta t_{prev}} \left(\frac{^{t+\Delta t}\psi - ^t \psi}{\Delta t} \right) + \frac{\Delta t}{\Delta t + \Delta t_{prev}} \left(\frac{^t \psi - ^t \psi_{prev}}{\Delta t_{prev}} \right) \tag{5.47}$$

Similar to (5.44) we have here for $\tilde{\mathbf{K}}, \tilde{J}$:

$$\tilde{\mathbf{K}} = \Delta t_{n-1} \left(\mathbf{K} \Delta t_n \left(\Delta t_n + \Delta t_{n-1} \right) + \mathbf{C} \left(2\Delta t_n + \Delta t_{n-1} \right) \right)$$

$$\begin{aligned} \tilde{J} = & -\mathbf{K}^{\,t+\Delta t}\psi \left(\left(\Delta t_n \right)^2 \Delta t_{n-1} + \Delta t_n \left(\Delta t_{n-1} \right)^2 \right) + \\ & + \mathbf{C} \left(-^{t+\Delta t}\psi \left(2\Delta t_{n-1} \Delta t_n + \left(\Delta t_{n-1} \right)^2 \right) + ^t\psi \left(2\Delta t_{n-1} \Delta t_n + \left(\Delta t_{n-1} \right)^2 + \left(\Delta t_n \right)^2 \right) - ^{t-\Delta t}\psi \left(\Delta t_n \right)^2 \right) \\ & + J \left(\left(\Delta t_n \right)^2 \Delta t_{n-1} + \Delta t_n \left(\Delta t_{n-1} \right)^2 \right) \end{aligned} \tag{5.48}$$

$$\Delta \psi = \left(\tilde{\mathbf{K}} \right)^{-1} \tilde{J}$$

7.2.3 Reduction of Oscillations and Convergence Improvement

The transport governing equations are prone to suffer from oscillations. As reported in (Jendele 2001) this can be improved by introducing a sort of Line Search method damping η. The basic idea is that Equation (5.39) gets replaced by

$$^{t+\Delta t}\underline{\psi} = ^{t+\Delta t(i)}\underline{\psi} = ^{t+\Delta t(i-1)}\underline{\psi} + ^{t+\Delta t(i)}\eta \Delta \underline{\psi} \tag{5.49}$$

where η is a new damping factor. The factor is typically set to something in range $<0.3...1>$ depending on the current convergence behavior of the problem.

7.3 Material Constitutive Model

The previous section referred to a material constitutive model, i.e., it was assumed that we know how to compute material diffusivity matrix D_h, (see(5.4)), and material capacity $w = w(h)$, (see(5.1). Calculation of these entities is described here.

Currently, ATENA has only two constitutive models available for transport analysis. The first one, i.e., CCModelBaXi94 is characterized as follows and the second one, i.e., CCTransportMaterial is briefly described later in this section.

CCModelBaXi94

For heat transport, a simple constant linear model is implemented. For moisture transport, a nonlinear model based on the model (Xi, Bazant et al. 1993; Xi, Bazant et al. 1994) has been developed.

It can be used for temperatures in range $T =< 5 ...75\ ^0C >$ and moisture $H =< 0 ...1 >$. It is important to note that the model was originally written only for mortar; hence, it is inaccurate for concrete with an aggregate having higher permeability (i.e., diffusivity) and/or absorption. The model has the following main parameters

- Type of cement
- Water-cement ratio $wc = \frac{w}{c}$

As already pointed out, the model does not account for aggregate, i.e., it predicts moisture move only in pores filled by water-cement paste.

The main entity of the model is water content $w = w(h,t,T,\frac{w}{c})$. It is defined as follows:

$$w = \frac{G_w}{G_{w,0} + G_c} \tag{5.50}$$

where

G_w is the water content in mortar at time t, $\left[\frac{kg}{m^3\ of\ morter}\right]$,

$G_{w,0}$ is the water content at time zero, $\left[\frac{kg}{m^3\ of\ morter}\right]$,

G_c is the amount of cement at time zero, $\left[\frac{kg}{m^3\ of\ morter}\right]$.

Mortar here stands for a mixture of water and cement. If concrete material is to be considered, then w can be calculated by

$$w = \frac{G_w \frac{V_{concrete}}{V_{mortar}}}{G_{w,0}\frac{V_{concrete}}{V_{mortar}} + G_c \frac{V_{concrete}}{V_{mortar}}} \doteq \frac{\widetilde{G}_w}{\widetilde{G}_{w,0} + \widetilde{G}_c} \tag{5.51}$$

where $\frac{V_{concrete}}{V_{mortar}}$ is the ratio of total volume to (only) volume of mortar (i.e., water and cement) and $\widetilde{G}$ are corresponding amounts of water and cements in concrete, (i.e., not only in mortar!) $\left[\frac{kg}{m^3\ of\ concrete}\right]$.

The model itself already accounts for moisture used by the hydration process. i.e., $\frac{\partial w}{\partial t} \neq 0$. As a result, w_h according to (5.19) need not be implemented.

On the other hand, if moisture losses due to hydration are to be computed by the model based on (5.19), it is important to fix $\frac{\partial w}{\partial t} = 0$ and to modify w_h, so that it predicts "relative" moisture content w used throughout whole derivation CCStructuresTransport. The original function for w_h was written for absolute weight of water and hence, for "relative" content of water Equations (5.8) must be rewritten into

$$w_h = \frac{0.21\,\widetilde{G}_c \left(\frac{t_e}{\tau_e + t_e} \right)^{\frac{1}{3}}}{\widetilde{G}_{w,0} + \widetilde{G}_c} = \frac{\widetilde{G}_c}{\widetilde{G}_{w,0} + \widetilde{G}_c} 0.21 \left(\frac{t_e}{\tau_e + t_e} \right)^{\frac{1}{3}} = \frac{G_c}{G_{w,0} + G_c} 0.21 \left(\frac{t_e}{\tau_e + t_e} \right)^{\frac{1}{3}} \quad (5.52)$$

and the constant c from (5.8) becomes equal to

$$c = \frac{G_c}{G_{w,0} + G_c} = \frac{\widetilde{G}_c}{\widetilde{G}_{w,0} + \widetilde{G}_c} \quad (5.53)$$

More detailed description of the model is beyond the scope of this document and the reader is referred to in (Xi, Bazant et al. 1993; Xi, Bazant et al. 1994).

CCTransportMaterial

CCTransport material is a simple constitutive law that allows users to enter laboratory-measured moisture and heat characteristics. Referring to Equations (5.1) and (5.5) heat and moisture flow governing equations can be written in the following general form:

Heat :

$$\frac{\partial Q}{\partial t} = C_{Th} \frac{\partial h}{\partial t} + C_{TT} \frac{\partial T}{\partial t} + C_{Tw} \frac{\partial w}{\partial t} + C_{Tt} = -\frac{\partial}{\partial x} \left(K_{Th}\, grad(h) + K_{TT}\, grad(T) + K_{Tw}\, grad(w) + K_{Tgrav} \right)$$

Moisture :

$$\frac{\partial w}{\partial t} = C_{wh} \frac{\partial h}{\partial t} + C_{wT} \frac{\partial T}{\partial t} + C_{ww} \frac{\partial w}{\partial t} + C_{wt} = -\frac{\partial}{\partial x} \left(D_{wh}\, grad(h) + D_{wT}\, grad(T) + D_{ww}\, grad(w) + D_{wgrav} \right) \quad (5.54)$$

The parameters C_{Th}, C_{TT} … K_{wgrav} are calculated as:

$$
\begin{aligned}
C_{Th} &= C_{Th}^0 \, f_{C_{Th}}^h(h) \, f_{C_{Th}}^T(T) \; f_{C_{Th}}^t(t) \\
C_{TT} &= C_{TT}^0 \, f_{C_{TT}}^h(h) \, f_{C_{TT}}^T(T) \; f_{C_{TT}}^t(t) \\
C_{Tw} &= C_{Tw}^0 \, f_{C_{Tw}}^h(h) \, f_{C_{Tw}}^T(T) \, f_{C_{Tw}}^t(t) \\
C_{Tt} &= C_{Tt}^0 \, f_{C_{Tt}}^h(h) \, f_{C_{Tt}}^T(T) \; f_{C_{Tt}}^t(t) \\
C_{wh} &= C_{wh}^0 \, f_{C_{wh}}^h(h) \, f_{C_{wh}}^T(T) \; f_{C_{wh}}^t(t) \\
C_{wT} &= C_{wT}^0 \, f_{C_{wT}}^h(h) \, f_{C_{wT}}^T(T) \; f_{C_{wT}}^t(t) \\
C_{ww} &= C_{ww}^0 \, f_{C_{ww}}^h(h) \, f_{C_{ww}}^T(T) \; f_{C_{ww}}^t(t) \\
C_{wt} &= C_{wt}^0 \, f_{C_{wt}}^h(h) \, f_{C_{wt}}^T(T) \; f_{C_{wt}}^t(T)
\end{aligned}
$$

$$
\begin{aligned}
K_{Th} &= K_{Th}^0 \, f_{K_{Th}}^h(h) \, f_{K_{Th}}^T(T) f_{K_{Th}}^t(t) \\
K_{TT} &= K_{TT}^0 \, f_{K_{TT}}^h(h) \, f_{K_{TT}}^T(T) \, f_{K_{TT}}^t(t) \\
K_{Tw} &= K_{Tw}^0 \, f_{K_{Tw}}^h(h) \, f_{K_{Tw}}^T(T) \, f_{K_{Tw}}^t(t) \\
K_{Tgrav} &= K_{Tgrav}^0 \, f_{K_{Tgrav}}^h(h) \, f_{K_{Tgrav}}^T(T) \, f_{K_{Tgrav}}^t(t) \\
D_{wh} &= D_{wh}^0 \, f_{D_{wh}}^h(h) \, f_{D_{wh}}^T(T) \, f_{D_{wh}}^t(t) \\
D_{wT} &= D_{wT}^0 \, f_{D_{wT}}^h(h) \, f_{D_{wT}}^T(T) f_{D_{wT}}^t(t) \\
D_{ww} &= D_{ww}^0 \, f_{D_{ww}}^h(h) \, f_{D_{ww}}^T(T) f_{D_{ww}}^t(t) \\
D_{wgrav} &= D_{wgrav}^0 \, f_{D_{wgrav}}^h(h) \, f_{D_{wgrav}}^T(T) \, f_{D_{wgrav}}^t(t)
\end{aligned}
\qquad (5.55)
$$

and the constant parameters C_{Th}^0 thru D_{wgrav}^0 and functions $f_{C_{Th}}^h(h)$ thru $f_{D_{wgrav}}^T(T)$ are input parameters, (to be possibly obtained from some experiments). The functions are defined as multilinear functions and only their ids are input into CCTransportMaterial model definition.

Note that gravity terms in RHS of (5.54) have a little physical justification in heat and moisture diffusion gathered transports; nevertheless, they are included to allow using this material law for the solution of other kinds of transport problems.

CCTransportMaterialLevel7 material

CCTransport materialLevel7 is an extension of the above CCMaterialTransport material in the way it automatically computes moisture and temperature capacity and conductivity/diffusivity incl. "sink" terms regarding hydration (i.e., rate of hydration heat and moisture consumption during concrete hydration). In terms of the above nomenclature, this upper material level calculates $C_{TT}, K_{TT}, C_{Tt,} C_{wh}, D_{wh}, C_{wt}$. As already mentioned, the presented material adds on its bottom level, i.e., CCMaterialTransport. All parameters and characteristics from the bottom level, (i.e., those from CCMaterialTransport) can still be input and used. They typically serve for a refinement/addition of parameters generated by the upper material level. The result from the bottom and upper levels are simply added to form the final characteristics of the material model CCTransportMaterialLevel7. Note that default values of $C_{TT}, K_{TT}, C_{Tt,} C_{wh}, D_{wh}, C_{wt}$ in the bottom level are by default set to zero.

Hydration heat and affinity hydration model

The most important part of the presented model is the computation of the concrete hydration maturity factor. It is accompanied by the calculation of generated hydration heat and consumed hydration moisture. The analysis is based on the affinity hydration model, which provides a framework for accommodating all stages of cement hydration.

Consider hydrating cement under isothermal temperature 25°C a relative humidity $h=1$. At this temperature, the rate of hydration maturity factor α, $\alpha \in \langle 0...1 \rangle$ can be expressed by *chemical affinity* $\tilde{A}_{25} = \tilde{A}_{25}(\alpha)$:

$$\frac{\partial \alpha}{\partial t} = \tilde{A}_{25} \tag{5.56}$$

where $\tilde{A}$ stands for the chemical affinity, [s^{-1}], The expression already includes coefficient $\exp\left(-\frac{E_a}{RT}\right)$. Hence $\tilde{A}_{25}$ is not normalized and refers to temperature 25°C. For different temperatures it is replaced by $\tilde{A}$, see (5.60). R is gas constant $8314.41\frac{\mathrm{J}}{\mathrm{kmol\,K}}$, T is temperature, [K] and E_a is 40 kJ/mol. It is worthy to note the incorporation of the maturity method into (5.56). A characteristic time might be introduced to express an affinity $\hat{A}$ (Bernard, Ulm et al. 2003).

The affinity property can be obtained experimentally or analytically. Using experimental approach, heat flow $q(t)$ that corresponds to the hydration heat $Q_h = Q_h(t)$ is measured by isothermal calorimetry.

Alternatively, the hydration material parameters are computed by an analytical micro-scale model that accounts for the majority of underlying chemical reactions as well as the topology of cement grains (with the consequence to hydration kinetics). The solution stems from (Smilauer and Bittnar 2006), and it employs discrete hydration model CEMHYD3D (Bentz 2005), allowing to account for the particle size distribution of cement, the chemical composition of cement, temperature and moisture history in concrete, etc.

Having history of Q_h (for $T = 273.15 + 25, \eta = 1$), the approximation of α parameter is given by

$$\frac{Q_h}{Q_{h,pot}} \approx \alpha \tag{5.57}$$

$$\frac{1}{Q_{h,pot}} \frac{\partial Q_h}{\partial t} \approx \frac{\partial \alpha}{\partial t} = \tilde{A}_{25} \tag{5.58}$$

where $Q_{h,pot}$ is potential hydration heat, [J/kg]. Hence the normalized heat flow $\frac{Q_h}{Q_{h,pot}}$ under isothermal 25°C equals to chemical affinity $\tilde{A}_{25}$.

Cervera et al. (Cervera, Oliver et al. 1999) developed an analytical form of the affinity which was refined in (Gawin, Pesavento et al. 2006). A slightly modified formulation is proposed here

$$\tilde{A}_{25} = B_1\left(\frac{B_2}{\alpha_\infty} + \alpha\right)\left(\alpha_\infty - \alpha\right)\exp\left(-\bar{\eta}\frac{\alpha}{\alpha_\infty}\right) \tag{5.59}$$

where $B_1, [s^{-1}], B_2$, [-] are coefficients to be calibrated, α_∞ is the ultimate hydration degree, [-], and $\bar{\eta}$ represents microdiffusion of free water through formed hydrates, [-]. The parameters in (5.59) express isothermal hydration at 25◦C.

When hydration proceeds under varying temperature, maturity principle expressed via Arrhenius equation scales the a□nity to arbitrary temperature T

$$\begin{aligned} A_r &= \exp\left[\frac{E_a}{R}\left(\frac{1}{273.15+25} - \frac{1}{T}\right)\right] \\ \tilde{A}_T &= \tilde{A}_{25} A_r \end{aligned} \tag{5.60}$$

For example, simulating isothermal hydration at 35°C means scaling $\tilde{A}_{25}$ with a factor of 1.651 at a given time. This means that hydrating concrete for 10 hours at 35°C 35◦C releases the same amount of heat as concrete hydrating for 16.51 hours under 25◦C. Note that setting $E_a = 0$ ignores the e□ect of temperature and proceeds the hydration under 25◦C.

Gawin *et al.* (Gawin, Pesavento et al. 2006), among others, added the effect of relative humidity. The extension of (5.58) leads to

$$\frac{1}{Q_{h,pot}}\frac{\partial Q_h}{\partial t} \approx \frac{\partial \alpha}{\partial t} = \tilde{A}_T \beta_h \tag{5.61}$$

$$\beta_h = \frac{1}{1 + \left(a - ah\right)^4} \tag{5.62}$$

where $\beta_h = \beta_h(h)$ accounts for the reduction of capillary moisture. h is relative humidity r, (Bazant and Najjar 1972). a is material parameter, typically $a = 7.5$. Depending on curing conditions α_∞ is calculated as follows:

Sealed curing:

$$\alpha_\infty = \frac{w/c}{0.42}, \alpha_\infty \le 1 \tag{5.63}$$

Saturated curing:

$$\alpha_\infty = \frac{w/c}{0.36}, \alpha_\infty \le 1 \tag{5.64}$$

w/c is the water-cement ratio.

Substituting (5.59) and (5.62) into (5.61) yields final equation to predict the development of hydration heat. As it is difficult to express α function analytically (from (5.59), (5.61)), the above equations are integrated numerically.

$$\frac{\partial \alpha(t)}{\partial t} = \tilde{A}_{25}(t_{25}) A_r \beta_h$$
$$\alpha(t_{end}) = \alpha(t_{start}) + \int_{t_{start}}^{t_{end}} \tilde{A}_{25}(\tau_{25}) A_r \beta_h \, d\tau \tag{5.65}$$

Substituting $d\tau_{25} = A_r \beta_h d\tau$

$$\alpha(t_{end}) = \alpha(t_{start}) + \int_{t_{25\,start}}^{t_{25,end}} \tilde{A}_{25}(\tau_{25}) d\tau_{25} \tag{5.66}$$

If the function $DoH_{25}(t) = \alpha_{25}(t)$ at reference temperature is known, (e.g. it was meassured in a calorimeter), $A_r \beta_h$ is constant within $< t_{25\,start} ... t_{25\,end} >$ and it is acceptable to use linear (Taylor) approximation of $\alpha_{25}(t_{25}) = \alpha_{25}(t_{25\,start}) + \frac{\partial \alpha}{\partial t_{25}}(t_{25} - t_{25\,start})$ within $< t_{25\,start} ... t_{25\,end} >$, we can write:

$$\frac{\partial \alpha_{25}(t_{25})}{\partial t_{25}} = \tilde{A}_{25}(t_{25})$$
$$\alpha(t_{end}) = \alpha(t_{start}) + \int_{t_{25\,start}}^{t_{25,end}} \frac{\partial \alpha_{25}(\tau_{25})}{\partial \tau_{25}} d\tau_{25} = \alpha_{t_{start}} + \left[\alpha_{25}(t_{25end}) - \alpha_{25}(t_{25\,start})\right] \tag{5.67}$$

In the above $t_{25\,start} = \alpha_{25}^{-1}(\alpha_{start})$, $t_{25\,end} = t_{25\,start} + (t_{end} - t_{start}) A_r \beta_h$ are equivalent time for the case of reference temperature. $\alpha_{25}^{-1}(...)$ is inverse function to $\alpha_{25}(...)$ so that $\alpha_{25}(\alpha_{25}^{-1}(\alpha)) = \alpha$.

Note that Q_h is calculated in the same unit as is entered the parameter $Q_{h,pot}$. If the governing equations are written for unit volume and $Q_{h,pot}$ is given per cement unit weight, then Q_h must be multiplied by fraction of cement mass m_{cement} and total volume of concrete V_{tot}.

Heat capacity

The model assumes the following components of concrete: aggregate, filler, water, and cement. The total mass of concrete in one cubic meter results from individual masses of components:

$$m_{concr} = m_{aggregate} + m_{filler} + m_{paste}$$
$$m_{paste} = m_{cement} + m_{water} \tag{5.68}$$

where m_{concr} is the mass of concrete per a unit volume. Similarly, for the mass of aggregate $m_{aggregate}$, the mass of filler m_{filler}, the mass of water m_{water}, and mass of cement m_{cement}. Corresponding volumes are $V_{aggregate} = m_{aggregate} / \rho_{aggregate}$, $V_{filler} = m_{filler} / \rho_{filler}$ etc. ρ_i stands for the mass density of the phase *i*. Having total volume $V_{concr} = V_{aggregate} + V_{filler} + V_{water} + V_{cement}$, we can calculate phase fractions $f_{aggregate} = V_{aggregate} / V_{concr}$ and similarly for the remaining phases.

Heat capacity and its evolution of cement paste (cement+water) were studied in (Bentz 2007) at 23^0C for w/c between 0.3 and 0.5. The capacity of fresh cement paste yields

$$C_{concrete} = f_{aggregate} C_{aggregate} + f_{filler} C_{filler} + \hat{C}_{paste} \tag{5.69}$$

where $C_{concrete}$ is the concrete capacity (per unit volume) and akin for aggregate, filler, and cement paste. The last term, i.e., C_{paste} also depends on the degree of hydration α and is calculated by

$$\hat{C}_{paste} = (f_{cement} C_{cement} + f_{water} C_{water})\left(1 - 0.26(1 - \exp(-2.9\alpha))\right) \tag{5.70}$$

where C_{cement} is cement capacity at time zero.

The heat capacity of structural concrete spans the range between 0.8 and 1.17 $Jg^{-1}K^{-1}$. A former Czech standard CSN 731208 declares 840 and 870 $Jkg^{-1}K^{-1}$ for dry and saturated mature concrete, respectively. $C_{aggregate}$ is approximately 840 $Jkg^{-1}K^{-1}$ for basalt and limestone, 790 $Jkg^{-1}K^{-1}$ for granite, 800 $Jkg^{-1}K^{-1}$ for sand. C_{cement} is about 750 $Jkg^{-1}K^{-1}$ and C_{water} is 4180 $Jkg^{-1}K^{-1}$.

Heat conductivity

The thermal conductivity of cement paste was found to remain in the range 0.9-1.05 $Wm^{-1}K^{-1}$ for an arbitrary degree of hydration, for both sealed and saturated curing conditions, and for w/c from 0:3 to 0.4 (Bentz 2007). Water in the capillaries has a thermal conductivity 0.604 $Wm^{-1}K^{-1}$ (Bentz 2007). The thermal conductivity of hardened concrete varies between 0.85 and 3.5 $Wm^{-1}K^{-1}$ (Neville 1997) p.375, depending strongly on an aggregate type.

Thermal conductivity also depends on the saturation state of concrete. For example, a structural concrete made from normal-weight aggregate with a unit mass of 2240 kg/m^3 yields λ = 1.696 $Wm^{-1}K^{-1}$ for protected and 1.904 for weather-exposed conditions (Neville 1997), p. 376.

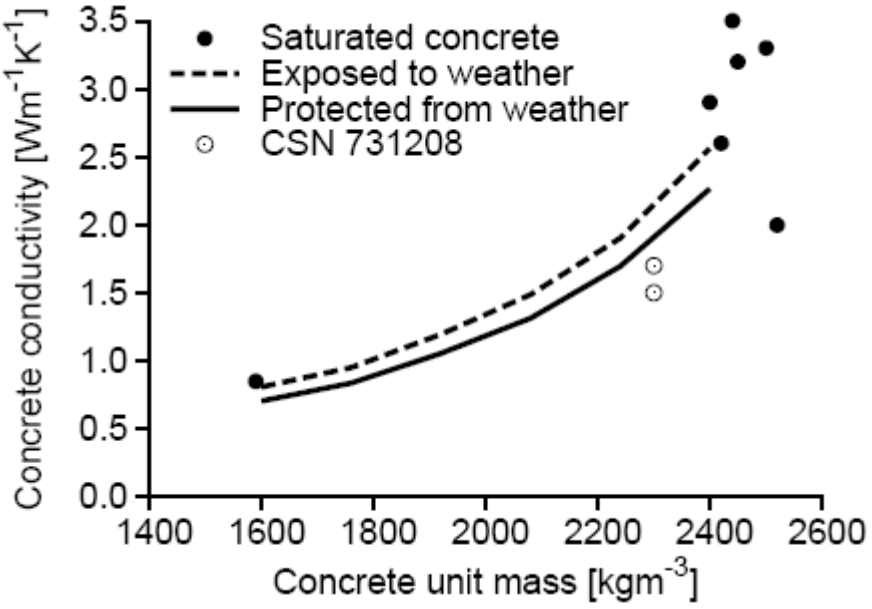


Figure 7-1. Thermal conductivity of concrete according to the Czech code CSN 731209.

Figure 7-1 summarizes thermal conductivities for ordinary concrete depending on concrete unit mass and saturation conditions, according to (Neville 1997) and a former Czech standard CSN

731208. The latter considers 1.5 for a dry concrete and 1.7 $Wm^{-1}K^{-1}$ for a water-saturated concrete.

Faria et al. (Faria, Azenha et al. 2006) applied the evolution of concrete conductivity with regards to α

$$\lambda = \lambda_0 \left(1.0 - 0.248\alpha\right)$$

where λ_∞ is the conductivity of fully hardened concrete, i.e., in infinite time.

The model implemented in Atena, i.e., CCTransportMaterialLevel3 stems from homogenization theories. Consider conductivity of cement paste λ_{paste} and aggregates $\lambda_{aggregate}$ such that $\lambda_{paste} \le \lambda_{aggregate}$. Corresponding volume fractions are $f_{paste}, f_{aggregate}$. Hashin-Shtrikman lower $\lambda_{concrete,low}$ and upper bounds $\lambda_{concrete,upper}$ are (Bentz 2007)

$$\begin{aligned}
\lambda_{concrete,low,\infty} &= \lambda_{paste} + \frac{3 f_{aggregate} \lambda_{paste} \left(\lambda_{aggregate} - \lambda_{paste}\right)}{3\lambda_{paste} + f_{paste}\left(\lambda_{aggregate} - \lambda_{paste}\right)} \\
\lambda_{concrete,upper,\infty} &= \lambda_{aggregate} + \frac{3 f_{paste} \lambda_{aggregate} \left(\lambda_{paste} - \lambda_{aggregate}\right)}{3\lambda_{aggregate} + f_{aggregate}\left(\lambda_{paste} - \lambda_{aggregate}\right)}
\end{aligned} \tag{5.71}$$

The presented model uses average conductivity, i.e.

$$\lambda_{concrete} = \frac{\lambda_{concrete.low,\infty} + \lambda_{concrete,upper,\infty}}{2}\left(1.33 - 0.33\alpha\right) \tag{5.72}$$

Figure 7-2 considers λ_{paste}=1.0 $Wm^{-1}K^{-1}$ and $\lambda_{aggregate}$ = 2.0 $Wm^{-1}K^{-1}$. Volume fraction of aggregates varies from 0 to 1. Important thermal conductivities: limestone 1.26 - 1.33, sandstone 1.7, granite 1.7 - 4.0 $Wm^{-1}K^{-1}$.

The above equations for homogenization are written for phases paste-aggregates. In ATENA, the homogenization is carried out as follows:

1. homogenize phases cement - water -> phase paste.
2. homogenize phases paste - filler -> phase paste with filler
3. homogenize phases paste with filler - air -> phase paste with filler and air
4. homogenize phase paste with filler and air - aggregates -> concrete

Note that filler and aggregate are in this case treated as one component, and the same applies for water and cement (being the component paste). The volume averaging technique is used to calculate the corresponding properties of paste and mixed aggregate.

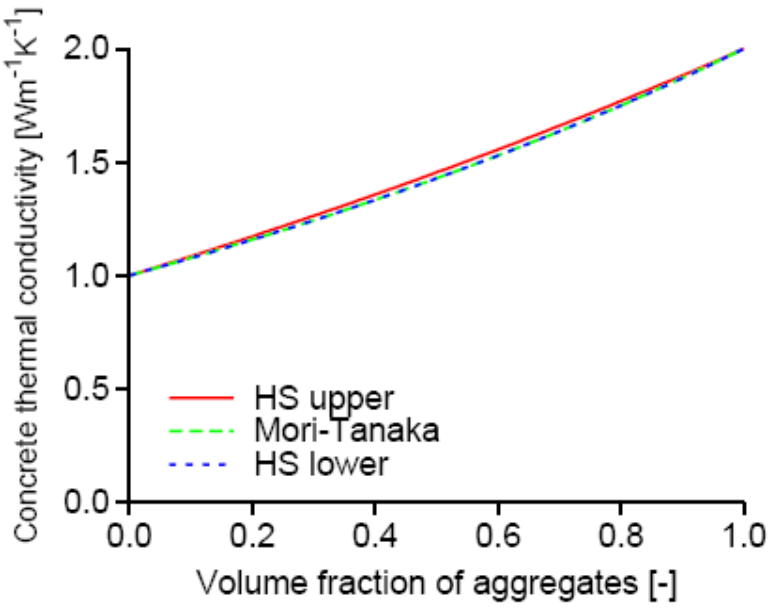


Figure 7-2 Predicted thermal conductivities of concrete from bounds.

Moisture consumption due to hydration

It is assumed that 1 kg of cement (in concrete) approximately consumes during the full hydration process about $Q_{w,pot}$ of water. Typically $Q_{w,pot}$ =0.42 kg of water per 1 kg of cement. Thus, e.g. concrete mixture with 300kg cement per 1m^3 of concrete needs 300*0.42=126kg o water per 1m^3 of concrete. Assuming linear dependence of water hydration consumption w^h on concrete hydration level α, ($\alpha = 0$ for fresh concrete and $\alpha = 1$ for fully hydrated concrete) the water sink term due to hydration is

$$C_{h,t} = \frac{\partial w_h}{\partial t} = \frac{\partial w_h}{\partial \alpha}\frac{\partial \alpha}{\partial t} \tag{5.73}$$

$$w_h = Q_{w,pot}\, c\, \alpha, [\text{kg}] \tag{5.74}$$

where c stands for weight of cement in 1m^3 of concrete.

Moisture capacity

The moisture content at unit volume $w, [\text{kgm}^{-3}]$ is calculated a simple expression

$$w = w_f \frac{(b-1)h}{b-h} \tag{5.75}$$

where $w_f, [\text{kgm}^{-3}]$ is the free water saturation and b is the dimensionless approximation factor, which must always be greater than one. It can be determined from the equilibrium water content w_{80} at relative humidity $h = 0.8$ by substituting the corresponding numerical values in equation (5.75):

$$b = \frac{h(w_f - w_{80})}{w_f h - w_{80}} \tag{5.76}$$

Moisture capacity $C, \left[\text{kgm}^{-3}\right]$ is calculated as derivative of moisture content with respect to h:

$$C_h = \frac{\partial w}{\partial h} = \frac{w_f (b-1) b}{(b-h)^2} \tag{5.77}$$

The above expression is applicable for analyses using reference unit volume. If reference unit weight of the structure is preferred, then we employ moisture capacity $\tilde{C} = C / \rho, \left[\text{kg/kg}\right]$, where ρ is concrete density, $\left[\text{kg/m}^3\right]$.

Moisture diffusion

The present model accounts for the diffusivity mechanism of moisture transport. It is valid for dense concrete, which has not mutually connected pores and moisture convection thru pores (being driven by water pressure) can be neglected. Hence, moisture flux $q_h, \left[\frac{kg}{m^2 s}\right]$ is calculated by the equation $\underline{q}_h = -\mathbf{D}_h \nabla h$, where total moisture diffusivity $D_h, \left[\frac{kg}{m s}\right]$ is calculated as sum of water D_h^w and water vapor D_h^{wv} diffusivity:

$$D_h = D_h^w + D_h^{wv} \tag{5.78}$$

Water liquid diffusivity D_h^w is calculated

$$D_h^w = D_w^w \frac{\partial w}{\partial h} \tag{5.79}$$

where water diffusivity $D_w^w, \left[m^2 / s\right]$ is

$$D_w^w = \frac{3.8\, A^2 1000^{\left(\frac{w}{w_f}-1\right)}}{\left(w_f\right)^2} \tag{5.80}$$

and A is the water absorption coefficient $\left[\frac{kg}{m^2 s^{0.5}}\right]$.

Water vapor permeability is computed from water vapor pressure-driven diffusivity $D_p^{wv}, \left[\frac{\text{kg}}{\text{m s Pa}}\right]$:

$$D_p^{wv} = \frac{\delta}{\mu} \tag{5.81}$$

where μ is the water vapor diffusion resistance factor and δ is the vapor diffusion coefficient in air $\left[\frac{\text{kg}}{\text{m s Pa}}\right]$

$$\delta = \frac{0.00002306\, p_a}{\frac{R}{M_w}(T+273.15)\, p_a}\left(\frac{T+273.15}{273.15}\right)^{1.81} \tag{5.82}$$

Atmospheric pressure $p_a = 101325\text{Pa}$, gas constant $R = 8314.41\text{Jkmol}^{-1}\text{K}^{-1}$ and molar mass of water is $M_w = 18.01528\text{kgkmol}^{-1}$

As in the presented model, relative humidity *h* is the primary variable used to analyze moisture transport, D_p^{wv} must be transformed to D_h^{wv} . This is done by:

$$D_h^{wv} = D_p^{wv}\frac{\partial p}{\partial h} = D_p^{wv}\frac{\partial (p_{sat} h)}{\partial h} = D_p^{wv} p_{sat} \tag{5.83}$$

Any expression to calculate the pressure of saturated water vapor can be used. The presented model uses

$$p_{sat} = 611 e^{\left(\frac{aT}{T_0+T}\right)}, \left[\text{Pa}\right] \tag{5.84}$$

In the above T is temperature $[^{\circ}\text{C}]$ and the remaining parameters are $T \geq 0 : T_0 = 234.18^{o}C, a = 17.08;\ T < 0 : T_0 = 272.44^{o}C, a = 22.44$

Some guidelines towards the model's parameters

Fitted parameters for cement paste hydration need to be considered for each concrete separately. Due to high cement variability, it is impossible to assign one particular cement to one concrete grade. The user needs first to select the cement parameters from the following table:

Table 7.3-1 Parameters of affinity hydration model used for CEM I.

ATENA input notation Notation in Eq. (4)	B1 B_1 [h^{-1}]	B2 B_2 [-]	ETA η [-]	QH_POT Q_{pot} [J/g]	ALPHAINF α_∞ [-]	EA E_a [J/mol]
CEM I 32.5R [10]	3.0e+6	1.4e-3	7.0	471.15	0.85	38300
CEM I 42.5R Mokrá	6.5e+6	8.0e-6	7.4	495.33	0.85	38300
CEM I 42.5R Prachovice [16]	5.0e+6	7.0e-6	6.7	509.21	0.85	38300
CEM I 52.5R ENCI [13]	7.0e+6	6.0e-5	5.8	517.6	0.85	38300
CEM I 52.5 [15]	7.0e+6	6.0e-5	5.8	505.9	0.85	38300
CEM I 52.5 Aalborg-white	7.0e+6	6.0e-5	5.8	448.7	0.85	38300

The above table is based on fitting predicted results from CEMHYD3D analysis by (5.59), see Table 7.3-5 and Figure 7-3. The simulations were carried out on CEMHYD3D's microstructures 50 × 50 × 50 μm and with the activation energy 38.3 kJ/mol. Saturated curing conditions were

assumed since sealed conditions will be obtained from coupling with moisture transport. Table 7.3-5 specifies input data for selected Portland cements.

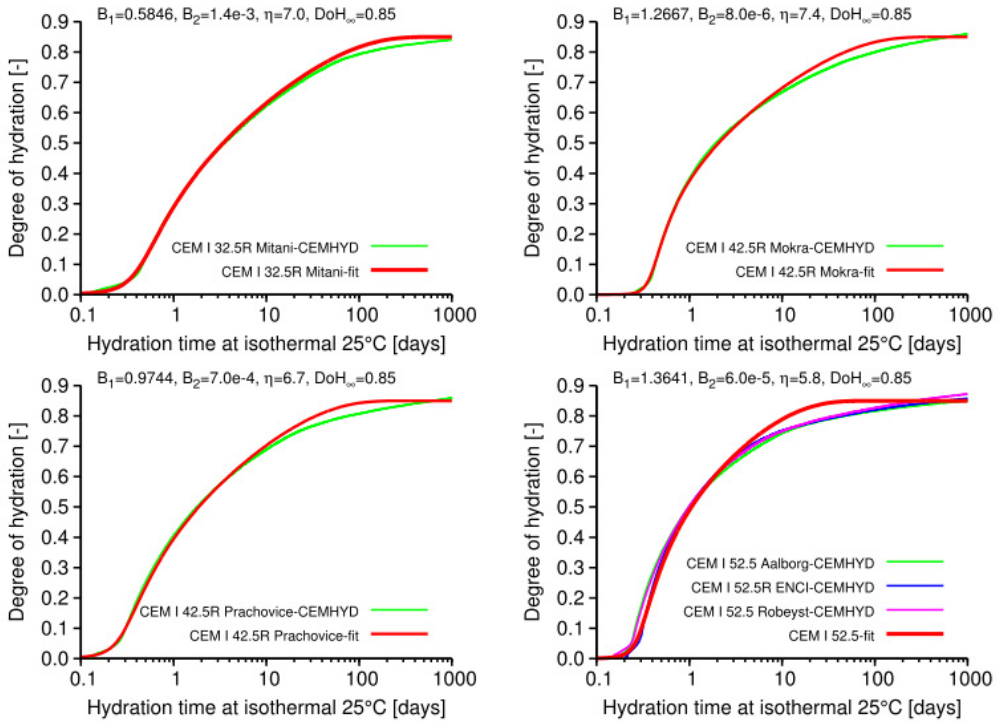


Figure 7-3 Fit of selected cements to the affinity model, w/c = 0.4

The majority of concretes is produced from blended cements (CEM II - CEM V); hence it is necessary to scale down Q pot by approximately 30 %. This is a common Portland clinker substitution in the majority of blended cements in Europe.

There are other default parameters, which are not specified here: QW POT= 0.42, TH INIT = 0, ALPHA INIT = 0, TEMPERATURE INCR MAX =0.1, H80 = 0.8, TEMP0 = 234.18, A WV = 17.08, TEMP0 ICE = 272.44 ,A WV ICE = 22.44

The parameter A ≈ 7.5 expresses hydration slow-down with regards to relative humidity. The hydration practically stops at $\alpha \approx 0.8$.

Parameters in Figure 7-1 are computed for saturated state. When $\alpha = 1$, the hydration proceeds as there is saturated water environment around concrete. Under standard circumstamces, hydration consumes water, which decreases relative humidity in the calculation. Three parameters are related to moisture transport and are given for ordinary structural concrete:

- W80 expresses total mass of free water at α=80%. Standard value is 50 kg/m 3 for structural concrete.
- A W is water absorption coe□cient, whose value spans the range 0.25-0.846 kgm− 2 h 0.5].
- MI WV is the water vapor di□usion resistance factor, spanning 210-260 [-] for structural concrete.

Parameters specifying specific heat capacity for concrete components are summarized in Table 7.3-2. Values are obtained from http://www.engineeringtoolbox.com/density-solids-d_1265.html, http://www.engineeringtoolbox.com/specific-heat-solids-d_154.html

Parameters specifying specific heat conductivity for concrete components are summarized in Table 7.3-3. Sources from http://www-odp.tamu.edu/publications/192_SR/109/109_5.htm

Concrete strength classes strongly depend on the amount of cement in concrete. Table 7.3-4 specifies approximate compositions for major concrete classes used in EN 206-1. The calculation assumes 5 % of entrained air in the concrete, cement density 3220 kg/m 3 and aggregate density 2800 kg/m 3 .

Table 7.3-2 Parameters specifying density and specific heat capacity for concrete components

ATENA's Parameter	Component	Density [kg/m^3]	Capacity$_m$ [J/g/K]	Capacity$_V$ [J/m^3/K]
C_AGGREGATE_TEMP_TEMP	Basalt light	2400	0.84	**2.02e+6**
	Basalt heavy	3100	0.84	**2.60e+6**
	Granite	2700	0.79	**2.13e+6**
C_FILLER_TEM_TEMP	Fly ash light	1900	0.84	**1.60e+6**
	Fly ash heavy	2400	0.84	**2.02e+6**
	Limestone	2750	0.84	**2.31e+6**
C_CEMENT_TEMP_TEMP	Portland cement	3220	0.75	**2.42e+6**
C_WATER_TEMP_TEMP	Water	1000	4.18	**4.18e+6**

Table 7.3-3 Parameters specifying specific heat conductivity for concrete components

ATENA's Parameter	Conductivity [W/m/K]
K_AGGREGATE_TEMP_TEMP	Basalt 1.7-2.0
	Granite 1.7-4.0
K_FILLER_TEMP_TEMP	Fly ash 0.4 [14]
	Limestone 1.26–1.33
K_CEMENT_TEMP_TEMP	Portland cement 1.55 [2]
K_WATER_TEMP_TEMP	0.604

Ready-mix concrete is assumed, which requires rather higher w/c due to workability and pumping issues. The parameters CEMENT DENSITY, WATER DENSITY, AGGREGATE DENSITY, FILLER DENSITY are provided in Table 7.3-2 in the units [kg/m 3].

Table 7.3-4 Approximate composition for major concrete classes used in EN206-1

Concrete class	w/c [-]	CEMENT_MASS [kg/m^3]	W_F [kg/m^3]	AGGREGATE_MASS [kg/m^3]	FILLER_MASS [kg/m^3]
C8/10	0.55	200	110	2202	0
C12/15	0.55	210	115.5	2179	0
C16/20	0.55	230	126.5	2134	0
C20/25	0.55	250	137.5	2088	0
C25/30	0.50	280	140	2056	0
C28/35	0.50	300	150	2013	0
C32/40	0.50	330	165	1948	0
C35/45	0.50	350	175	1905	0
C40/50	0.45	380	171	1891	0
C45/55	0.45	400	180	1851	0
C50/60	0.40	430	172	1847	0
C55/67	0.38	450	171	1833	0
C60/75	0.35	480	168	1817	0

Table 7.3-5 CEMHYD3D parameters for fittiong of affinity hydration model.

Acronym, refer.	C_3S^a	C_2S^a	C_3A^a	C_4AF^a	Gypsum[b]	Fineness	W/c	Temp.	E_a	Q_{pot}	t_0	β
	Mineral composition [wt. %]				[vol. %]	[m^2/kg]	[-]	[°C]	[kJ/mol]	[J/g_{cem}]	[h]	[h/$cycle^2$]
CEM I 32.5R [10]	50.86^d	24.45^d	7.85^d	7.50^d	4.47	277	0.40	25.0	$38.3^\dagger$	471.151	$3.0^\flat$	3.0E-4
CEM I 42.5R Mokrá^{CTU}	64.35^{dk}	11.81^{dk}	3.99^{dk}	11.87^{dk}	$5.0^\flat$	306	0.40	25.0	$38.3^\dagger$	495.33	6.0	3.0E-4
CEM I 42.5R Prachovice [16]	56.33^d	18.80^d	7.83^d	10.92^d	4.7	320	0.40	25.0	$38.3^\dagger$	509.21	$3.0^\flat$	3.0E-4
CEM I 52.5R ENCI [13]	66.88^{de}	11.31^{de}	6.16^{de}	9.89^{de}	$5.1^\flat$	530	0.40	20.0	$38.3^\dagger$	517.6	5.0	3.0E-4
CEM I 52.5 [15]	67.13^{df}	6.64^{df}	5.59^{df}	10.68^{df}	$5.0^\flat$	390	0.40	20.0	$38.3^\dagger$	505.9	3.5	3.0E-4
CEM I 52.5 Aalborg-white^{w,CTU}	66.60^c	23.80^c	3.40^c	0.40^c	3.6	390	0.40	20.0	$38.3^\dagger$	448.7	4.0	3.0E-4

a	Four clinker minerals are later normalized in the CEMHYD3D input routine	j	Best fit to measured PSD
b	Clinker volume is replaced by gypsum or anhydrate in specified volume fractions	k	Assumed 1.0 wt. % of free lime
c	Mineral composition determined directly from XRD / Rietveld analysis	n	Normalized
CTU	Tested at Czech Technical University in Prague (CTU)	s	Saturated curing conditions
d	Based on the Taylor formulas	w	White cement
e	Assumed no free lime	$\flat$	Estimated
f	Assumed 1.5 wt. % of free lime	$\dagger$	Irrelevant
g	Specified in the reference based on the Bogue calculation		
h	Based on the original Bogue calculation.		
i	Given 1.5 wt. % free lime, anhydrate 2.94 vol. % and gypsum 3.31 vol. %		

7.4 Fire Element Boundary Load

When undertaking heat transfer calculations, it is important that relevant thermal properties of materials and heat transfer coefficients at the boundaries are defined for the entire temperature interval of the load.

7.4.1 Hydrocarbon Fire

Hydrocarbon fires are those sustained by hydrocarbon-based products, such as chemicals, gas, and petroleum. Depending on the heat load, different HC-curves can be derived in accordance with Equation (5.85). The magnitude of the maximum temperature of the radiation source (T_1) is crucial for the time temperature development. The nominal HC-curve is represented by the heat load 200 kW/m2 and reaches maximum temperature of 1100 °C. The curve representing 345 kW/m2 is called the "modified" or "increased" HC-curve for tunnel applications. It reaches at maximum 1300 °C.

$$T(t) = T_1(1 - 0.325e^{-0.167t} - 0.204e^{-1.417t} - 0.471e^{-15.833t}) \tag{5.85}$$

where:

$T(t)$= temperature of radiation source as function of time [°C],

T_1= maximum temperature of radiation source [°C] according to (5.85)

t = time [minutes]

Time development of temperature of the radiation source is depicted in the figure below. For time $t \to 0$ Equation (5.85) yields $T(0) = 0$ and hence, it is necessary to supplement (5.85) by requirement $T(t) \geq T_{ambiant,ini}$, where $T_{ambiant,ini}$ is initial ambient temperature prior the fire broke up, (typically something about 20 °C).

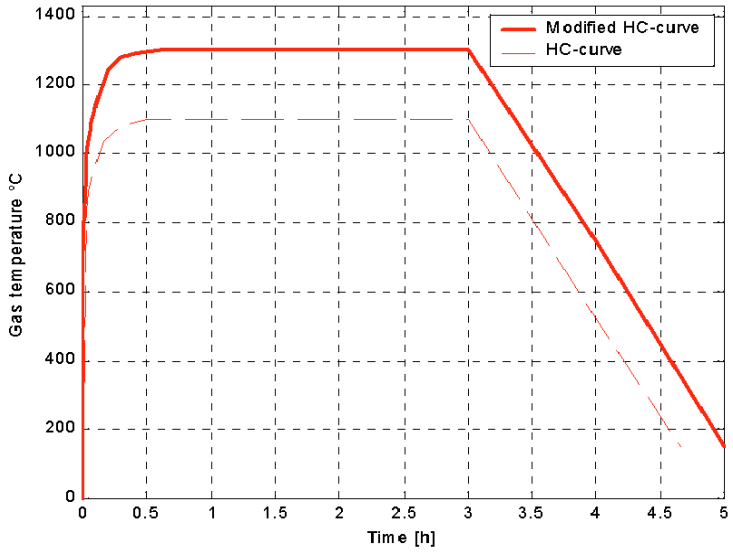


Fig. 7-3 Temperature of radiation source

7.4.2 Fire Exposed Boundary

The nature of the structural ambient conditions is essential for the determination of the temperature fields. Depending on the geometry, view factors, and ambient conditions, various types of boundary conditions may be considered.

Fire exposed boundary

The heat is transferred from the fire gas to the exposed structure through radiation and convection. At high temperatures, the radiation dominates. The radiation is expressed by the resulting emissivity factor, which takes into account the emissivity of the fire source ε, and absorptivity of the heated surface α. The convection is calculated from the temperature difference between the structure and ambient gas, depending on the gas velocity. Emissivity and convection factors used for exposed surfaces are shown below

$$\begin{aligned} \varepsilon_r &= 0.56, \quad [-] \\ h_c &= 50, \quad \left[\frac{W}{m^2 K}\right] \end{aligned} \tag{5.86}$$

The convection and emissivity heat flux on a boundary exposed to fire is calculated as follows:

$$q_n = h_c(T_g - T_b) + \varepsilon_r \sigma (T_g^4 - T_b^4) \tag{5.87}$$

where

σ = Stefan-Boltzmann constant [5.67x10^{-8} W/m^2 K^4],

T_g = absolute temperature of radiation source [K],

T_b =boundary temperature of the structure,

ε_r = resulting emissivity factor of the radiation source and the heated surface [-],

q_n = heat flow at the fire exposed boundary [W/m^2],

h_c = convection heat transfer coefficient [W/m^{2}K].

Adiabatic boundary

Adiabatic boundary surface refers to a boundary surface, where no heat can pass in (and/or out) the structure. Structural symmetry lines and areas are good examples of this boundary conditions.

7.4.3 Implementation of Fire Exposed Boundary in ATENA

The described fire boundary load conditions are ATENA modeled by CCFireElementBoundaryLoad load. It is essentially an element boundary load that applies the heat flow q_n at the element boundary, i.e., at a surface exposed to fire. Unlike other loads in ATENA (that are of incremental nature and constant within one load step), this load is considered variable and has kind of a total load.

Four types of heat source definitions are implemented:

- Nominal HV fire – Temperature of the heat source is calculated by (5.85) and T_1 (unless it is manually inputted as *temp_g_ref*) is set to 1100 [°C].
- Modified HC fire – This definition is much the same as the above with the only difference that default value for T_1 is 1300 [°C].
- Generic fire (also refered to as User curve fire) - Temperature of the heat source is assumed constant and is set value of *temp_g_ref* . If *temp_g_ref* is not inputted, then 1100 [°C] is used.

In any case, the generated (or directly inputted) curve for $T(t)$ can be additionally modified in time by a user-supplied function *time_id.* The function takes one parameter, which is time of the fire and it specifies a coefficient by which the generated initially (or inputted) boundary conditions should be multiplied. Of course, load variation in space can be modified by *coeff_x, coeff_y* coefficients etc. in the same way as for any other generated element load, (for more details see Atena Input file manual).

7.5 Moisture-Heat Element Boundary Load[10]

This type of boundary load is used for modeling heat and moisture fluxes from the structure to the ambient air. Hence, it is typically applied as a boundary element load on external surfaces of the structure. It resembles the fire boundary load described above and is implemented in a similar way.

The moisture flux consists of three parts.

$$q_{h,1} = h_{cw}(h_g - h_b)$$
$$q_{h,2} = \Theta(x_g - x_b)$$
$$q'_{h,3} = 1.38\overline{}E^{-9}\left(\left(\frac{9}{5}T_{Cb}+32\right)^{2,5} - h_g\left(\frac{9}{5}T_{Cg}+32\right)^{2,5}\right)(1+0.9000000000v) \quad [\text{kg/m}^2 s] \quad (5.88)$$
$$q_{h,3} = transform\_units(q'_{h,3})$$
$$q_h = q_{h,1} + q_{h,2} + q_{h,3}$$

The first part $q_{h,1}$ includes moisture flux driven by the gradient between ambient and surface relative humidity h_g and h_b. h_{cw} stands for moisture convection coefficient of the concrete-air interface. The second part $q_{h,2}$, accounts for moisture flux due to evaporation driven by the gradient of humidity air ratio at the interface, i.e., x_b and x_g with the evaporation coefficient Θ. By default $\Theta = (25+19v)$, $\left[\frac{kg}{m^2 s}\right]$, where v is ambient air velocity, [m/s]. For more information, see http://www.engineeringtoolbox.com/evaporation-water-surface-d_690.html.

[10] Available starting from ATENA version 5.

The humidity air ratio, [-] is calculated as follows (*i* reflects conditions in ambient air, i.e., *i*=*g*, or in the surface of the structure i.e., *i*=*b*):

$$x_i = \frac{m_{wv,i}}{m_a} = \frac{\rho_{wv,i}}{\rho_a} \tag{5.89}$$

It is calculated at state variables h_i, T_i, i.e., relative humidity and temperature at *i* conditions.

In the above $m_{wv,i}$, $\rho_{wv,i}, m_a$, ρ_a is mass and density of water vapor in REV corresponding to *i* conditions and mass and density of dry air, [kg/m^3], respectively.

$$\rho_a = \frac{p_a}{\frac{R}{M_a}(T_i + 273.15)} \tag{5.90}$$

where M_a is weight of 1kmol of dry air (assumed $M_a \doteq 28.96\,\text{kg/kmol}$). R is gas constant, (R=8313JK^{-1}), T_i is the temperature in ^{o}C. p_a is the partial pressure of dry air, [Pa]

$$p_a = p - h_i\, p_{vw.sat} \tag{5.91}$$

Here p stands for total air pressure (typically normal air pressure p=101325Pa), h_i is relative humidity and $p_{vw.sat}$ is the partial pressure of saturated water vapor at T_i, (see http://en.wikipedia.org/wiki/Density_of_air)

$$p_{wv,sat} = 610.78\ 10^{\left(\frac{7.5T_i}{T_i+237.3}\right)} \tag{5.92}$$

The density of water vapor at *i* th conditions is calculated similar to (5.90):

$$\rho_{wv,sat} = \frac{p_{wv,sat}}{\frac{R}{M_{wv}}(T_i + 273.15)} \tag{5.93}$$

In the above M_{wv} is the weight of 1kmol of saturated water vapor, (assumed $M_a \doteq 18.06\,\text{kg/km}$.

The third part is moisture flux evaporated from concrete calculated by CEMSTONE, see http://www.cemstone.com/concrete-evaporation-forecast-engineers.cfm. It yields nearly the same values as provided by ACPA calculator; see http://www.apps.acpa.org/apps/EvaporationCalculator.aspx. T_{Cg}, T_{Cb} is the ambient and surface temperatures in Celsia.

The heat flux also consists from two parts.

$$\begin{aligned} q_{T1} &= h_{cT}(T_g - T_b) + \varepsilon_{rT}\sigma(T_{Kg}^4 - T_{Kb}^4) \\ q_{T2} &= q_h h_{we} \\ q_T &= q_{T1} + q_{T2} \end{aligned} \tag{5.94}$$

The first part of the heat flux q_T represents the usual flux due to heat convection and emission. Its computation resembles (5.87). h_{cT} stands for heat convection coefficient of the concrete-air interface $\left[\frac{W}{m^2 K}\right]$, ε_{rT} is heat emissivity coefficient [-], T_{Kg}, T_{Kb} are ambient and surface temperatures in Kelvins and σ is Stephan-Boltzmann constant, $\sigma = 5.67E-8, \left[\frac{W}{m^2 K^4}\right]$. The second part takes into account heat consumption due to the evaporation of moisture flux. By default $h_{we} = 2270, \left[\frac{kJ}{kg}\right]$ is assumed. More information available at http://www.engineeringtoolbox.com/evaporation-water-surface-d_690.html.

Both moisture and heat fluxes are typically computed using only their first or second part. Therefore, the related ATENA input commands allow reading some boolean flags that specify, which parts of the above fluxes should by accounted for and which should be skipped. For more information, refer to the ATENA input file manual.

7.6 References

BAZANT, Z. P. (1986). Mathematical Modelling of Moisture Diffusion and Pore Pressure, Chapter 10. Concrete at High Temperature. Z. P. Bazant**:** 198-237.

BAZANT, Z. P. and W. THONGUTHAI (1978). Pore Pressure and Drying of Concrete at High Temperature. Proceedings of the ASCE.

CELIA, M. A. and P. BINNING (1992). "A Mass Conservative Numerical Solution for Two-Phase Flow in Porous Media with Application to Unsaturated Flow." Water Resour. Res **28**(10): 2819-2828.

CELIA, M. A., T. BOULOUTAS, et al. (1990). "A General Mass-Conservative Numerical Solution for the Unsaturated Flow Equations." Water Resour. Res **27**(7): 1438-1496.

DIERSCH, H. J. G. and P. PERROCHET (1998). On the primary variable switching technique for simulating unsaturated-saturated flows, http://www.wasy.de/eng/prodinfo/flow/swpool/swpool.htm#fef_manuals.

HUGHES, J. R. (1983). Analysis of Transient Algorithms with Particular Reference to Stability Behaviour. Computational Methods for Transient Analysis, Elsevier Science Publishers B.V.

JENDELE, L. (2001). ATENA Pollutant Transport Module - Theory. Prague, Edited PIT, ISBN 80-902722-4-X.

JENDELE, L. and D. V. PHILLIPS (1992). "Finite Element Software for Creep and Shrinkage in Concrete." Computer and Structures **45 (1)**: 113-126.

REKTORYS, K. (1995). Přehled užité matematiky. Prague, Prometheus.

SEAGER, M. K. and A. GREENBAUM (1988). A SLAP for the Masses, Lawrence Livermore National Laboratory.

WOOD., W. L. (1990). Practical-Time Stepping Schemes. Oxford, Clarenton Press.

XI, Y., Z. P. BAZANT, et al. (1993). "Moisture Diffusion in Cementitious Materials, Adsorbtion Isotherms." Advn. Cem. Bas. Mat. **1**: 248-257.

XI, Y., Z. P. BAZANT, et al. (1994). "Moisture Diffusion in Cementitious Materials, Moisture Capacity and Diffusivity." Advn. Cem. Bas. Mat. **1**: 258-266.

ZIENKIEWICZ, O. C. and R. L. TAYLOR (1989). The Finite Element Method, Volume 1: Basic Formulation and Linear Problem. London, McGraw-Hill, 4th edition.

8 DYNAMIC ANALYSIS

ATENA software support four methods to execute dynamic analyses. These are:

- Newmark's β method,
- Hughes α method (Hughes 1983),
- Wilson θ
- Modified Wilson θ.

Note that Hughes α method with $\alpha = 0$ reduces to Newmark's β method and Modified Wilson θ is just an extension to Wilson θ.

The governing equations for dynamic analysis read:

Hughes α method:

$$\mathbf{M}\ddot{\bar{u}}^{t+\Delta t} + \mathbf{C}\left((1+\alpha)\dot{\bar{u}}^{t+\Delta t} - \alpha\dot{\bar{u}}^{t}\right) + \mathbf{K}\left((1+\alpha)\bar{u}^{t+\Delta t} - \alpha\bar{u}^{t}\right) = (1+\alpha)\bar{R}^{t+\Delta t} - \alpha\cdot\bar{R}^{t}$$

Newmark β method:

$$\mathbf{M}\ddot{\bar{u}}^{t+\Delta t} + \mathbf{C}\dot{\bar{u}}^{t+\Delta t} + \mathbf{K}\bar{u}^{t+\Delta t} = \bar{R}^{t+\Delta t}$$

(Modified) Wilson θ method:

$$\mathbf{M}\ddot{\bar{u}}^{t+\theta\Delta t} + \mathbf{C}\dot{\bar{u}}^{t+\theta\Delta t} + \mathbf{K}\bar{u}^{t+\theta\Delta t} = \bar{R}^{t+\theta\Delta t}$$

(5.95)

where

$\ddot{\bar{u}}^{t+\Delta t}, \dot{\bar{u}}^{t+\Delta t}, \bar{u}^{t+\Delta t}$ is acceleration, velocity, and displacement at a time $t+\Delta t$, (similar for time t and $t+\theta\Delta t$),

$\mathbf{M}, \mathbf{C}, \mathbf{K}$ is mass, damping, and stiffness matrix, respectively,

$\bar{R}$ is the vector of external forces, i.e., concentrated loads,

α is the Hughes damping parameter.

They are is solved for displacement at time $t+\Delta t$. The displacement, acceleration and velocity at time $t+\Delta t$ is calculated as functions of (already known) $\ddot{\bar{u}}^{t}, \dot{\bar{u}}^{t}, \bar{u}^{t}$ and displacement increments $\Delta\bar{u}^{t+\Delta t} + \Delta u$. If *l*-th iteration is solved, then we solve for displacement increment Δu and $\Delta\bar{u}^{t+\Delta t} = \sum_{k=1}^{l}\Delta u_k$

Hughes α method:

Newmark β method:

$$\overline{u}^{t+\Delta t} == \overline{u}^{t} + \Delta\overline{u}^{t+\Delta t} + \Delta\overline{u}$$

$$\dot{\overline{u}}^{t+\Delta t} = \left(\frac{1}{2}\frac{\left(2\beta\Delta t^2 - \Delta t^2\right)\gamma}{\Delta t\beta} + \Delta t\left(1-\gamma\right) \right)\ddot{\overline{u}}^{t} + \frac{1}{2}\frac{\left(-2\dot{\overline{u}}^{t}\Delta t + 2\Delta\overline{u} + 2\Delta\overline{u}^{t+\Delta t}\right)\gamma}{\Delta t\beta} + \dot{\overline{u}}^{t}$$

$$\ddot{\overline{u}}^{t+\Delta t} = \frac{1}{2}\frac{2\ddot{\overline{u}}^{t}\beta\Delta t^2 - \ddot{\overline{u}}^{t}\Delta t^2 - 2\dot{\overline{u}}^{t}\Delta t + 2\Delta\overline{u} + 2\Delta\overline{u}^{t+\Delta t}}{\Delta t^2\beta}$$

Modified Wilson θ method:

Wilson θ method:

$$\overline{u}^{t+\Delta t} = \overline{u}^{t} + \Delta\overline{u}^{t+\Delta t} + \Delta u$$

$$\dot{\overline{u}}^{t+\Delta t} = -\frac{3\overline{u}^{t}}{\Delta t} - 2\dot{\overline{u}}^{t} - \frac{1}{2}\ddot{\overline{u}}^{t}\Delta t + \frac{3(\overline{u}^{t} + \Delta\overline{u}^{t+\Delta t} + \Delta\overline{u})}{\Delta t}$$

$$\ddot{\overline{u}}^{t+\Delta t} = -\frac{6\overline{u}^{t}}{\Delta t^2} - \frac{6v\_t}{\Delta t} - 2\ddot{\overline{u}}^{t} + \frac{6(\overline{u}^{t} + \Delta\overline{u}^{t+\Delta t} + \Delta\overline{u})}{\Delta t^2} \tag{5.96}$$

Substituting (5.96) into (5.95) and after some mathematical manipulation, the requested displacement increment at iteration *l* can be calculated:

$$\Delta\overline{u} = \mathbf{K}_{eff}^{inv}\,\overline{R}_{eff} \tag{5.97}$$

where (for using structural damping $\mathbf{C} = \delta_M\mathbf{M} + \delta_K\mathbf{K}$) effective stiffness and RHS vector are:

$$\begin{aligned} \mathbf{K}_{eff} &= \mathbf{M}\overline{\zeta}_M + \mathbf{K}\overline{\zeta}_K \\ \overline{R}_{eff} &= \mathbf{M}(\overline{\xi}_M^1 + \overline{\xi}_M^2) + \mathbf{K}(\overline{\xi}_K^1 + \overline{\xi}_K^2) + \overline{\xi}_0 \end{aligned} \tag{5.98}$$

The coefficients above are calculated using the following expressions. They are summarized (by solution method):

Hughes α method:

$$\overline{\zeta}_M = \left(\left(\frac{\left(\frac{1}{2}\alpha + \frac{1}{2}\right)\delta_M \gamma}{\beta} + \left(-\alpha - 1\right)\delta_M \right) \Delta t - 1 + \frac{1}{2\beta} \right) \ddot{\overline{u}}^t + \left(\frac{\left(1+\alpha\right)\delta_M \gamma}{\beta} - \delta_M + \frac{1}{\Delta t \beta} \right) \dot{\overline{u}}^t -$$

$$\frac{\Delta \overline{u}^{t+\Delta t}}{\Delta t^2 \beta} + \frac{\left(-\alpha \Delta \overline{u}^{t+\Delta t} - \Delta \overline{u}^{t+\Delta t}\right)\gamma \delta_M}{\beta \Delta t}$$

$$\overline{\zeta}_K = \left(\frac{\left(\frac{1}{2}\alpha + \frac{1}{2}\right)\delta_K \gamma}{\beta} + \left(-\alpha - 1\right)\delta_K \right) \Delta t \ddot{\overline{u}}^t + \left(\frac{\left(1+\alpha\right)\delta_K \gamma}{\beta} - \delta_K \right) \dot{\overline{u}}^t + \frac{\left(-\alpha \Delta \overline{u}^{t+\Delta t} - \Delta \overline{u}^{t+\Delta t}\right)\gamma \delta_K}{\beta \Delta t}$$

$$\overline{\xi}_M^1 = \left(\left(\frac{\left(\frac{1}{2}\alpha + \frac{1}{2}\right)\delta_M \gamma}{\beta} + \left(-\alpha - 1\right)\delta_M \right) \Delta t - 1 + \frac{1}{2\beta} \right) \ddot{\overline{u}}^t + \left(\frac{\left(1+\alpha\right)\delta_M \gamma}{\beta} - \delta_M + \frac{1}{\Delta t \beta} \right) \dot{\overline{u}}^t$$

$$\overline{\xi}_M^2 = -\frac{\Delta \overline{u}^{t+\Delta t}\left(\alpha \gamma \Delta t \delta_M + \gamma \Delta t \delta_M + 1\right)}{\Delta t^2 \beta}$$

$$\overline{\xi}_K^1 = \left(\frac{\left(\frac{1}{2}\alpha + \frac{1}{2}\right)\delta_K \gamma}{\beta} + \left(-\alpha - 1\right)\delta_K \right) \Delta t \ddot{\overline{u}}^t + \left(\frac{\left(1+\alpha\right)\delta_K \gamma}{\beta} - \delta_K \right) \dot{\overline{u}}^t \qquad (5.99)$$

$$\overline{\xi}_K^2 = -\frac{\Delta \overline{u}^{t+\Delta t}\left(1+\alpha\right)\gamma \delta_K}{\beta \Delta t}$$

$$\overline{\xi}_0 = \overline{R}^{t+\Delta t}(1+\alpha) - \overline{R}^t \alpha - \overline{F}^{t+\Delta t}(1+\alpha) + \overline{F}^t \alpha$$

Newmark β method:

$$\overline{\zeta}_M = \left(\left(\frac{\delta_M \gamma}{2\beta} - \delta_M \right) \Delta t - 1 + \frac{1}{2\beta} \right) \ddot{\overline{u}}^t + \left(\frac{\delta_M \gamma}{\beta} - \delta_M + \frac{1}{\Delta t \beta} \right) \dot{\overline{u}}^t -$$

$$\frac{\Delta \overline{u}^{t+\Delta t}}{\Delta t^2 \beta} + \frac{\left(-\Delta \overline{u}^{t+\Delta t}\right)\gamma \delta_M}{\beta \Delta t}$$

$$\overline{\zeta}_K = \left(\frac{\delta_K \gamma}{2\beta} - \delta_K \right) \Delta t \ddot{\overline{u}}^t + \left(\frac{\delta_K \gamma}{\beta} - \delta_K \right) \dot{\overline{u}}^t + \frac{\left(-\Delta \overline{u}^{t+\Delta t}\right)\gamma \delta_K}{\beta \Delta t}$$

$$\overline{\xi}_M^1 = \left(\left(\frac{\delta_M \gamma}{2\beta} - \delta_M \right) \Delta t - 1 + \frac{1}{2\beta} \right) \ddot{\overline{u}}^t + \left(\frac{\delta_M \gamma}{\beta} - \delta_M + \frac{1}{\Delta t \beta} \right) \dot{\overline{u}}^t$$

$$\overline{\xi}_M^2 = -\frac{\Delta \overline{u}^{t+\Delta t}\left(\gamma \Delta t \delta_M + 1\right)}{\Delta t^2 \beta}$$

$$\overline{\xi}_K^1 = \left(\frac{\delta_K \gamma}{2\beta} - \delta_K \right) \Delta t \ddot{\overline{u}}^t + \left(\frac{\delta_K \gamma}{\beta} - \delta_K \right) \dot{\overline{u}}^t$$

$$\overline{\xi}_K^2 = -\frac{\Delta \overline{u}^{t+\Delta t} \gamma \delta_K}{\beta \Delta t} \qquad (5.100)$$

$$\overline{\xi}_0 = \overline{R}^{t+\Delta t} - \overline{F}^{t+\Delta t}$$

Wilson θ method and Modified Wilson θ method:

$$\overline{\zeta}_M = \frac{3\delta_M}{\theta\Delta t} + \frac{6}{\theta^2\Delta t^2}$$

$$\overline{\zeta}_K = 1 + \frac{3\delta_K}{\theta\Delta t}$$

$$\overline{\xi}_M^1 = \left(\frac{1}{2}\frac{6\theta - 2}{\theta^3} + \frac{1}{2}\frac{\left(3\theta^2 - 2\theta\right)\Delta t\delta_M}{\theta^3} \right)\ddot{\overline{u}}^t + \left(\frac{6}{\theta^2\Delta t} + \frac{1}{2}\frac{\left(6\theta^2 - 2\right)\delta_M}{\theta^3} \right)\dot{\overline{u}}^t$$

$$\overline{\xi}_M^2 = -\frac{3\Delta\overline{u}^{t+\Delta t}\left(\theta\Delta t\delta_M + 2\right)}{\theta^2\Delta t^2}$$

$$\overline{\xi}_K^1 = \left(\frac{1}{2}\frac{\left(3\theta^2 - 2\theta\right)\Delta t\delta_K}{\theta^3} + \frac{1}{2}\frac{\left(\theta^3 - \theta^2\right)\Delta t^2}{\theta^3} \right)\ddot{\overline{u}}^t + \left(\frac{1}{2}\frac{\left(6\theta^2 - 2\right)\delta_K}{\theta^3} + \frac{1}{2}\frac{\left(2\theta^3 - 2\theta\right)\Delta t}{\theta^3} \right)\dot{\overline{u}}^t$$

$$\overline{\xi}_K^2 = -\frac{3\Delta\overline{u}^{t+\Delta t}\delta_K}{\theta\Delta t}$$

Wilson θ method :

$$\overline{\xi}_0 = \frac{\overline{R}^{t+\Delta t}}{\theta^2} + \left(-\frac{1}{\theta^2} + \frac{1}{\theta^3} \right)\overline{R}^t - \overline{F}^{t+\Delta t} + \left(1 - \frac{1}{\theta^3} \right)\overline{F}^t$$

Modified Wilson θ method

$$\overline{\xi}_0 = \left(\frac{1}{\theta} - \frac{1}{\theta^3} \right)\overline{F}^t - \frac{\overline{F}^{t+\theta\Delta t}}{\theta} + \frac{\overline{R}^{t+\theta\Delta t}}{\theta^3} \tag{5.101}$$

The parameters β, γ are the integration parameters used by Newmark β and Hughes α method. Their value is essential for convergence of this time marching scheme. It can be shown that $\gamma = \frac{1}{2}, \beta = \frac{1}{6}$ corresponds to linear acceleration within the time step. Values $\gamma = \frac{1}{2}, \beta = \frac{1}{4}$ yield constant acceleration. The integration scheme is unconditionally stable, if $\gamma \geq \frac{1}{2}, \beta \geq 0.25(\frac{1}{2} + \gamma)^2$ and it is only conditionally stable for $\gamma > \frac{1}{2}, \beta < 0.25(\frac{1}{2} + \gamma)^2$ provided that the stability limit is fulfilled:

$$\omega\Delta t_{crit} = \frac{\xi\left(\gamma - \frac{1}{2}\right) + \left[\frac{\gamma}{2} - \beta + \xi^2\left(\gamma - \frac{1}{2}\right)^2\right]^{\frac{1}{2}}}{\left(\frac{\gamma}{2} - \beta\right)} \tag{5.102}$$

where ξ is the modal damping parameter.

The above defines the condition for time increment Δt for a linear conditionally stable case:

$$\frac{\Delta t}{T_n} \le \frac{1}{\pi\sqrt{2}\sqrt{\gamma - \frac{2}{3}}} \le 0.551 \tag{5.103}$$

As for Wilson θ and Modified Wilson θ method, they use θ parameter. Its value is $\theta \ge 1$ and the scheme is unconditionally stable for $\theta \ge 1.4$. It essentially specifies the time, for which time we calculate the governing equations (5.95), i.e $t + \theta\Delta t$. For $\theta = 1$ Wilson θ and Modified Wilson θ method yield the same solution expressions and equations, and these are also the same as those for Newmark and Hughes methods with $\gamma = \frac{1}{2}, \beta = \frac{1}{6}, \alpha = 0$.

Modified Wilson θ method assembles the governing equations for time $t + \theta\Delta t$. As a result, all Von Neumann boundary conditions must be given for $t + \theta\Delta t$, (e.g., concentrated load, load by MASS_ACCELERATION etc.). It does not apply to Dirichlet boundary conditions that are (as usually) input for $t + \Delta t$ (e.g., prescribed displacement, acceleration etc.).

The fact that the Modified Wilson θ method executes for $t + \theta\Delta t$ also affects output/draw of results in structural material points. Within iterations (e.g., for monitors at iterations), they are printed/drawn for $t + \theta\Delta t$. After the iterations process has been completed, they are printed/drawn for $t + \Delta t$ as usual. Internal forces are always printed for $t + \theta\Delta t$ and the same for external forces.

As described above Modified Wilson θ method behaves in a bit nonstandard way. Particularly input for $\overline{R}^{t+\theta\Delta t}$ is unpractical. To alleviate these difficulties and inconvenience, Atena also offers Wilson θ method. Although it still solves the governing equations for time $t + \theta\Delta t$, it uses several extrapolations (e.g., $\overline{R}^{t+\theta\Delta t} = \overline{R}^{t} + \theta(\overline{R}^{t+\Delta t} - \overline{R}^{t})$, $\overline{F}^{t+\theta\Delta t} = \overline{F}^{t} + \theta(\overline{F}^{t+\Delta t} - \overline{F}^{t})$) so that it suffices with $\overline{R}^{t+\Delta t}$ and $\overline{F}^{t+\Delta t}$ only. Consequently, it inputs all boundary conditions and print/draw all result for $t + \Delta t$ akin to any other solution method for dynamic analysis. On the other hand, it is at price of accuracy because the extrapolation is linear, whereby the loading and internal forces are not!

Remind that for dynamic analysis, concentrated forces, element body/boundary load, etc., is input in the incremental form, and it is "cumulated" in the structure. The same applies to prescribed displacements.

Prescribed velocities, accelerations, etc., must be input as total load. MASS_ACCELERATION must also be input in total values (and in each step, it is also recalculated from scratch).

More details on the methods' convergency can be found in (Hughes 1983) and (Wood. 1990).

8.1 Structural Damping

As far as damping matrix $\mathbf{C}$ is concerned, Atena uses the well known proportional damping:

$$\mathbf{C} = \delta_M \mathbf{M} + \delta_K \mathbf{K} \tag{5.104}$$

where δ_M, δ_K are user-defined damping coefficients. These coefficients can be directly set as user input data, or they can be generated based on knowledge of modal damping parameters ξ. The parameters ξ are defined by

$$\underline{\phi_i}^T \mathbf{C} \underline{\phi_i} = \underline{\phi_i}^T (\delta_M \mathbf{M} + \delta_K \mathbf{K}) \underline{\phi_i} = 2\omega_i \xi_i \tag{5.105}$$

where:

$\underline{\phi_i}$ is *i*-th structural eigenvector,

ω_i is *i*-th structural eigenmode,

ξ_i is modal damping parameter associated with ω_i and $\underline{\phi_i}$.

Using the fundamental properties of eigenmodes $\underline{\phi_i}^T M \underline{\phi} = 1$, $\underline{\phi_i}^T K \underline{\phi} = \omega_i$, we can rewrite (5.105)

$$\delta_M + \delta_K \omega_i = 2\omega_i \xi_i \tag{5.106}$$

Equations (5.104) introduces 2 parameters for damping and, thus, if only 2 values of ξ_i are to be used, they are directly substituted in (5.105), (resp. (5.106)) and solved for from this set of equations.

However, in practice, structural damping is more complicated and some sort of compromise must be done. In this case, structural damping properties are typically measured for more eigenmodes, and optimal values of coefficients δ_M, δ_K are calculated by the least square method, i.e., we are seeking a minimum of the expression $\left(\delta_M + \delta_K \omega_i - 2\omega_i \xi_i\right)^2$. It yields the following set of equations

$$\begin{aligned} \delta_M \sum_i \frac{w_i^2}{\omega_i^2} + \delta_K \sum_i w_i^2 &= 2 \sum_i \frac{w_i^2 \xi_i}{\omega_i} \\ \delta_M \sum_i w_i^2 + \delta_K \sum_i w_i^2 \omega_i^2 &= 2 \sum_i w_i^2 \omega_i \xi_i \end{aligned} \tag{5.107}$$

which is used to calculate the required damping parameters δ_M, δ_K.

There exist other assumptions to account for structural damping; however, their use is typically significantly more complex and more costly in terms of both RAM and CPU.

8.2 Spectral analysis

A proper selection of the solution time increment *dt* is essential for each dynamic analysis. If it is too large, the computed results will suffer from unacceptable inaccuracies. We will probably miss some important peaks in the loading history, and the analysis as a whole may even diverge. On the other hand, the use of a too-small value of *dt* will yield an analysis that is pointlessly expensive in terms of execution time and its demands towards CPU/RAM resources. In addition, its postprocessing is more laborious and prone to errors.

The spectral analysis described in this section is designed to assist the engineer in setting suitable *dt.* The main idea of the procedure is based on approximation of the structural loading $f(t)$ by Fourier series $f_{FT}(t)$, i.e. $f(t) \doteq f_{FT}(t)$, refer e.g., to http://en.wikipedia.org/wiki/Fourier_series . Both $f(t), f_{FT}(t)$ have one independent variable, which is structural time t .

The function $f_{FT}(t)$ is assembled in the following form:

$$f_{FT}(t) = \frac{a_0}{2} + \sum_{n=1}^{N} a_n \sin\left(\frac{2\pi}{T} n t\right) + b_n \cos\left(\frac{2\pi}{T} n t\right) \tag{5.108}$$

where *N* denotes the number of harmonics used for the approximation, *n* is harmonic-*th* id and $\sin\left(\frac{2\pi}{T} n t\right)$ and $\cos\left(\frac{2\pi}{T} n t\right)$ are n-*th* approximation functions, (i.e., n-*th* harminics). Eqn. (5.108) is suitable for approximation of a function (e.g. $f(t)$) in interval $t \in <0...T>$. Its Fourier coefficients are calculated as follows, see http://stelweb.asu.cas.cz/~slechta/fourier/fourier.html , http://www.mathstools.com/section/main/fourier_series_calculator#.VCFKkhZIpKI:

$$\begin{aligned} a_0 &= \frac{2}{\pi}\int_0^T f(\xi)\, d\xi \\ a_n &= \frac{2}{\pi}\int_0^T f(\xi)\sin\left(\frac{2\pi}{T} n\xi\right) d\xi \\ b_n &= \frac{2}{\pi}\int_0^T f(\xi)\cos\left(\frac{2\pi}{T} n\xi\right) d\xi \end{aligned} \tag{5.109}$$

Now let us introduce a coefficient $c_n = \sqrt{a_n^2 + b_n^2}$ and create a spectrum diagram of the loading. For each harmonic from (5.108), plot a point, whose coordinates are $\left[\frac{2\pi}{T} n,\ c_n'\right]$. Such a point

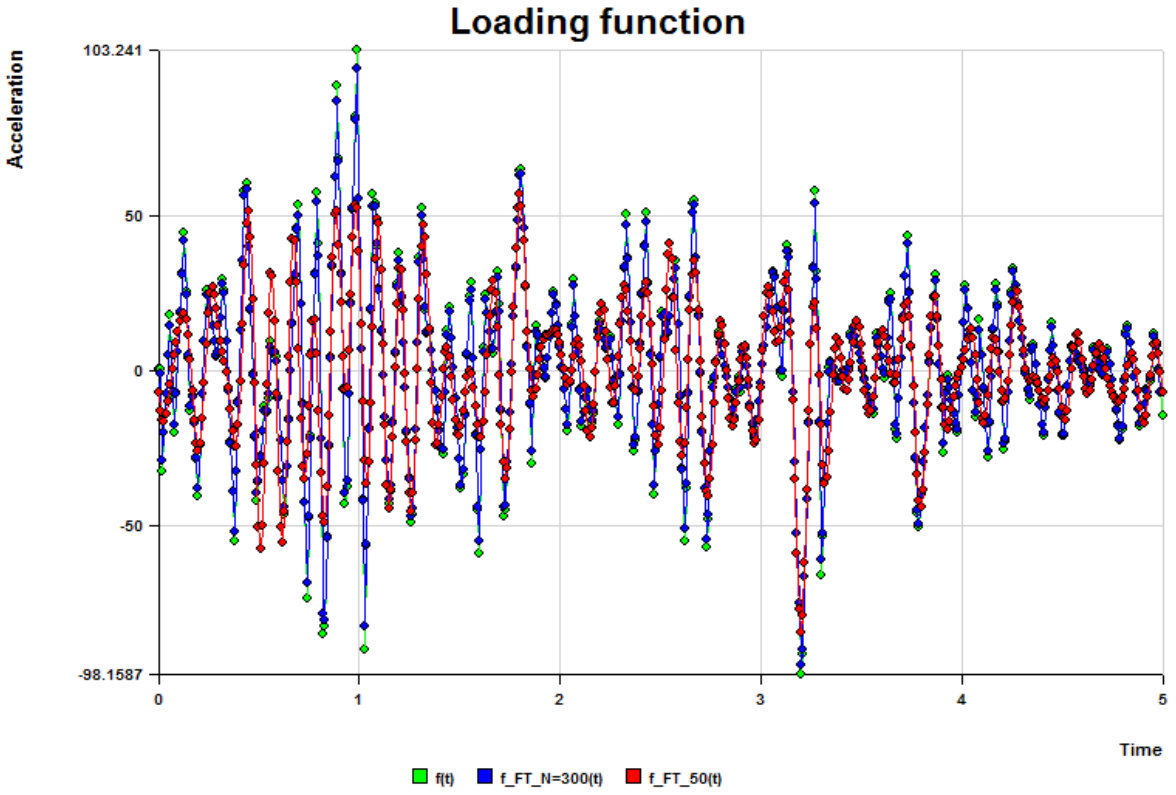
shows how much important is the n*th* harmonic (i.e., the harmonic with circular frequency $\frac{2\pi}{T}n$) for the loading function, i.e., how much it is excited by the load function $f(t)$.

The recommended solution time increment should be set so that the highest important harmonics are integrated in about 10 steps, i.e.,

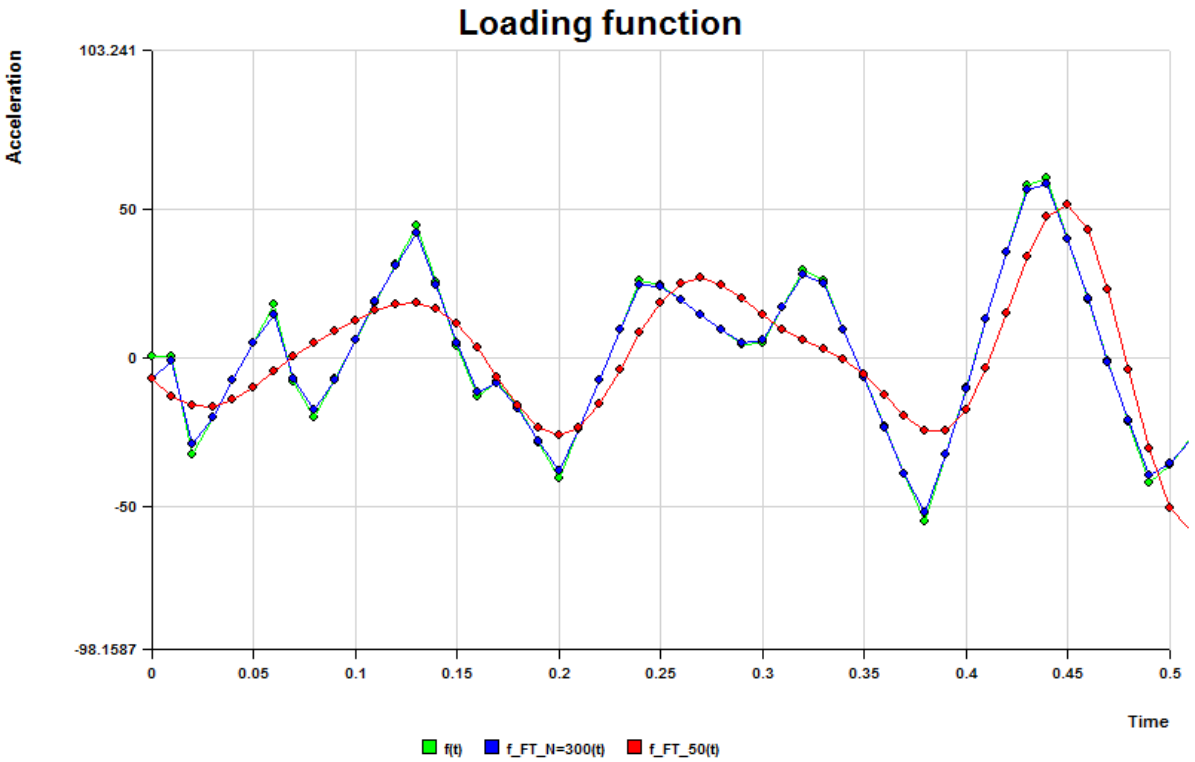
$$dt \le \min(n_{significant})\frac{T}{10} \tag{5.110}$$

By default, the FFT analysis uses a full modal spectrum, i.e., $n = 1..N$ in (5.108). However, the modal spectrum can be filtered, e.g. $n = n_1..m_1, n_2..m_2, ...n_k..m_k, ...n_L..m_1$. In this case, only values n from within the L intervals are used. This technique can be used to filter out some noise signals, etc.

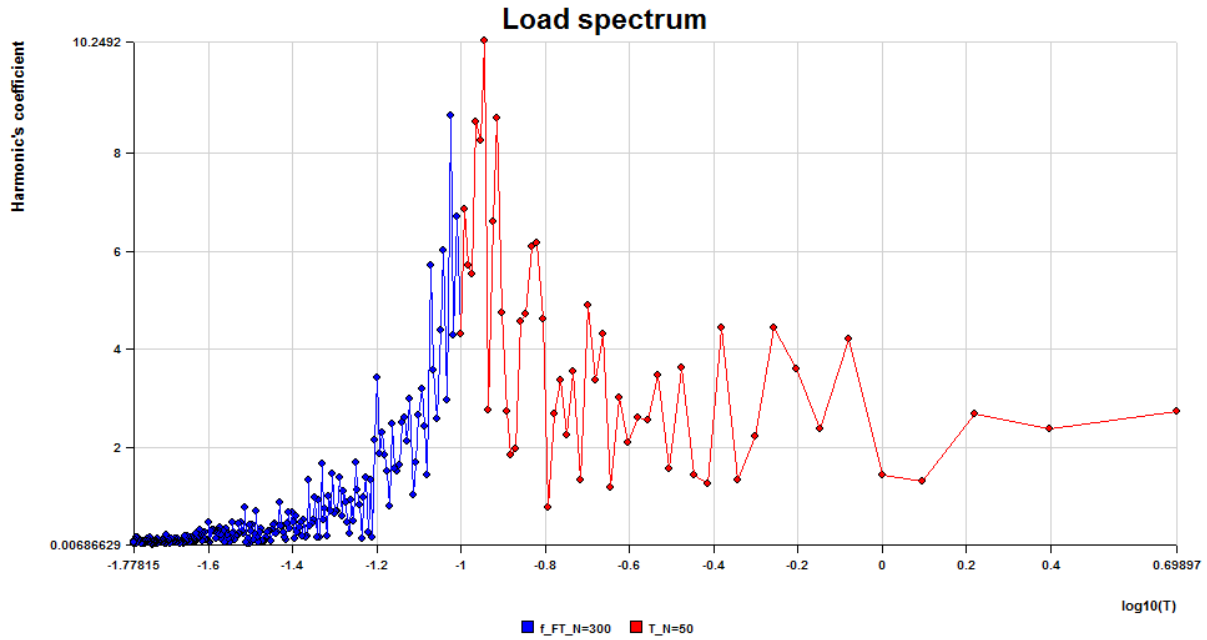
Let's take an example: Assume a simplified ElCentro accelerogram loading conditions, whose acceleration in time are depicted by the green line in the figure below:



Let's approximate this function by the Fourier series. In the first case, we use 300 harmonics, i.e., $N = 300$. The approximated accelerations are shown by the blue line, as seen in the figure above. In the second case, we use only 50 harmonics, and the corresponding approximation function is drawn by the red line. Plotting the functions in more detail, it can be seen that the approximation with $N = 300$ is fairly accurate whilst the approximation with $N = 50$ is rather crude, see the figure below. This conclusion is endorsed by the calculated average relative absolute error of the approximations. These are respectively 0.0314256 and 0.878789



The spectrum diagram below shows the contribution of individual approximation harmonics. It detects what harmonics are or are not important. Looking at the diagram, we see that the highest important harmonics is the one with $\log 10(T_n) = -1$, i.e. $T_{\min} = 0.1$. Therefore, the recommended solution time increment is $dt \le \frac{T_{\min}}{10} = 0.01$. This dt should ensure reasonably accurate results from dynamic analysis of a structure that is loaded by the investigated accelerogram.



The described spectrum analysis is fully supported by Atena (incl. all the plots). Its use is simple as it requires only a few input commands. For more details, please refer to the examples of commands for the input of a multilinear function (in the Atena input file documentation).

9 EIGENVALUES AND EIGENVECTORS ANALYSIS

This section describes methods used by ATENA software to calculate structural eigenvalues and eigenvectors. Good knowledge of eigenmodes of a structure is, in many cases, essential for understanding its behavior and selection of a method for its further analysis. It applies to statics and particularly to dynamic analyses, in which case it helps to choose a proper time increment in subsequent loading steps. It also helps in avoiding or reducing oscillation of the structure.

9.1 Inverse Subspace Iteration

Currently, ATENA uses the Inverse subspace iteration method to compute the eigenvalues and eigenvectors. The method is in detail described in (Bathe 1982), and hence, only its main features are presented here. The current implementation can be used only for symmetric matrices. The same applies to Jacobi and Rayleigh-Ritz methods that are mentioned later in this section.

It consists of three methods; each of them is capable of solving the eigenvalue problem on its own. However, if they are used simultaneously, they yield a very efficient scheme for solving eigenvalues and eigenvectors of large sparse structural systems. The significant advantage of this approach is that it is possible to search for a selected number of the lowest eigenmodes only. The lowest eigenmodes are typically the most important for the behavior of the structure because they represent the highest energy that the structure can absorb. On the other hand, the highest eigenmode is of low importance, can be neglected, and thereby save a lot of CPU time and other computational resources.

The Inverse subspace iteration consists of

- Inverse iteration method
- Rayleigh-Ritz method
- Jacobi method

It solves general eigenvalues and eigenvector problem of the following form:

$$\mathbf{K}\bar{u} = \omega^2 \mathbf{M}\bar{u} \tag{8.1}$$

where

$\mathbf{K}, \mathbf{M}$ is stiffness and mass matrix of structure,

$\bar{u}$ is the vector of eigenvector's nodal displacements,

ω is circular eigenfrequency

We are looking for a non-trivial solution, so that we solve for ω^2 that comes from

$$\det(\mathbf{K} - \omega^2 \mathbf{M}) = 0 \tag{8.2}$$

9.1.1 Rayleigh-Ritz Method

This method is used to transform the original eigenproblem of dimension *n* into an associated eigenproblem of dimension *m*<<*n*. The solution is to search in space $V_m << V_n$. Let vectors $\overline{\psi}_k$ constitute linearly independent bases in V_n. An eigenvector $\overline{u}_i$ is computed as a linear combination $\overline{c}_i$ of the base vectors ψ_k , i.e.

$$\overline{u}_i = \mathbf{\Psi}\overline{c}_i \tag{8.3}$$

where

$\mathbf{\Psi}$ is the matrix of base vectors $\overline{\psi}_k, k = 1..m$,

$\overline{c}_i$ is the vector of coefficients of the linear combination.

Rayleigh quotient is defined as

$$\rho(\overline{u}_i) = \frac{\overline{u}_i^T \mathbf{K} \overline{u}_i}{\overline{u}_i^T \mathbf{M} \overline{u}_i} \tag{8.4}$$

It can be proved that $\rho(\overline{u}_i)$ converges from the upper side to the corresponding circular frequency ω_i^2 . The condition of a minimum of $\rho(\overline{u}_i)$ yields:

$$\frac{\partial \rho(\overline{u}_i)}{\partial \overline{c}_{i,k}} = 0, \; k = 1..m \tag{8.5}$$

where $\overline{c}_{i,k}$ is k component of the vector $\overline{c}_i$

If we introduce

$$\mathbf{A} = \mathbf{\Psi}^T \mathbf{K} \mathbf{\Psi}, \qquad \mathbf{B} = \mathbf{\Psi}^T \mathbf{M} \mathbf{\Psi} \tag{8.6}$$

the condition (8.5), after substituting (8.6), results in

$$\mathbf{A}\overline{c}_i = \omega_i^2 \mathbf{B}\overline{c}_i \tag{8.7}$$

which is an equation of eigenproblem of matrices **A**,**B**. This problem has dimension *m*, which is significantly smaller than the original dimension *n.*

9.1.2 Jacobi Method

Jacobi method is used for the solution of full symmetric eigensystems of lower dimension. In the frame of the Inverse subspace iteration method, it is used to solve (8.7). (Note, however, that that the eigenproblem (8.7) can be used by any other method).

The Jacobi method is based on the property that if we have a matrix **A**, an orthogonal matrix **C,** and a diagonal matrix **D**, whereby

$$\mathbf{C}^T \mathbf{A} \mathbf{C} = \mathbf{D} \tag{8.8}$$

then the matrices **A** and **D** have identical eigenvalues, and they are diagonal elements of the matrix **D.** The transformation matrix **C** is calculated in an iterative manner

$$\mathbf{C} = \mathbf{S}_1 \mathbf{S}_2 ..... \mathbf{S}_k, k = 1..\infty \tag{8.9}$$

where the individual $\mathbf{S}_k$ has the following form

$$S_k = \begin{bmatrix} 1 & & & 0 & & 0 & 0 \\ & 1 & & & & & \\ & & 1 & & & & \\ 0 & & & \cos(\alpha) & 0 & -\sin(\alpha) & 0 \\ & & & 0 & 1 & 0 & \\ 0 & & & \sin(\alpha) & & \cos(\alpha) & 0 \\ 0 & & & 0 & & 0 & 1 \end{bmatrix} \tag{8.10}$$

The entries $\cos(\alpha), \pm\sin(\alpha)$ are put in *i,j* rows and columns, and they are constructed in the way that they will zeroize a_{ij} after the transformation. The other diagonal elements are equal to 1 and the remaining off-diagonal elements are 0.

In the case of a general eigenproblem, the whole procedure of constructing $\mathbf{S}_k$ is very similar. The matrices $\mathbf{S}_k$ now adopt the shape

$$S_k = \begin{bmatrix} 1 & & & 0 & & 0 & 0 \\ & 1 & & & & & \\ & & 1 & & & & \\ 0 & & & 1 & 0 & a & 0 \\ & & & 0 & 1 & 0 & \\ 0 & & & b & & 1 & 0 \\ 0 & & & 0 & & 0 & 1 \end{bmatrix} \tag{8.11}$$

Notice that the matrix $\mathbf{S}_k$ is not orthogonal anymore. The two variables *a,b* are calculated to zeroize off-diagonal elements *i,j* of both matrices **K** and **M.** Eigenmodes of the problem are then calculated as

$$\omega_i^2 = \frac{a_{ii}^{'}}{b_{ii}^{'}} \tag{8.12}$$

where $a_{ii}^{'}, b_{ii}^{'}$ are diagonal elements of transformed (and diagonalized) matrices **A, B**.

Eigenvectors of the problem are columns of the transformation matrix **C.**

9.1.3 Inverse Iteration Method

Inverse iteration method is carried out as follows: Starting with an initial transformation of eigenvector $\overline{u}_{i,1}$, we calculate a vector of corresponding inertia forces (step 1)

$$\overline{f}_{i,1} = \mathbf{M}\,\overline{u}_{i,1} \tag{8.13}$$

Knowing $\overline{f}_{i,1}$, we can compute a new approximation of $\overline{u}_i$, (step 2)

$$\overline{u}_{i,2} = K^{-1}\overline{f}_{i,1} \tag{8.14}$$

and repeat the step 1. Hence, for iteration *k* we have

$$\begin{aligned} \overline{f}_{i,k} &= \mathbf{M}\,\overline{u}_{i,k} \\ \overline{u}_{i,k+1} &= K^{-1}\overline{f}_{i,k} \end{aligned} \tag{8.15}$$

and the iterating is stop, when $\overline{u}_{i,k+1} \approx \overline{u}_{i,k}$. The above-described algorithm tends to converge to the lowest eigenmodes. If any of these are to be skipped, the initial eigenvector $\overline{u}_{i,1}$ must be orthogonal to the corresponding eigenvectors. In practice, the vector $\overline{u}_{i,k}$ must be orthogonalized with respect to the skipped eigenvectors even during the iterating procedure, as the initial orthogonality may get (due to some round-off errors) lost.

9.1.4 Algorithm of Inverse Subspace Iteration

Having briefly described the above three methods, we can now proceed to the actual solution algorithm of the Inverse subspace iteration method itself:

Step 1 - Inverse iteration method:

$$\mathbf{K}\tilde{\mathbf{U}}_{k+1} = \mathbf{M}\mathbf{U}_k$$

Step 2 - Raylegh quotient method:

$$\begin{aligned} \mathbf{A}_{k+1} &= \tilde{\mathbf{U}}_{k+1}^T \mathbf{K}\tilde{\mathbf{U}}_{k+1} \\ \mathbf{B}_{k+1} &= \tilde{\mathbf{U}}_{k+1}^T \mathbf{M}\tilde{\mathbf{U}}_{k+1} \end{aligned}$$

Step 3 – Jacobi method:

$$\mathbf{A}_{k+1}\mathbf{C}_{k+1} = \mathbf{B}_{k+1}\mathbf{C}_{k+1}\boldsymbol{\Delta}^2$$

Step 4 - Correct the eigenvectors:

$$\mathbf{U}_{k+1}^T = \tilde{\mathbf{U}}_{k+1}^T\,\mathbf{C}_{k+1}$$

(8.16)

In the above

m is the number of projection eigenmodes (reasonably higher than the number of required eigenmodes),

$\mathbf{U}_k$ is the matrix of columnwise arranged *m* eigenvectors after *k*- th iteration,

$\mathbf{A}_{k+1}, \mathbf{B}_{k+1}$ are transformed stiffness and mass matrices of the problem, (having dimension $m<<n$),

$\mathbf{C}_{k+1}$ is the matrix of eigenvectors of $\mathbf{A}_{k+1}, \mathbf{B}_{k+1}$, see (8.9)

$\mathbf{\Delta^2}$ is a matrix with eigenmodes (on its diagonal). Notice that eigenmodes for transformed and the original eigenmode problem are the same.

The steps 1 thru 4 are repeated until the difference between the two subsequent operations is negligible.

The solution algorithm (8.16) is in ATENA a bit modified in order to reduce CPU time and RAM resources and is described below:

Step 1 - Inverse iteration method:

$$\hat{\mathbf{U}}_{k+1} = \mathbf{M}\mathbf{U}_k$$

$$\mathbf{K}\tilde{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_{k+1}$$

Step 2 - Raylegh quotient method:

$$\mathbf{A}_{k+1} = \tilde{\mathbf{U}}_{k+1}^T \mathbf{K}\tilde{\mathbf{U}}_{k+1} = \tilde{\mathbf{U}}_{k+1}^T \hat{\mathbf{U}}_{k+1}$$

$$\hat{\mathbf{U}}_{k+1} = \mathbf{M}\tilde{\mathbf{U}}_{k+1}$$

$$\mathbf{B}_{k+1} = \tilde{\mathbf{U}}_{k+1}^T \mathbf{M}\tilde{\mathbf{U}}_{k+1} = \tilde{\mathbf{U}}_{k+1}^T \hat{\mathbf{U}}_{k+1}$$

Step 3 – Jacobi method:

$$\mathbf{A}_{k+1}\mathbf{C}_{k+1} = \mathbf{B}_{k+1}\mathbf{C}_{k+1}\mathbf{\Delta^2}$$

Step 4 - Correct the eigenvectors:

$$\mathbf{U}_{k+1}^T = \tilde{\mathbf{U}}_{k+1}^T \mathbf{C}_{k+1} \tag{8.17}$$

The advantage of this procedure over the one defined in (8.16) is that now you don't need to store the original and factorized form of the matrix **K. O**nly the factorized form is needed during the iterations.

A special issue in this method is how to set up the initial vectors $\mathbf{U}_1$. This is what we do in ATENA. The first vector contains the diagonal elements of **M**. The next vectors are constructed in the way that they have zeros everywhere except one entry. This entry corresponds to maximum $\frac{m_{ii}}{k_{ii}}$ and is set to 1.

The procedure as it is (because of the Inverse iteration method) cannot solve for zero eigenmodes. This may be a problem, especially if we want to analyze structural rigid body motions or spurious energy modes. If this is the case, shift matrix **K** by an arbitrary value λ_s, i.e., solve the associated eigenproblem

$$(\mathbf{K} - \lambda_s \mathbf{M})\overline{u}_s = \omega_s^2 \mathbf{M}\overline{u}_s \tag{8.18}$$

The original eigenvalues and eigenvectors are then calculated by

$$\begin{aligned} \overline{u} &= \overline{u}_s \\ \omega^2 &= \omega_s^2 - \lambda_s \end{aligned} \tag{8.19}$$

Another problem of Inverse subspace iteration is to compute multiple eigenvectors. Unfortenatly, it is not that rare case and it happens, e.g., if the structure has an axis of symmetry. The occurrence of multiple eigenmodes in the structure may yield non-orthogonal eigenvectors, and thus, some eigenmodes can be missed. There are some techniques for resolve this problem (Jendele 1987); however, they have not been implemented in ATENA yet. Good news is that in reality, no eigenmodes are usually quite identical due to some round-off errors. The case of multiple structural eigenmodes thus typically causes only some worsening of accuracy and no eigenmode gets missed.

Nevertheless, if we want to be sure that no eigenmode was missed, we can assess it by Sturm sequence property test.

9.1.5 Sturm Sequence Property Check

This property says (Bathe 1982) that if we have an eigenproblem (8.1), perform a shift λ_s and factorize that matrix (i.e., **D** is a diagonal matrix, **L** is a lower triangular matrix),

$$\mathbf{K} - \lambda_s \mathbf{M} = \mathbf{L}\,\mathbf{D}\,\mathbf{L}^T \tag{8.20}$$

then the number of negative diagonal elements in **D** equal to the number of eigenvalues smaller than the shift λ. This way, we can simply test, whether we missed an eigenvalue with the calculated set of *m* eigenmodes or not

There are other methods that can be used to compute eigenvalues and eigenvectors of large sparse eigensystems. Particularly popular is e.g., Lanczosh method (Bathe 1982). There exist also several enhancements for the present Inverse subspace iteration method. For instance, using a shifting technique may significantly improve the convergency of the method (especially if some eigenvalues are close to each other).

These improved techniques may be implemented in the future. In any case, the current ATENA implementation of eigenmodes analysis proves to solve the eigenmodes problem in most cases quite successfully.

9.2 References

BATHE, K. J. (1982). Finite Element Procedures in Engineering Analysis. Englewood Cliffs, New Jersey 07632, Prentice Hall, Inc.

JENDELE, L. (1987). The Orthogonalization of Multiple Eigenvectors in Subspace Iteration Method. IKM - XI. Internationaler Kongress ueber Anwendungen der Mathematik in der Ingenieurwissenschaften, Weimar.

WOOD., W. L. (1990). Practical-Time Stepping Schemes. Oxford, Clarenton Press.

10 GENERAL FORM OF DIRICHLET BOUNDARY CONDITIONS

A unique feature of ATENA software is the way in which it implements Dirichlet boundary conditions. It supports to constraint any degree of freedom (DOF) by a linear of any number of other structural DOFs. The proposed method of applying and processing the boundary conditions is computationally efficient and memory economical because all constraint degrees of freedoms (DOFs) are eliminated already during assembly of structural global stiffness matrix and load vectors. The adopted concept has a wide range of use, and several of its possibilities are discussed. At the end of the Section, a few samples are given.

10.1 Theory Behind the Implementation

A crucial part of a typical finite element analysis (whether linear or nonlinear) is the solution of a set of linear algebraic equations in the following form

$$\sum_{j=1}^{n} K_{ij} u_j = r_i \ , \ i = 1..n \tag{9.1}$$

where K_{ij} is an element i, j of a predictor matrix **K**, (i.e., usually structural stiffness matrix), r_i is an external force (or unbalanced force), applied into *i*-th structural degree of freedom (DOF), and finally u_i is displacement (or displacement increment) at the same DOF. Such a set of equations is always accompanied by many boundary conditions (BCs). They can be one of the following:

Von-Neumann boundary conditions, (also called right-hand side (RHS) BCs). Number and type of these BCs have no impact on dimension *n* of the problem (9.1). They are accumulated in the vector $\overline{r}$. This vector is assembled on the per-node basis for concentrated nodal forces and/or per-element basis for nodal forces being equivalent to element loads.

The second type of boundary conditions are Dirichlet boundary conditions (also called left-hand side (LHS) BCs). ATENA implementation of this type of BCs is now described. A simple form of such BCs reads

$$\begin{aligned} u_l &= 0, \ l \in <1, n> \\ u_l &= u_{l0}, \ l \in <1, n> \end{aligned} \tag{9.2}$$

These kinds of BCs typically represent structural supports with no displacements (the first equation) or with prescribed displacements u_{l0}, (the second equation). Although most LHS BCs are of the above form (and only a few finite element packages offer anything better), there are cases when a more general LHS BC is required. Therefore, ATENA software provides a solution for implementing a form of Dirichlet BCs, where each degree of structural freedom can be a linear combination of any other degrees of freedom. Mathematically, this is expressed by

$$u_l = u_{l0} + \sum_{k \in <1,n>} \alpha_{lk} u_k, \ l \in <1, n> \tag{9.3}$$

There are many cases in which the above form of Dirichlet conditions proves helpful. Some examples are discussed later in the Chapter. The important point about implementing Equations (9.3) is that they are utilized already during the assembling of the problem (9.1). It means that if we have *m* of these BCs, then the final dimension of the matrix **K** becomes only $(n-m)$. This fact significantly reduces requirements for computer storage.

In the following, we shall call such boundary conditions as "Complex Boundary Conditions", or CBCs, (see also ATENA Input file manual, where the same name is used).

10.1.1 Single CBC

The procedure of implementing Dirichlet BCs of the form (9.3) is now presented. Let us start with just one BC equation (9.4). It says that u_l equals to a constant prescribed displacement u_{l0} plus α_{lk} multiple of a displacement u_k.

$$u_l = u_{l0} + \alpha_{lk} u_k \tag{9.4}$$

Substituting (9.4) into the Equation (9.1) yields

$$\sum_{j=1, j\neq l}^{n} K_{ij}\, u_j + K_{il} u_l = \sum_{j=1, j\neq l}^{n} K_{ij}\, u_j + K_{il}(u_{l0} + \alpha_{lk} u_k) = r_i \ , \ i = 1..n \tag{9.5}$$

which after some manipulation can be simplified into the form

$$\sum_{j=1}^{n} \left(K_{ij} + K_{il}\alpha_{lk}\delta_{kj} \right) u_j = r_i - K_{il} u_{l0} \ , \ i = 1..n \tag{9.6}$$

The above set of equations could be already used to solve for the unknown displacements (or displacement increments) u_j. δ_{kj} stands for $k.j$ Kronecker delta tensor. The trouble is, however, that even though the matrix **K** might be symmetric, the set of equations (9.6) is not symmetric anymore. Thus, to preserve the symmetry, add an α_{lk} multiple of the row *l*, i.e.,

$$\alpha_{lk} \left(\sum_{j=1}^{n} \left(K_{lj} + K_{ll}\alpha_{lk}\delta_{kj} \right) u_j \right) = \alpha_{lk} \left(r_l - K_{ll} u_{l0} \right) \tag{9.7}$$

to the row *k*, i.e.,

$$\sum_{j=1}^{n} \left(K_{kj} + K_{kl}\alpha_{lk}\delta_{kj} \right) u_j = r_k - K_{kl} u_{l0} \tag{9.8}$$

This results in the row *k* getting the form

$$\begin{aligned} &\sum_{j=1}^{n} \left(K_{kj} + K_{kl}\alpha_{lk}\delta_{kj} + \alpha_{lk} \left(K_{lj} + K_{ll}\alpha_{lk}\delta_{kj} \right) \right) u_j = \\ &\sum_{j=1}^{n} \left(K_{kj} + \alpha_{lk} K_{lj} + \left(K_{kl}\alpha_{lk} + \alpha_{lk}\alpha_{lk} K_{ll} \right) \delta_{kj} \right) u_j = \\ &r_k - K_{kl} u_{l0} + \alpha_{lk} \left(r_l - \alpha_{lk} K_{ll} u_{l0} \right) \end{aligned} \tag{9.9}$$

Hence, the final form of the governing set of equations will read

$$\sum_{j=1}^{n}\left(K_{ij}+K_{il}\alpha_{lk}\delta_{kj}+\delta_{ik}\alpha_{lk}K_{lj}+\delta_{ik}\delta_{kj}\alpha_{lk}^{2}K_{ll}\right)u_j = \\ r_i-K_{il}u_{l0}+\delta_{ik}\alpha_{lk}\left(r_l-K_{ll}u_{l0}\right) \tag{9.10}$$

The above equations can be written as

$$\sum_{j=1}^{n}\tilde{K}_{ij}\,u_j=\tilde{r}_i\ ,\ i=1..n \tag{9.11}$$

where

$$\tilde{K}=$$

$$\begin{bmatrix} K_{11} & \dots & K_{1i} & \dots & K_{1k}+K_{1l}\alpha_{lk} & \dots & K_{1j} & \dots & K_{1n} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ K_{i1} & \dots & K_{ii} & \dots & K_{ik}+K_{il}\alpha_{lk} & \dots & K_{ij} & \dots & K_{in} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ K_{k1}+\alpha_{lk}K_{l1} & \dots & K_{ki}+\alpha_{lk}K_{li} & \dots & K_{kk}+2K_{kl}\alpha_{lk}+\delta_{kk}\delta_{kk}\alpha_{lk}^{2}K_{ll} & \dots & K_{kj}+\alpha_{lk}K_{lj} & \dots & K_{kn}+\alpha_{lk}K_{ln} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ K_{j1} & \dots & K_{ji} & \dots & K_{jk}+K_{jl}\alpha_{lk} & \dots & K_{jj} & \dots & K_{jn} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ K_{n1} & \dots & K_{ni} & \dots & K_{nk}+K_{nl}\alpha_{lk} & \dots & K_{nj} & \dots & K_{nn} \end{bmatrix} \tag{9.12}$$

$$\tilde{r}=$$

$$\begin{bmatrix} r_1-K_{1l}u_{l0} \\ \dots \\ r_i-K_{il}u_{l0} \\ \dots \\ r_k-K_{kl}u_{l0}+\alpha_{lk}\left(r_l-K_{ll}u_{l0}\right) \\ \dots \\ r_j-K_{jl}u_{l0} \\ \dots \\ r_n-K_{nl}u_{l0} \end{bmatrix} \tag{9.13}$$

Providing the original matrix **K** is symmetric, i.e. $K_{ij}=K_{ji}$, then the matrix $\tilde{\mathbf{K}}$ is now also symmetric, i.e. $\tilde{K}_{ij}=\tilde{K}_{ji}$.

The displacement u_l constrained by Equation (9.4) has a constant part u_{l0} and a variable part $\alpha_{lk}\,u_k$, in which u_l depends only on a single u_k. A more general form of this BC would be if u_l depends on more displacements. It corresponds to the following form of the boundary condition:

$$u_l = u_{l0} + \sum_k \alpha_{lk} u_k \tag{9.14}$$

In this case, the displacement u_l is calculated as a constant part u_{l0} plus a linear combination α_{lk} of displacements u_k. k can be any displacement, i.e. $k \in <1..n>$. Replacing BC defined by Equation (9.4) by the above Equation (9.14), the equation will change to the form

$$\begin{aligned} &\sum_{j=1}^{n} \left(K_{ij} + \sum_{k,k\neq l} \left(K_{il}\alpha_{lk}\delta_{kj} + \delta_{ik}\alpha_{lk}K_{lj} + \delta_{ik}\delta_{kj}\alpha_{lk}^2 K_{ll} \right) \right) u_j = \\ &r_i - K_{il}u_{l0} + \sum_{k,k\neq l} \left(\delta_{ik}\alpha_{lk} \left(r_l - K_{ll}u_{l0} \right) \right) \end{aligned} \tag{9.15}$$

10.1.2 Multiple CBCs

The previous paragraph derived all the necessary relations for implementing a single boundary condition. Now we will proceed to the case of multiple boundary conditions. Each particular BC is again written in the form (9.14).

$$u_l = u_{l0} + \sum_k \alpha_{lk} u_k, \quad l \in <1, n>, l = \{l_1, l_2, ...l_r\} \tag{9.16}$$

The problem is, however, that displacements u_k in (9.16) need not be free but fixed by another BC, *k* can also run through *l*, (resulting in a recursive formulation), more BCs can be specified for the same u_l, a particular BC can be specified more times and in more forms etc. For example, we may have a set of boundary equations that contains BCs

$$u_1 = u_2, \quad u_2 = u_1 \tag{9.17}$$

or it can contain

$$u_1 = u_2, \quad u_2 = u_1, \quad u_1 = 0.003 \tag{9.18}$$

Both of these are correct. Unfortunately, the set can also contain

$$u_1 = u_2, \quad u_2 = -0.003, \quad u_2 = u_1, \quad u_1 = 0.003 \tag{9.19}$$

which is definitely wrong. Therefore, before any use of such set of BCs it is necessary to detect and fix all redundant and contradictory multiple BCs present in it. It is easily done in case of a simple set of BCs like the one above, but in real analyses with thousands of BCs in the form (9.16), (some of them quite complex, i.e., *k* runs through many DOFs) the only way to proceed is to treat (9.16) as a set of equations to be solved prior their use in (9.13). Redundant BCs are ignored, and contradictory BCs are fulfilled after their summation. Let us suppose that all structural constraints are specified in the set of equation (9.16). This can be written in matrix form

$$\overline{u}_l = \overline{u}_{l0} + \mathbf{A}\overline{u}_k \tag{9.20}$$

The above relationship represents a system of algebraic linear equations. The system is typically non-symmetric, sparse and has a different number of rows (i.e., the number of BCs) and columns, (i.e., the number of master and slave DOFs). Moreover, it is often ill-

conditioned, with a number of equations being linear combinations of the others, e.g., see the example in (9.17). In the beginning, it is often not known which DOF is dependent, (i.e., slave) and which is independent, (i.e., master), (e.g., see also (9.17)).

Based on the above properties, the following procedure has been developed to solve the problem (9.20):

1. Allocate "columns" for all slave and master DOFs, i.e., loop through all BCs in (9.16) and allocate DOFs for both slave (i.e., LHS) and master (i.e., RHS) displacements u_i.
2. Allocate storage for the matrix $\mathbf{A}$ and vectors $\overline{u}_l, \overline{u}_{l0}$ in (9.20). The matrix has l_r the number of rows (see (9.16)) and l_c the number of columns. l_c is the dimension of the DOFs map created in the point add. 1.
3. Assemble the matrix $\mathbf{A}$ and the vectors $\overline{u}_l, \overline{u}_{l0}$.
4. Detect constant BCs, i.e., $u_l = u_{l0}$ and swap rows of **A** so that the rows corresponding to constant BCs are pushed to the bottom.
5. Detect constant fixed DOFs, i.e., those with $\alpha_{lk} = 0$ and variable fixed DOFs, i.e., that are those dependent on other (master) DOFs and having $\alpha_{lk} \neq 0$.
6. Swap columns of $\mathbf{A}$, so that the former DOFs are pushed to the right and the latter DOFs to the left. The operations described at the point 5 and 6 are needed to assure order, in which the constrained DOFs are eliminated. This is important for later calculation of the structural reactions.
7. Using the Gauss method to triangulate the set of BC equations. The triangulation is carried out in the standard way with the following differences.
    a. Before eliminating entries of **A** located in column below a_{kk}, check for a non-zero entry in the row *k.* If all its entries are zero, then ignore this row and proceed to the next one. (It is the case of multiple BCs having the same content).
    b. Check, whether the row *k* specifies BC for constant or variable DOF, (see explanation in the point 5 above). In the former case push the row *k* to the bottom and proceed to the next row.
    c. Swap columns $< k...l_c >$ so that $abs(a_{kk})$ becomes maximum.
    d. If $a_{kk} = 0$, swap lines $< k...l_r >$ to find a non-zero entry in a_{kk}. Thereafter, swap columns $< k...l_c >$ to find maximum a_{kk}.
    e. Eliminate entries below a_{kk} as usually.

As it was already mentioned, the matrix **A** is typically very sparse. Hence, a special storage schemes are used that stores only non-zero entries of **A.** The data are stored by rows. Each row has a number of data series, i.e., sequences or chunks of consecutive non-zero data (within the row). The data are in a three-dimensional container. For each such chunk of data, we also need to store its first position and length. This is done in two two-dimensional containers.

As an example, suppose that we have the following matrix **A:**

$$\mathbf{A} = \begin{bmatrix} a_{11} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_{22} & a_{23} & 0 & 0 & a_{26} & a_{27} \\ 0 & 0 & a_{33} & 0 & 0 & 0 & 0 \\ 0 & a_{42} & 0 & a_{44} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{55} & a_{56} & a_{57} \\ 0 & a_{62} & 0 & 0 & 0 & a_{66} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{77} \end{bmatrix} \tag{9.21}$$

It is stored as follows (*A.data* stores the actual data, *A.rowbase* stores base indices for non-zero entries in rows, *A.rowlength* contains dimension of non-zero data chunks; all arranged by rows):

$$\begin{aligned}
&A.data(1)(1)(1) = a_{11} \\
&A.data(2)(1)(1) = a_{22}, A.data(2)(1)(2) = a_{23}, A.data(2)(2)(1) = a_{26}, A.data(2)(2)(2) = a_{27} \\
&A.data(3)(1)(1) = a_{33}, \\
&A.data(4)(1)(1) = a_{42}, A.data(4)(2)(1) = a_{44} \\
&\text{.....} \\
&A.rowbase(1)(1) = 1 \\
&A.rowbase(2)(1) = 2, A.rowbase(2)(2) = 6 \\
&A.rowbase(3)(1) = 3 \\
&A.rowbase(4)(1) = 2, A.rowbase(4)(2) = 4 \\
&\text{.....} \\
&A.rowlength(1)(1) = 1 \\
&A.rowlength(2)(1) = 2, A.rowlength(2)(2) = 2 \\
&A.rowlength(3)(1) = 1 \\
&A.rowlength(4)(1) = 1, A.rowlength(4)(2) = 1
\end{aligned} \tag{9.22}$$

A number of optimisation techniques are used to speed up the process of triangularization of the matrix **A.** These are summarized below:

The data are stored by rows and the elimination is also carried out by rows. (Row-based storage is also more convenient during assembling the **A** from (9.16)). All the operations needed for the elimination are carried out only for nonzero data. Their horizontal position is stored in *A.rowbase* and *A.rowlength*, hence it is no problem to skip all zero entries. A typical total number of columns l_c, see (9.16), is of order from thousands to hundred thousands DOFs. On the other hand *a.rowlength* is on average only of order of tens. This is where the CPU savings comes from.

By the way, the same mapping of non-zero entries is also used for columns. This is achieved by additional arrays *A.columnbase* and *A.columnlength* that are also included in the storage scheme *A*. (Their construction is similar to *A.rowbase* and *A.rowlength*; instead by rows

they are arranged by columns). These two additional arrays make possible to skip all zero entries during column-base operations. The resulting significant increase of triangularization speed pays off for a small amount of an extra RAM that is needed to store $A.columnbase$ and $A.columnlength$.

The adopted procedure of triangularization many times swaps lines and/or columns of **A**. In view of the adopted storage scheme, it can be a quite expensive procedure. To alleviate this problem, the storage scheme includes four additional arrays, namely $A.rowindex$, $A.rowinverseindex$, $A.columnindex$ and $A.rowinverseindex$. In the beginning, $A.rowindex(i) = i$ and similarly $A.rowinverseindex(i) = i,\ i = 1...l_c$. When a row r_1 should be swapped with a row r_2, the data in $A.data$ remains unchanged and we swap only corresponding row indices in $A.rowindex$, (and accordingly also entries in the array for inverse mapping $A.rowinverseindex$). The same strategy is used for swapping the columns. As a result, any swapping operation does not require any moving of actual data (except of swapping corresponding indices for mapping the rows and columns) and thus it is extremely fast. On the other hand, in order to access a_{ij} we must use $a_{i'j'}$, where $i' = rowindex(i)$ and $j' = columnindex(j)$. The incurred CPU overhead is well acceptable, because the matrix **A** is very sparse.

10.2 Application of Complex Boundary Conditions

This section presents several examples where the developed Dirichlet boundary conditions are advantageously used. In each case, the corresponding finite element model exploits the general form of BC defined by Equation (9.16).

10.2.1 Finite Element Mesh Refinement

Suppose we need to refine a mesh as shown in Fig. 10-1. The mesh should refine from 5 elements per row to 10 elements per row. The figure depicts three possible techniques to achieve the goal.

In the case A, the fine and coarse parts of the mesh (consisting of quadrilateral elements) are connected by a row of triangular elements. This way of mesh refinement is used the most often. However, mixing quadrilateral and triangular elements is not always the best solution.

In the case B, the refinement is achieved by using hierarchical finite elements, see (Bathe 1982). The coarse mesh near the interface employs five nodes hierarchical elements. This refinement is superior to the others; however, it requires special finite elements and special mesh generator; both of these rarely available in a typical finite element package.

In the case C, the fine and coarse parts of the mesh are generated independently. After the generation of all nodes and elements, the interface nodes are connected by complex boundary conditions. For example, we can use $u_i = u_m, u_k = u_n, u_j = 0.5u_m + 0.5u_n$. The main advantage of this approach is that it is simple for both finite element pre/postprocessor and finite element modeler (namely its finite element library). Hence it is preferable!

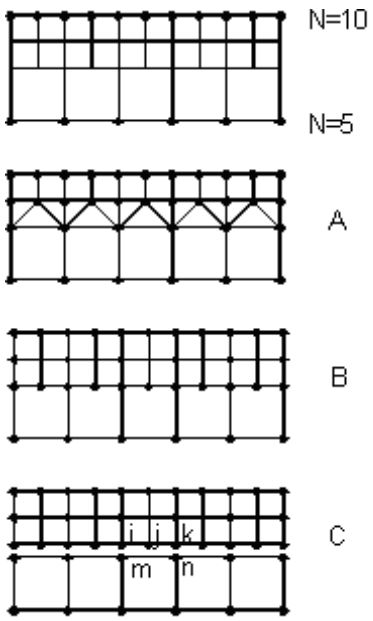
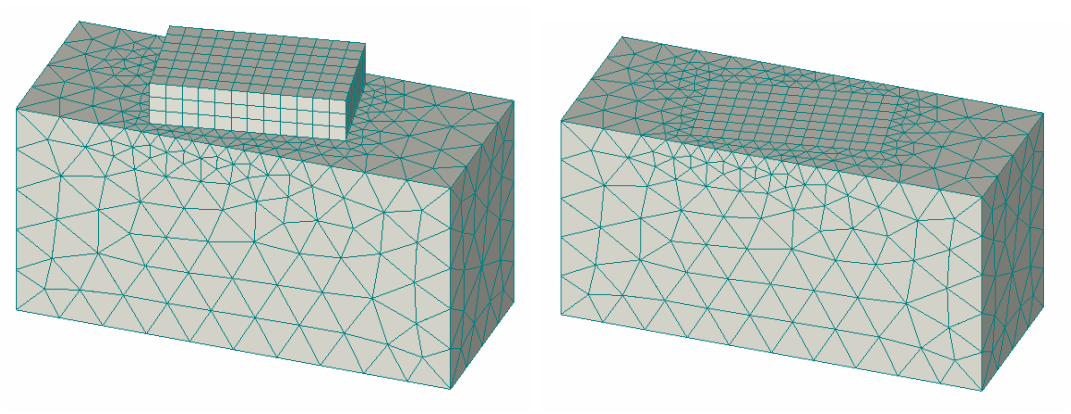


Fig. 10-1 Mesh refinement

Note that all the above techniques are supported in ATENA finite element package, the last one requiring implementation of CBCs in the form (9.14).

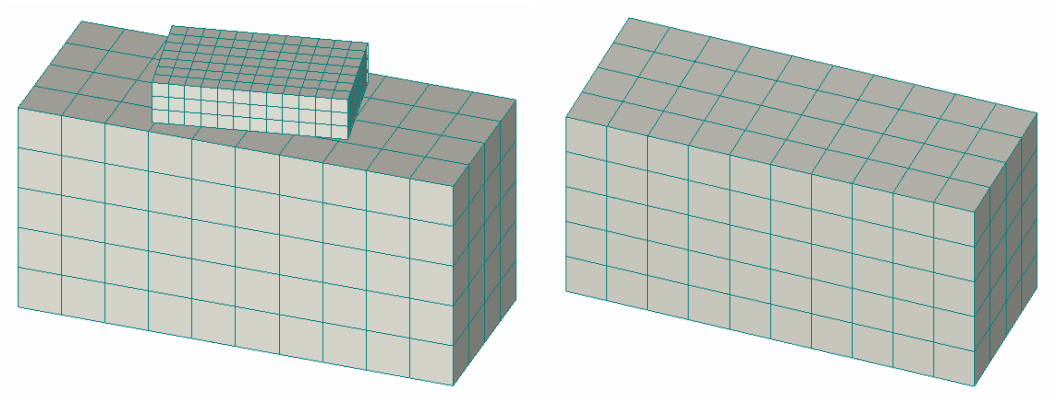
10.2.2 Mesh Generation Using Sub-Regions

This example demonstrates another advantage of using the proposed CBCs: It is possible to generate meshes within sub-regions without requirement of nodes coincidence on their interfaces. Because mesh structure on the sub-regions' surfaces is not prescribed, this approach provides more flexibility to mesh generation. This feature is heavily used by ATENA 3D pre-processor.



Compatible meshes on the contact between the blocks

Fig. 10-2 Mesh generation from simple blocks



Incompatible meshes on the contact between the blocks using CBCs

Fig. 10-2 (cont) Mesh generation from simple blocks

In the above example, two blocks are connected to form a structure, where the top (smaller) block is placed atop of the bottom (larger) block. The position of the top block is arbitrary with respect to the bottom block. Unless the concept of CBCs is used, the meshes on the interface of the two blocks must be compatible (see top of Fig. 10-2). On the other hand, the proposed CBCs allow using of incompatible meshes (see the bottom of Fig. 10-2). In this case, the mesh in each block is generated independently, which is significantly simpler. After they are done, the proposed CBCs are applied to connect the interface nodes. (Typically, the surface with the finer mesh is fixed to the surface with the coarse mesh). The latter approach also demonstrates the possibility of a mesh refinement while still using well-structured and transparent meshes. This is particularly useful in the case of complex numerical models.

10.2.3 Discrete Reinforcement Embedded in Solid Elements

In this example, the described boundary conditions are used to simplify the modeling of the reinforced concrete beam, see Fig. 10-3. The procedure to create the model is as follows. Firstly, the mesh for solids, i.e., concrete elements are generated. It poses no problem, as it is a regular mesh consisting of 48 quadrilateral elements. At this point, no attention needs to be paid to the geometry of reinforcing bars present in the beam. Thereafter, the reinforcing bars are inserted and their meshes are generated based on the existing mesh of solid elements. The first step is to find all nodes, where the bar changes direction. These nodes are called principal nodes; see e.g., node *n* in Fig. 10-3. Then, the intersection of all straight parts of the bar with underlying solid elements are detected, e.g., the nodes *m,p*. Thus, all end nodes of embedded bar elements are defined. The last step is to link displacements of the nodes of the bar to the underlying solid elements.

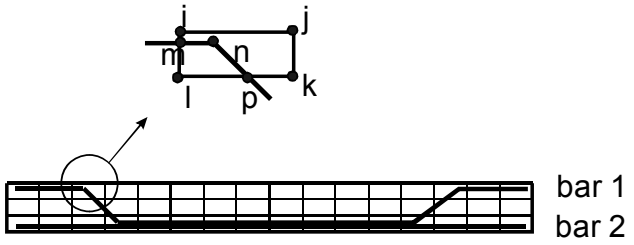


Fig. 10-3 Discrete bar reinforcement

For example, if we want to connect the node *n* to the an embedding solid element, i.e., to nodes *i,j,k,l*, see Fig. 10-3, we use the standard interpolation $u(r,s)=\sum_{i=1}^{4} h_i(r,s)U_i$, where $h_i(r,s),U_i$ are element interpolation function and U_i are nodal displacements for the underlying solid element, respectively. For displacement at the node *n* we can write $u(r_n,s_n)=\sum_{i=1}^{4} h_i(r_n,s_n)U_i$. (r_n,s_n) are coordinates of the node *n*. Comparing this formula with (9.3), it is obvious that $\alpha_{ni}=h_i(r_n,s_n)$, $u_{n0}=0$. Consequently, the bar DOFs are always kinematically dependent on the DOFs of underlying solid elements.

This technique can also be applied when bond elements are inserted between solid and embedded bar elements. This is treated in a separate paper by authors in ref. (Jendele, 2003).

Currently, ATENA software can generate discrete reinforcement to all 2D and 3D linear and nonlinear elements (triangles, quads, tetrahedral elements, wedges, bricks…). The user only draws the position of the principal nodes of reinforcement bars and the rest is done automatically.

10.2.4 Curvilinear Nonlinear Beam and Shell Elements

In the following text, another possible use of the present boundary conditions is presented. A curvilinear nonlinear beam from Chapter **3.17** is discussed. A particular feature we would like to point out here is that although it originally has only three displacements and three rotations in the nodes 13,14,15, see **Fig. 3-40**, its implementation in ATENA has also 3 displacements in the nodes 1 to 12. However, these DOFs are linked to the original DOFs in the nodes 13 to 15 by the proposed CBCs. This concept has several advantages.

- The beam finite element has native 3D geometry and its pre- and post-processing visualization is more realistic than using its original 1D geometry.
- It is simple to connect such beam elements to any adjacent 3D finite elements, e.g., brick elements.
- Mesh generation is easily done by any 3D solid element generator that can pull off nonlinear hexahedral elements. It suffices to generate only the nodes 1 to 12 (with 3 displacement DOFs) and the three original beam nodes (each beam node has 3 displacement and 3 rotation DOFs) are generated automatically. The pre-processor need not to support rotational DOFs.

- The post-processing of this element and an ordinary nonlinear hexahedral element is the same. Consequently, this element does not need any extra support for the visualisation of the results. It makes its implementation and use simple.

Derivation of all α_{ij} coefficients and u_{i0} constants for all nodes 1 to 12 is beyond the scope of this document. Nevertheless, a similar procedure is used, as it was in the previous example.

ATENA package also covers Ahmad element for curved shell structures, see Chapter **3.12**. The usual 2D shape of the shell element is in the same manner, replaced by geometry of a 3D nonlinear hexahedral element. Originally, the shell element has 3 displacements and 2 rotations at each node located in the middle thickness of the shell. These 5 DOFs are in by use of CBCs replaced by 3 displacements at the top and 2 displacements at the bottom at the respective nodes from the hexahedron, (i.e., brick) geometry. Advantages of this approach are the same as those in the case of the curvilinear beam above: simpler pre/post-processing, simpler connection to the adjacent 3D elements, no need to support rotational DOFs during pre/post-processing, no need for extra support for geometry of the shell element.

10.3 References

BATHE, K.J. (1982), Finite Element Procedures In Engineering Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632, ISBN 0-13-317305-4.

JENDELE, L, CERVENKA, J, CERVENKA V., " Bond in Finite Element Modelling of Reinforced Concrete", Proceedings Euro-C 2003, Computatinal Modelling of Concrete Structures, Swets & Zeitlinger, Lisse, The Netherlands, ISBN 90 809 536 3, 793-8036

INDEX

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

W

X