Eilmer3: a CFD code and its validation (A quarter of a century spent "doing sums".)

Peter Jacobs and *Many* Others
School of Mechanical and Mining Engineering
The University of Queensland

19 June 2014

Potted history of CFD technology for high-speed flow

- in the late 1980s, the state of the art for scramjet simulations involving reactive flow was JP Drummond *SPARK* code
- Flow solver component based on Bob McCormack's (1969) finite-difference shock-capturing technique.
- All configuration hard-coded into the Fortran source code and compiled to run on a Cray supercomputer.
- To capture shocks in the T4 scramjet experiments, needed excessively large artificial-pressure coefficients to suppress oscillations.
- In the 1980s, a new CFD technology (upwind flux) was being developed by the applied mathematics people and parallel computing environments were being developed by the computer science people (cluster computers).

The early years (of Eilmer) – last century.

- Dec 1990: following a CFD lesson on the chalk-board from Bob Walters and Bernard Grossman, *cns4u* was started with the intention to be like SPARK but with new technology
- Dec 1992: back in Brisbane started *L1d* (at WBM-Stalker) to reverse-engineer other groups shock tunnels
- 1993 built *sm3d*, a space-marching code for 3D scramjet flows
- 1995 through 1999: the postgrad years expanded scope of experimentation and application
  - *sm3d+* Chris Craddock, chemistry, optimization, scramjets
  - *pamela* Andrew McGhee, MPL parallelization
  - *u2de* Paul Petrie, unstructured, adaptive grid, shock tubes
  - *sf2d, sf3d* Ian Johnston, structured moving grid with chemistry and shock capturing, aerothermodynamics of entry vehicles
- 1996: code reformulation around fluxes (frequent discussions with Mike Macrossan); all code still in C with a preprocessor having a little command interpreter built in.

...the dark ages...

- 1997: discovered scripting languages Tcl and Python
- 1999-2000: Vince Wheatley, rarefied flows, L1d, another go at finite-rate chemistry
- 1999-2000: James Faddy, experiment with solution-adaptive 2D solver
- 1999-2002: Richard Goozee, SPH experiment, get to grips with MPI
- 2001-2002: Michael Elford: validation exercise with Fastran (commercial code)
- May 2003: *scriptit.tcl* provided fully programmable environment for simulation-preparation.
- Aug 2004: *Elmer* began as a hybrid code using Python and C.

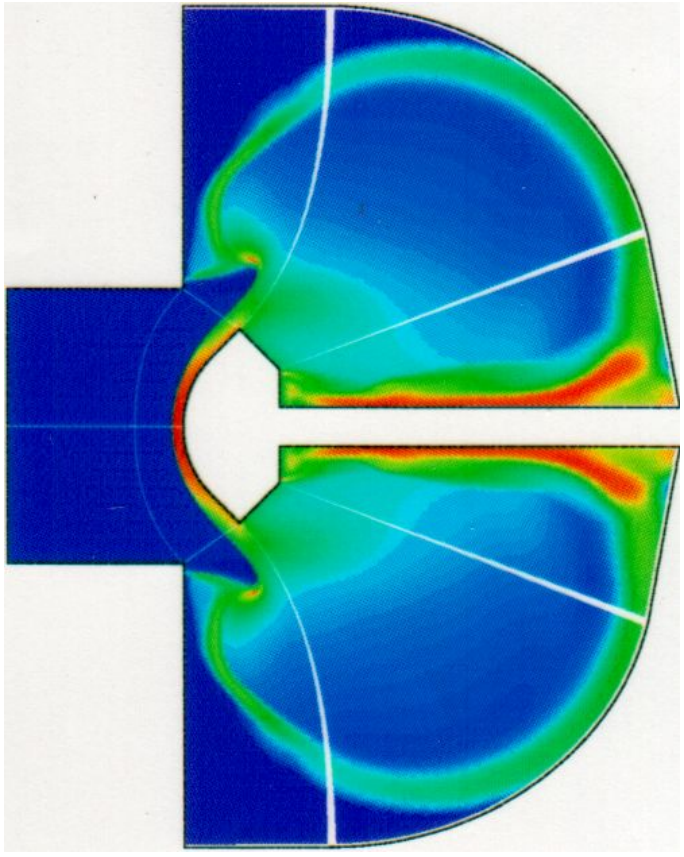
...the Renaissance.

- Feb 2005: Realized that Python was much nicer for occasional users. (*scriptit.py*)
- Jun 2005: rewrite of *Elmer(2)* in C alone so that Andrew Denman could get on with his thesis
- 2005: more experimentation in CFD methods; Joseph Tang wrote a hierarchical-grid solver based on virtual-cell embedding.
- Mar 2006: *mbcns2* was a rewrite of *mbcns* but with C++ to manage code complexity.
- Jul 2006: rewrite *Elmer2* in C++
- Sep 2007: this is crazy; we should merge the 2D and 3D codes
- Jan 2008: make *mbcns2* look more like *Elmer2*
- Nov 2008: and call it *Eilmer3*.

Since 2008: changes to documentation and capability.

- User Guide (447pp) and Theory Book.
- class-based implementation of boundary conditions; easier to extend and maintain.
- generalization of the solution files; expandable content (easy to add variables without breaking postprocessing)
- generalization of the postprocessing code as a library; specialized postprocessors are easy and can be mixed with preparation of new simulations.
- rework of housekeeping (subdirectories for files storage; gzipped grid and flow files) so that we can scale.
- generalization of thermochemistry, easily extendable multiple temperature model (Rowan Gollan and Dan Potter)
- coupled radiation (Dan Potter)
- turbomachinery (Carlos Ventura, Jason Czapla, Jason Qin)
- meso-scale combustion (Anand, Xin)

Eilmer3 in a nutshell



- Eulerian description of the gas (finite-volume, axisymmetric or 3D)
- Transient, time-accurate
- Shock capturing
- Multiple-block, structured grids
- Parallel computation on a cluster computer, using MPI
- *Really good* thermochemistry module by Rowan Gollan and Dan Potter

Mathematical gas dynamics (in differential form, by RJG)

Conservation of mass:

$$\frac{\partial}{\partial t}\rho + \nabla \cdot \rho \mathbf{u} = 0 \tag{1}$$

Conservation of species mass:

$$\frac{\partial}{\partial t}\rho_i + \nabla \cdot \rho_i \mathbf{u} = -\left(\nabla \cdot \mathbf{J}_i\right) + \dot{\omega}_i \tag{2}$$

Conservation of momentum:

$$\frac{\partial}{\partial t}\rho \mathbf{u} + \nabla \cdot \rho \mathbf{u}\mathbf{u} = -\nabla p - \nabla \cdot \left\{-\mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\dagger}) + \tfrac{2}{3}\mu(\nabla \cdot \mathbf{u})\boldsymbol{\delta}\right\} \tag{3}$$

Conservation of total energy:

$$\begin{aligned}\frac{\partial}{\partial t}\rho E + \nabla \cdot (e + \frac{p}{\rho})\mathbf{u} = \nabla \cdot [k\nabla T + \sum_{s=1}^{N_v} k_{v,s}\nabla T_{v,s}] + \nabla \cdot \left[\sum_{i=1}^{N_s} h_i \mathbf{J}_i\right] \\ - \left(\nabla \cdot \left[\left\{-\mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\dagger}) + \tfrac{2}{3}\mu(\nabla \cdot \mathbf{u})\boldsymbol{\delta}\right\} \cdot \mathbf{u}\right]\right) - Q_{\mathrm{rad}}\end{aligned} \tag{4}$$

Conservation of vibrational energy:

$$\frac{\partial}{\partial t}\rho_i e_{v,i} + \nabla \cdot \rho_i e_{v,i}\mathbf{u} = \nabla \cdot [k_{v,i}\nabla T_{v,i}] - \nabla \cdot e_{v,i}\mathbf{J}_i + Q_{T-V_i} + Q_{V-V_i} + Q_{\mathrm{Chem}-V_i} - Q_{\mathrm{rad}_i} \tag{5}$$

More maths...

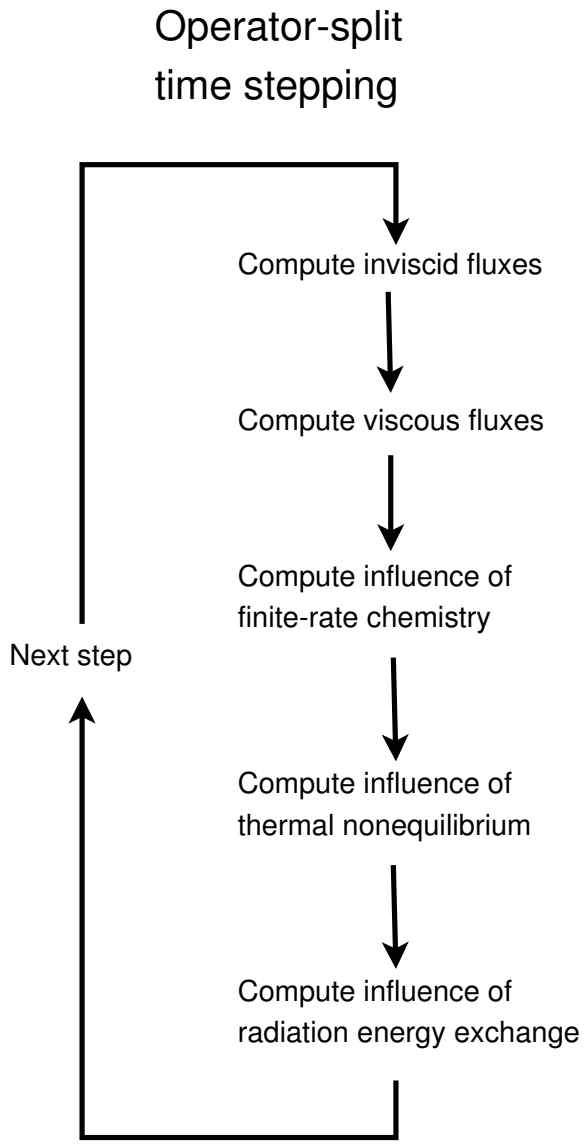
Thermodynamic model of the gas...
Finite-rate chemical kinetics...
Radiation energy exchange...
Boundary conditions...

Numerical Methods

- nonlinear function solvers – secant, Newton, fixed-point methods
- linear equation solvers – full, direct methods, iterative Jacobi, Gauss-Seidel
- (single an multidimensional) interpolation and reconstruction of data – polynomials, Bezier curves, splines, NURBS
- finite-differences (to turn our PDEs into algebraic equations or ODEs)
- quadrature / integration – Newton-Cotes, Gaussian
- ordinary-differential-equation integrators – Euler, predictor-corrector, Runge-Kutta schemes

Software implementation

- C++ for the core solver and update computations for the physical processes
- Parallel computation on a cluster computer, using MPI
- Python scripting for pre- and post-processing
- Lua for user-controlled run-time configuration in boundary conditions and source terms
- Lua for thermochemical configuration.



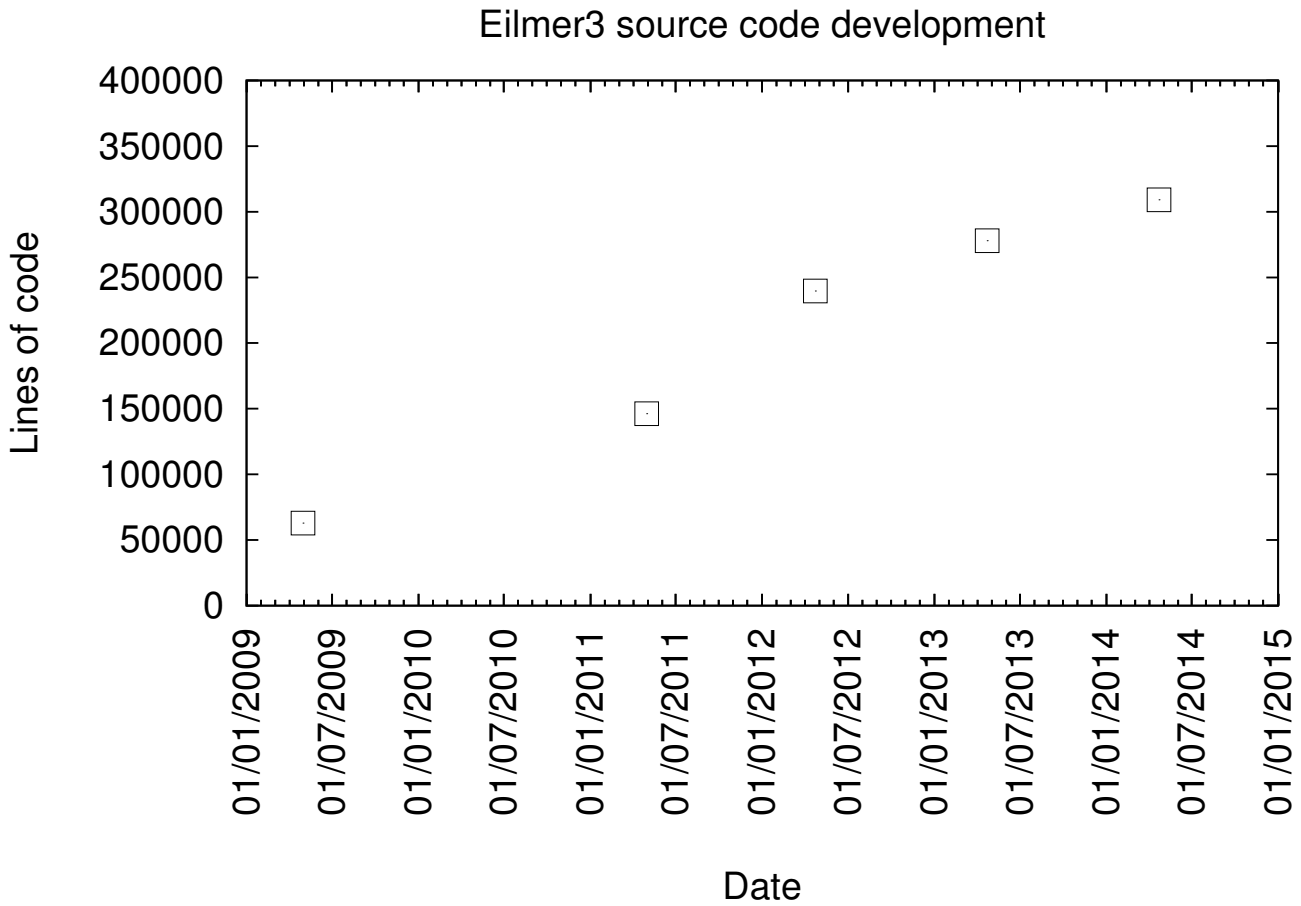
Software Engineering

- Languages
  - Fortran is OK, but why bother.
  - C/C++ is for experts, and all roads lead to C.
  - Python convenient for the top-level code that the user sees.
  - Lua is easily embeddable (core C/C++ code calls user-defined functions written in Lua)
  - D is the future for statically-typed, compiled code.
- Other tools
  - Editors (emacs) with syntax-highlighting.
  - Source code revision control – mercurial.
  - OS environment – Linux on cluster computers.
  - Compilers (to machine code)
  - Code checkers – static, memory access (Valgrind)

Kernighan's Law: "Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it."

Source-code line count

Just Eilmer3, since the 2008 reboot. It's a 5000 page *document*.



Source-code line counts

Lines of code, including white-space and comments (April 2014):

Code module	cxx+hh	Python	Lua	LaTeX	Total
gas-dynamics core	45 149	15 579	0	136	60 864
thermochemistry	32 779	6 015	58 825	2280	99 899
radiation	20 296	22 573	107	3047	46 023
geometry2	15 048	2 403	228		17 679
cfpylib	0	9 245	0	0	9 245
examples	0	26 699	32 442	16 274	75 415
Total	113 272	82 514	91 602	21 737	309 125

Of the Lua code 92% is for a database in human-readable format.
36% for collision integrals; 36% for reactions schemes; 20% for thermo and transport properties. (Nov 2012)

Count lines in files *.cxx *.hh *.py *.lua

```
$ find .  -name "*.cxx" -exec wc '{}' \; | awk 'BEGIN {sum=0} {sum+=$1} END {print sum}'
```

Verification and Validation Examples

Verification Examples:

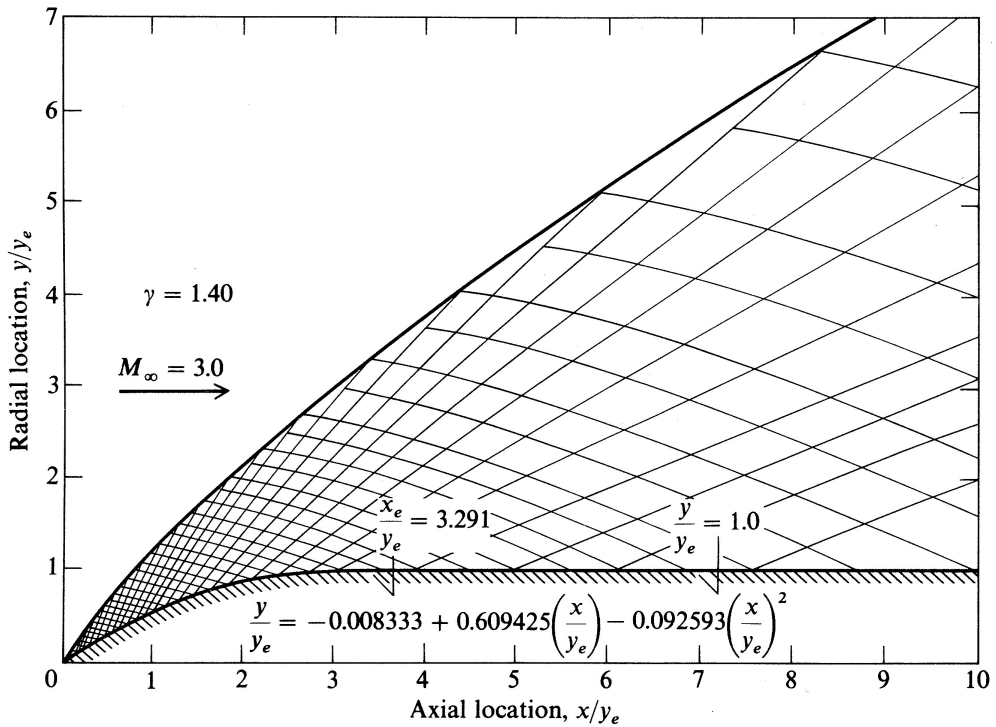
- Are we solving the equations correctly?
- Compare with numerical solutions from other codes.
- Exact solution: oblique detonation wave (idealized chemistry).
- Manufactured solution that we must match (using special source terms and BCs).

Validation Examples:

- Are we solving the correct gas-dynamic equations?
- Sphere in reacting flow.
- Shock + boundary-layer interaction.
- Turbomachine compressor.
- Need high quality experimental data.

Sharp nosed projectile

- Original Zucrow & Hoffman; also Anderson's Hypersonics text
- Shape of surface defined by polynomial equation
- Can compare numerical solutions



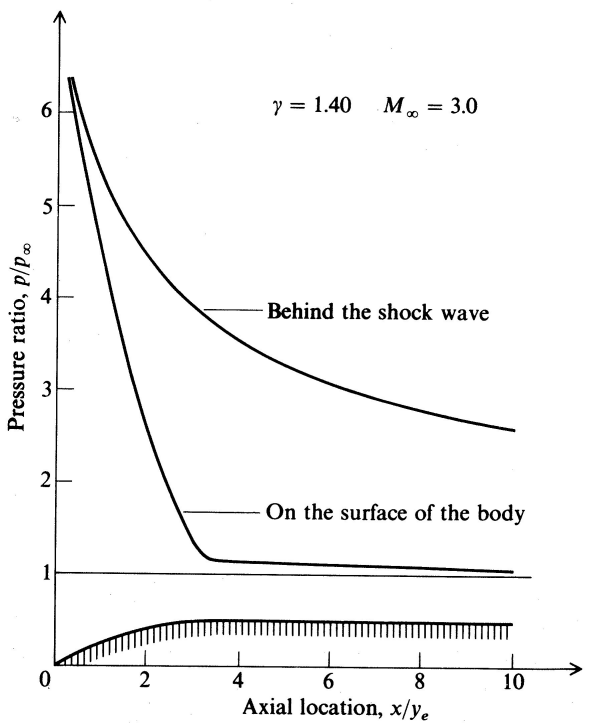


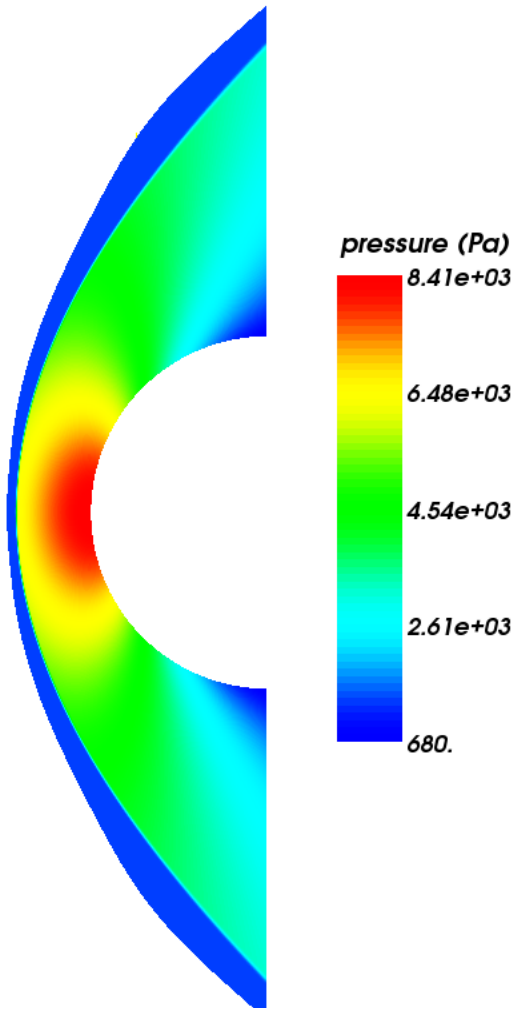
FIGURE 5.5
A typical characteristics mesh. (*From Zucrow and Hoffman, Ref. 53.*)

Validation exercise for gas dynamics

Schwartz & Eckerman tested their measurement technique of shock detachment distance by firing ball bearings into noble gases.

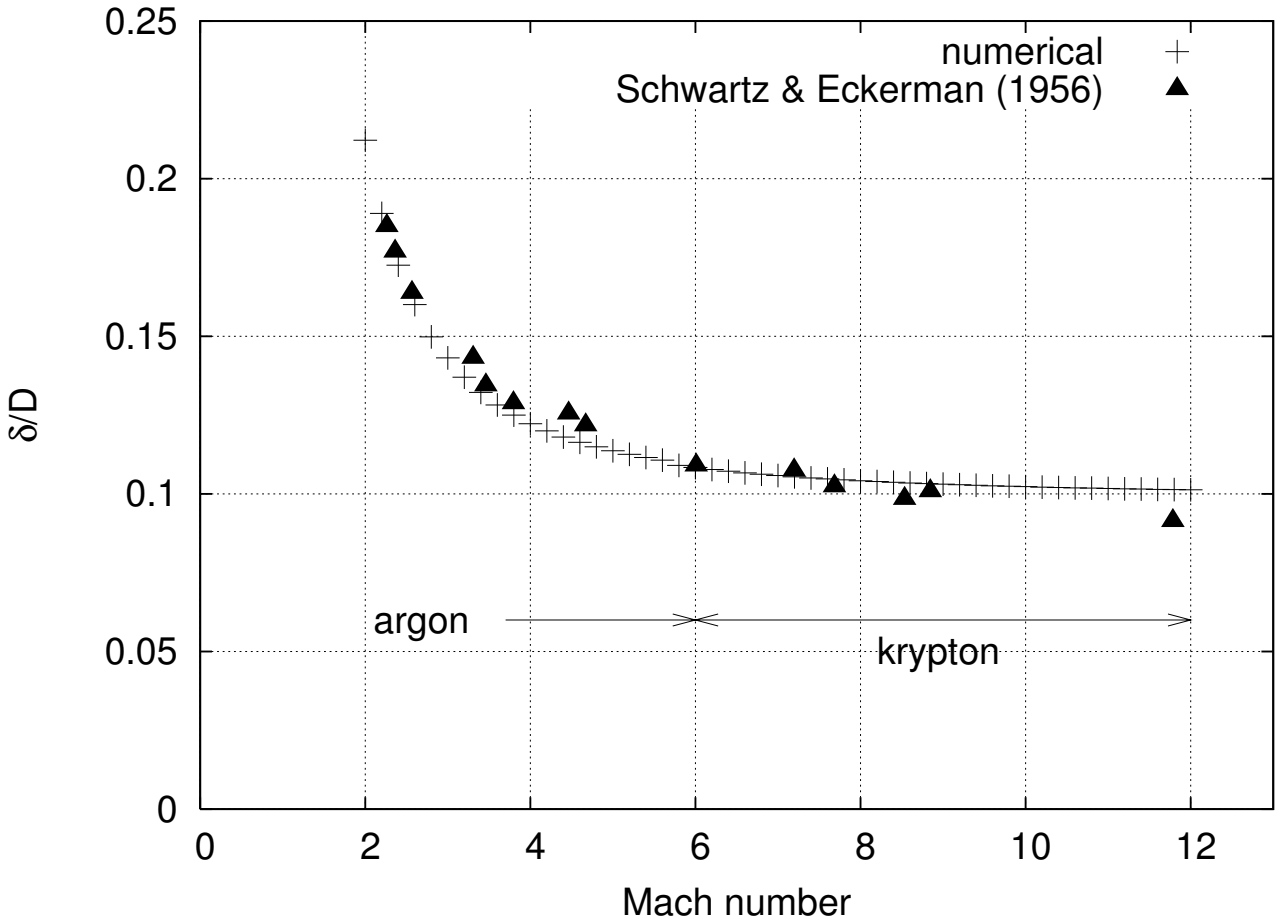
Argon flows	
M_∞	2 – 6
T_∞	300 K
p_∞	10, 100, 200 mmHg

Krypton flows	
M_∞	6 – 12
T_∞	300 K
p_∞	10, 100, 200 mmHg



Argon flow, M = 2.0, p = 10.0 mm Hg)

Shock detachment in noble gases



Mohammadian's convex ramp

Experiment done in the Imperial College gun tunnel with the Mach 12 nozzle. Experiment provides surface pressure and heat transfer measurements.

FIGURE 4. Schlieren photograph of flow over the rear part of the convex surface; $M = 12{\cdot}25$, $R = 0{\cdot}86 \times 10^5$.

MOHAMMADIAN

Mohammadian's convex ramp – surface pressure

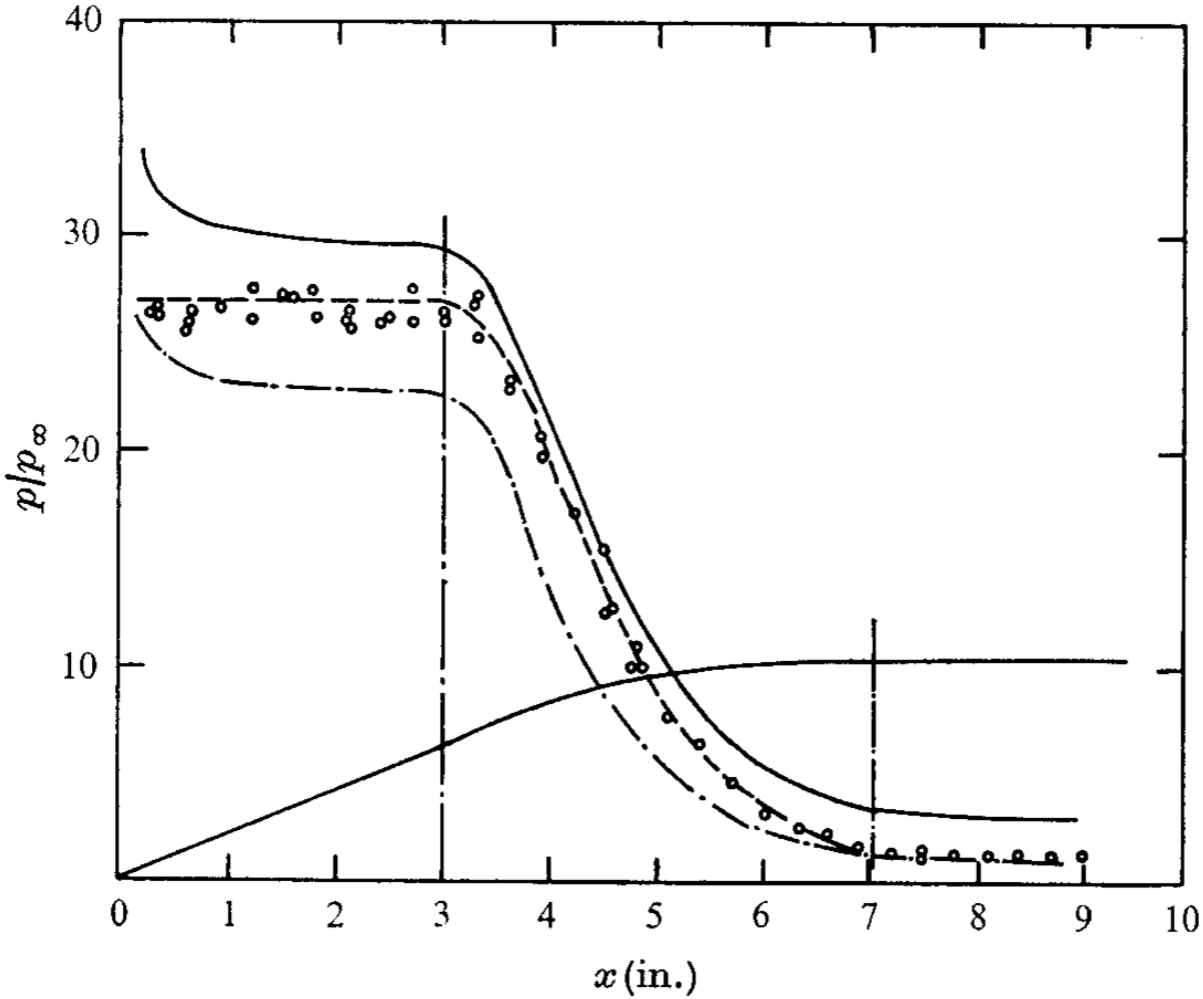


FIGURE 12. A comparison of pressure measurements on the convex model with results from different theories; $M = 12{\cdot}25$, $p_0 = 1600$ psia, $T_0 = 1300\,^\circ$K, $R = 0{\cdot}86 \times 10^5$. ○, experimental data; —·—, Cheng's original method, ——, modified Cheng method; – – –, tangent-wedge (inviscid) method.

Mohammadian's convex ramp – surface heat transfer

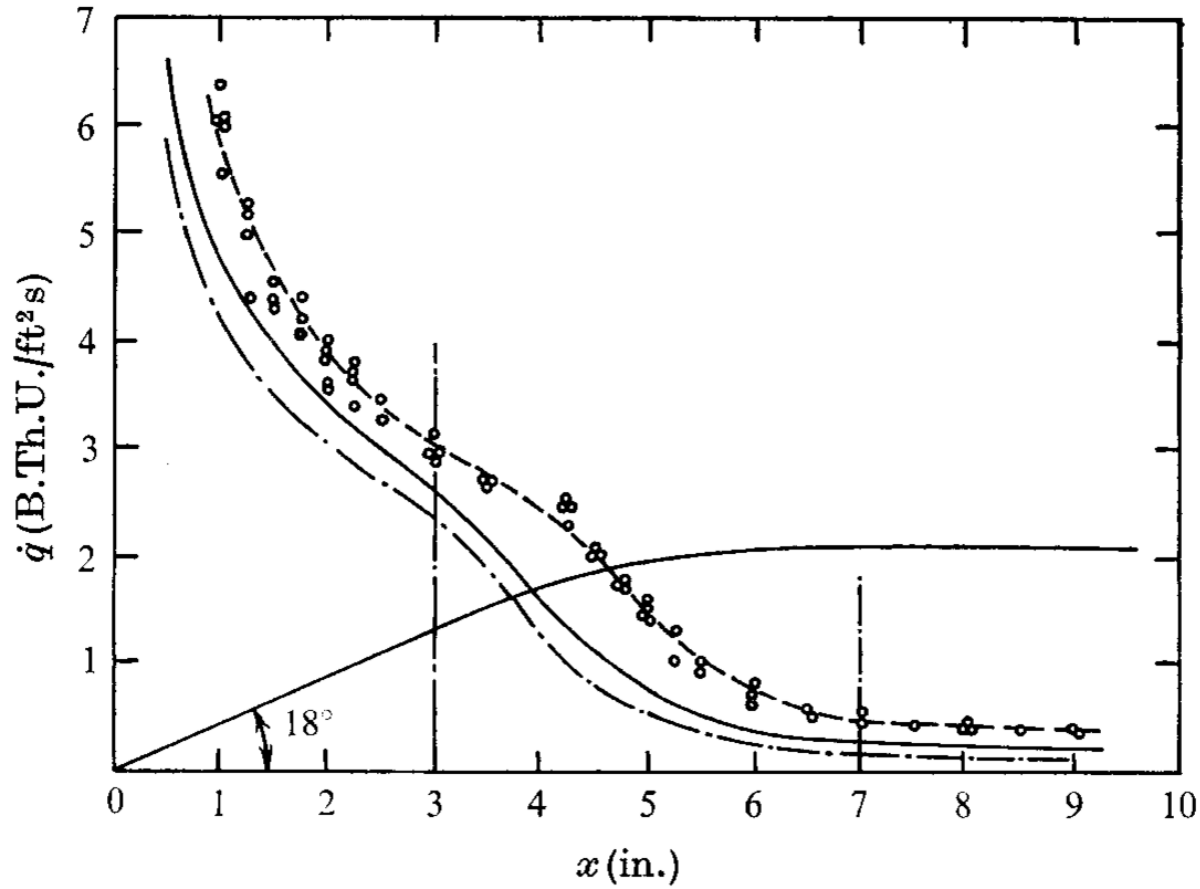
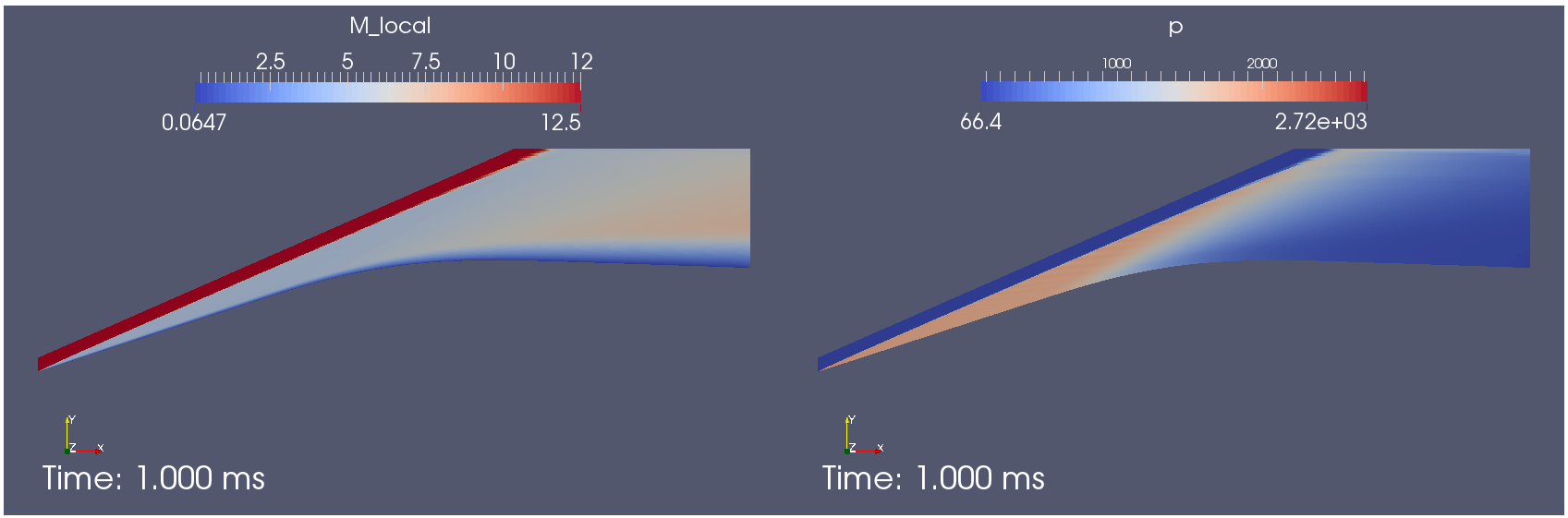


FIGURE 13. A comparison of heat-transfer measurements on the convex model with results from the viscous-interaction theories; $M = 12{\cdot}25$, $T_0 = 1300\,°\mathrm{K}$, $p_0 = 1300\,\mathrm{psia}$, $R = 0{\cdot}86 \times 10^5$. ○, experimental data; —·—, Cheng's original theory; ——, modified Cheng method.

Mohammadian's convex ramp – Eilmer3 simulation

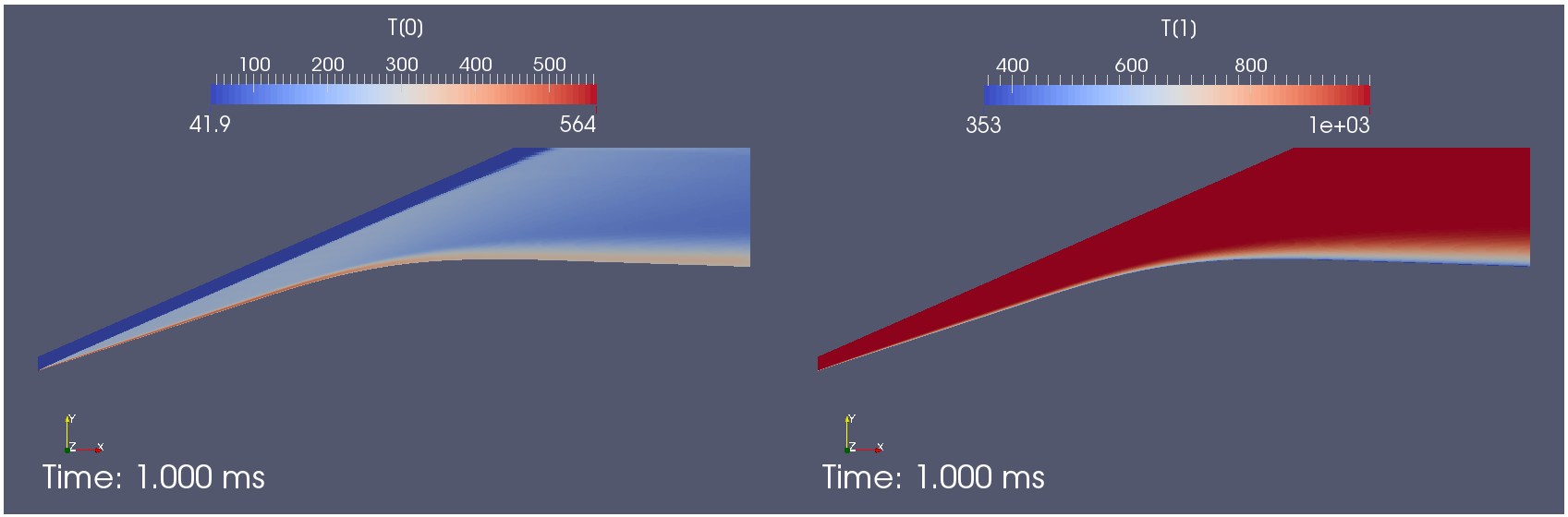
We reverse-engineered the free-stream conditions and the ramp shape from incomplete descriptions in the JFM paper.



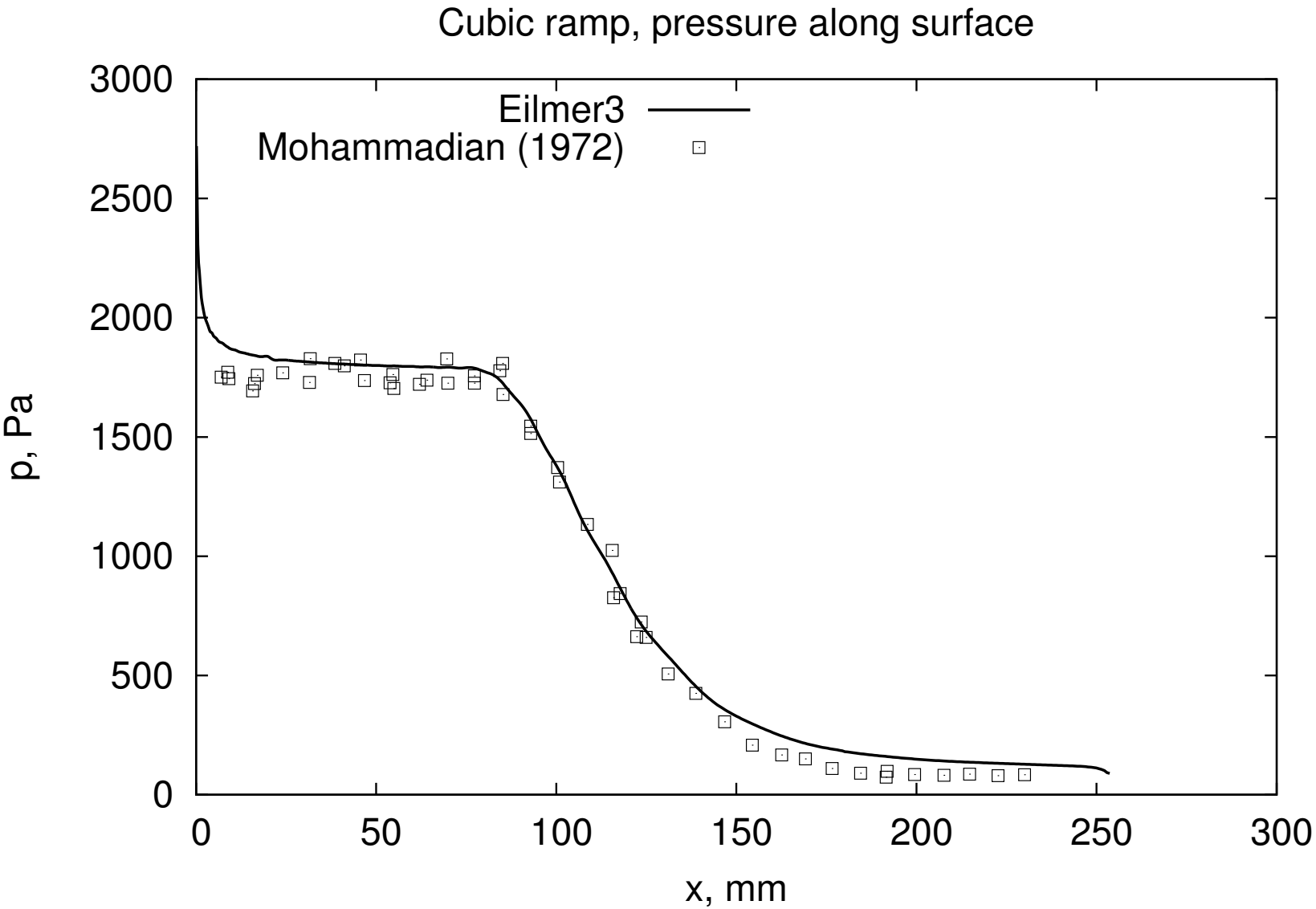
Mohammadian's convex ramp – Eilmer3 simulation

Thermodynamic nonequilibrium is included.

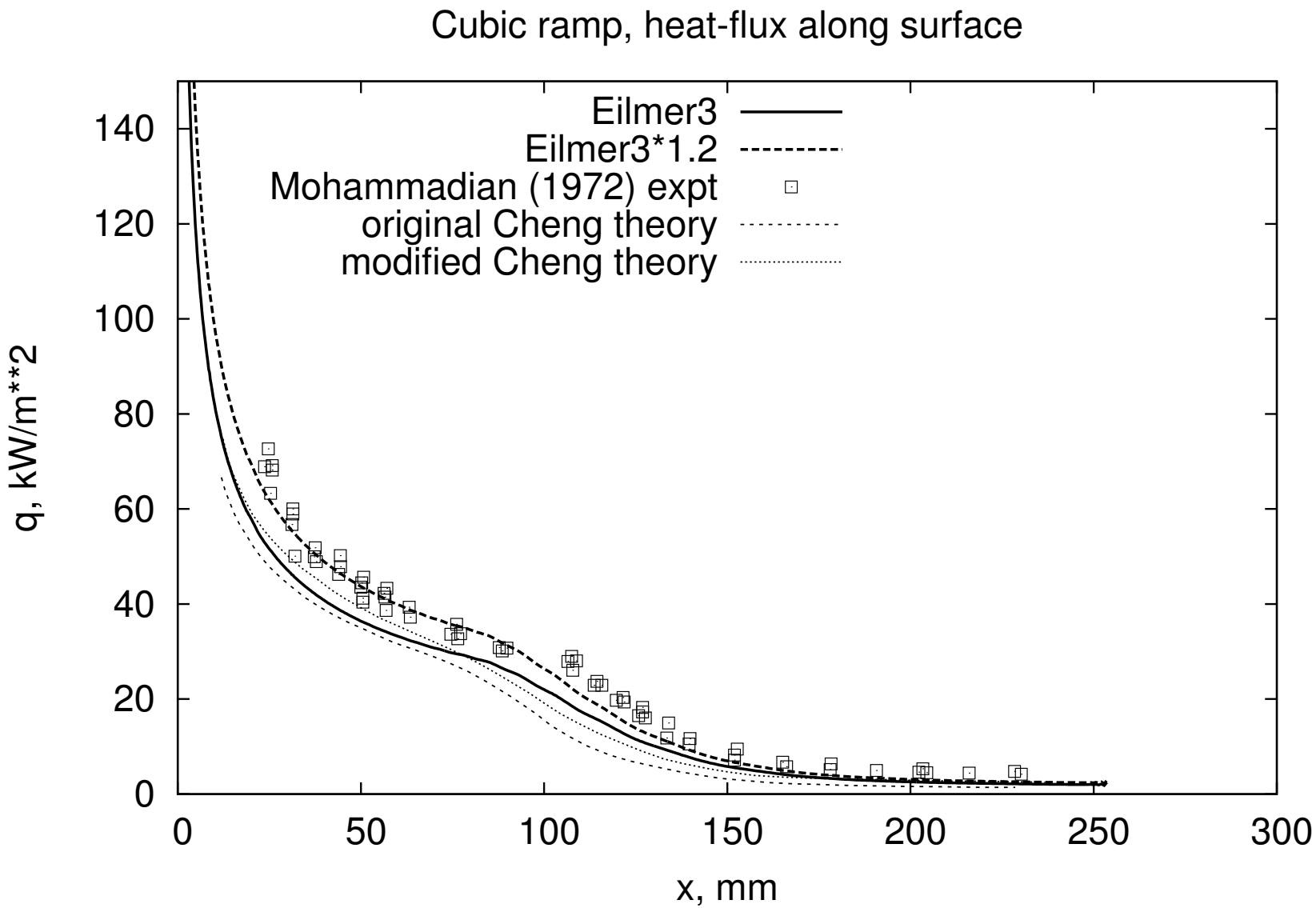
- T[0] is the static temperature;
- T[1] is the vibrational temperature.



Mohammadian's convex ramp – compare pressure

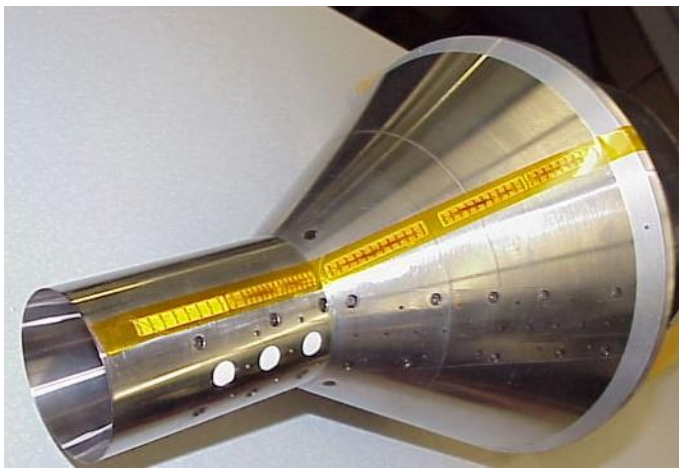


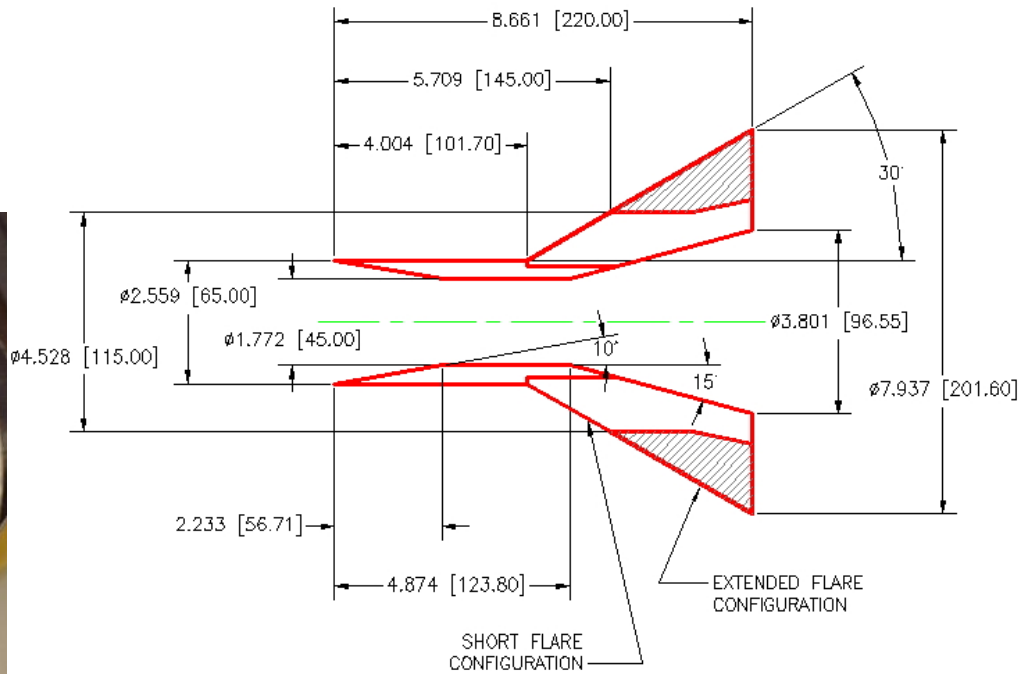
Mohammadian's convex ramp – compare heat transfer



Cylinder with flare – CUBRC experiment

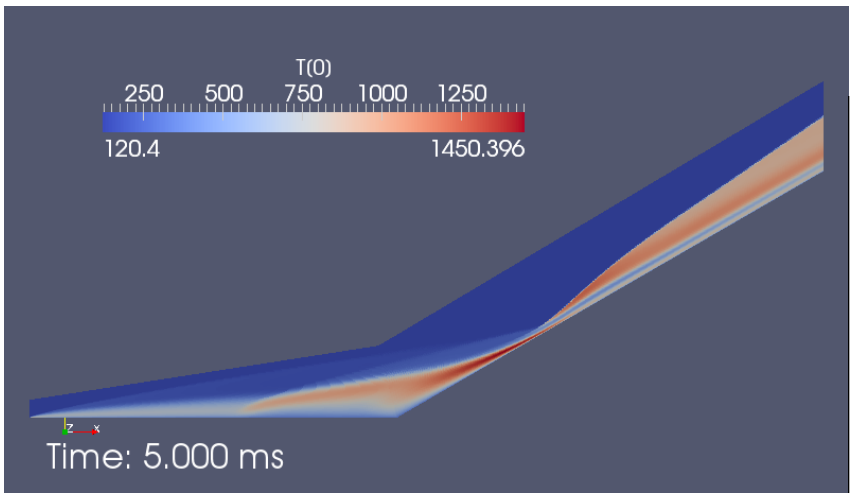
- Holden and crew have gone to great lengths to generate high-quality experimental data for CFD validation.

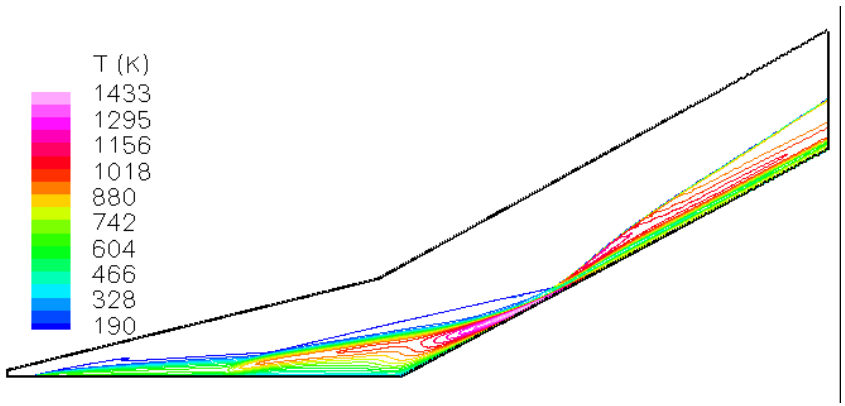




Cylinder with flare – compare simulations

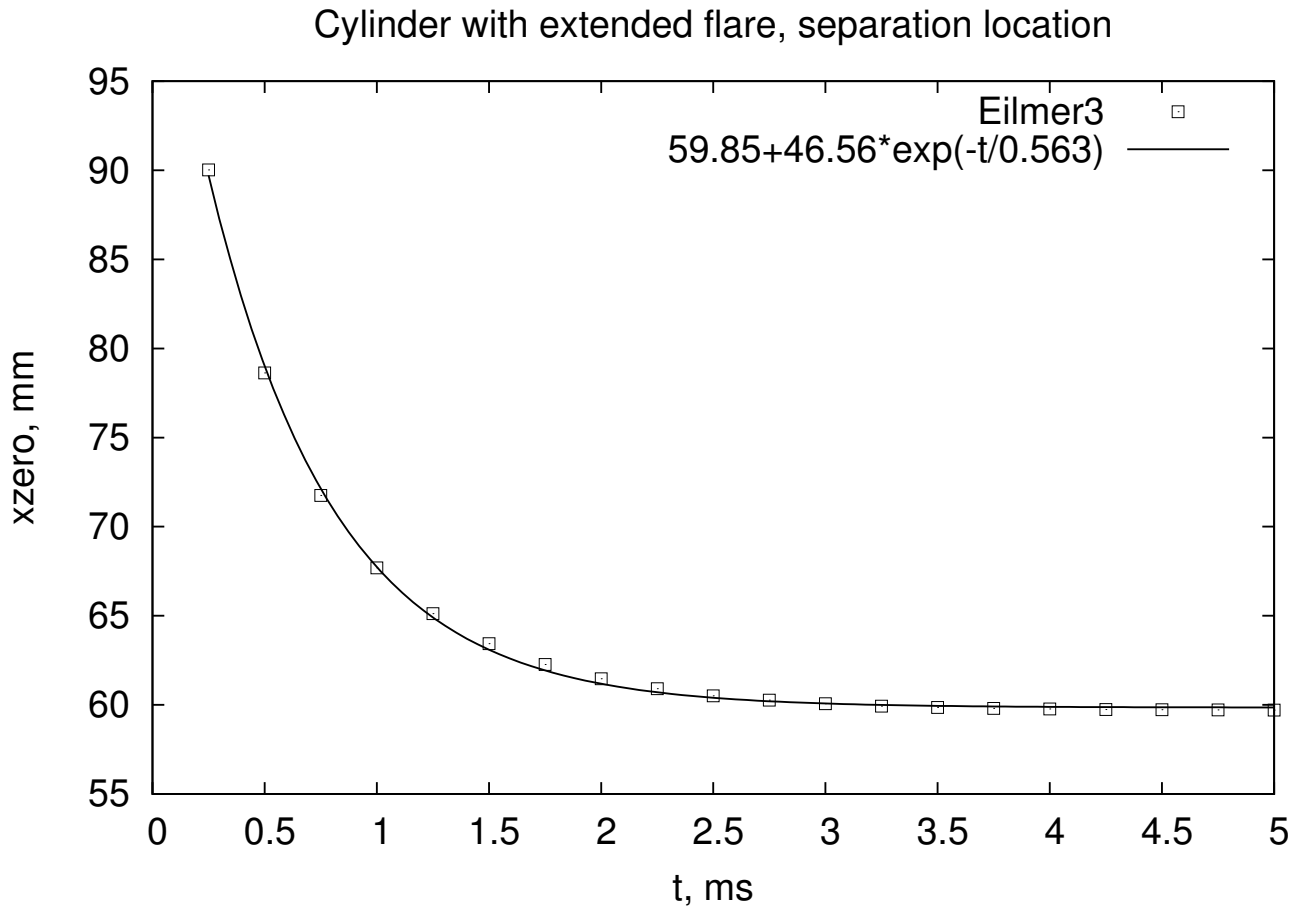
- MacLean's CFD data from AIAA Paper 2004-0529.
- Also, describes the experiment's free-stream conditions.
- Strong viscous interaction at leading edge.
- Separation zone sensitive to on-coming boundary layer conditions and shock interaction.





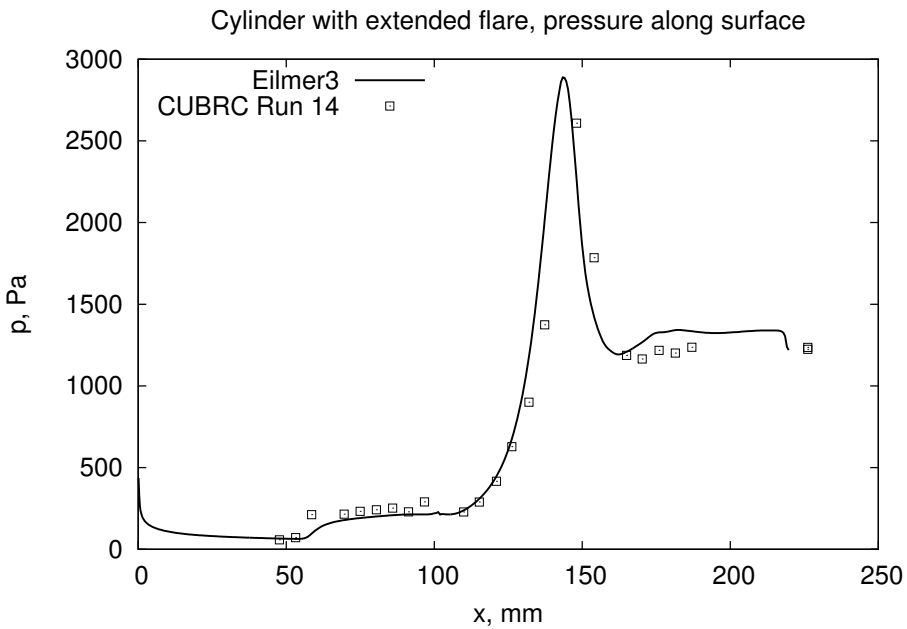
Cylinder with flare – separation zone

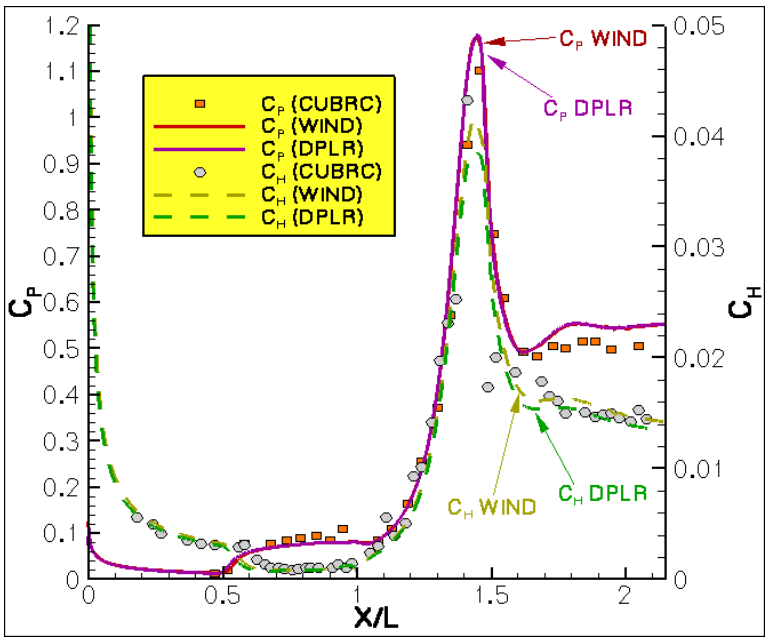
- Remember that Eilmer3 is a *simulation code*.
- The flow features develop according to the rules of gas-dynamics (see slide 8).



Cylinder with flare – compare surface pressure

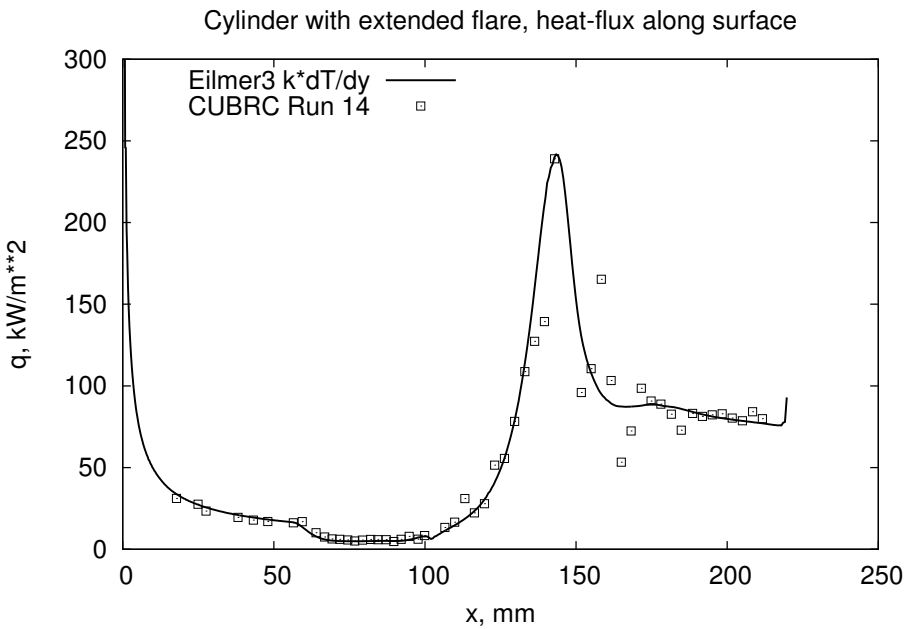
- MacLean's CFD data from AIAA Paper 2004-0529.
- Experimental data supplied in a spreadsheet file (in dimensional form).

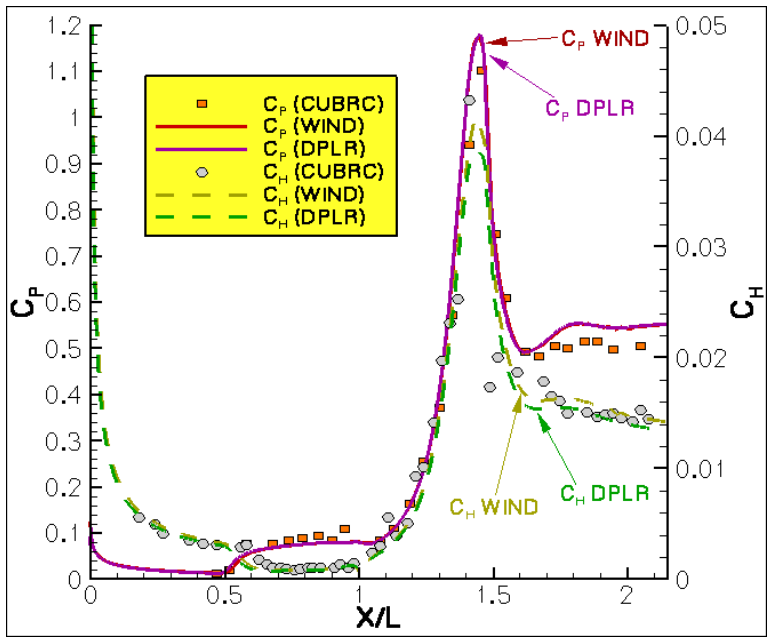




Cylinder with flare – compare heat transfer

- MacLean's CFD data from AIAA Paper 2004-0529.
- Experimental data supplied in a spreadsheet file (in dimensional form).



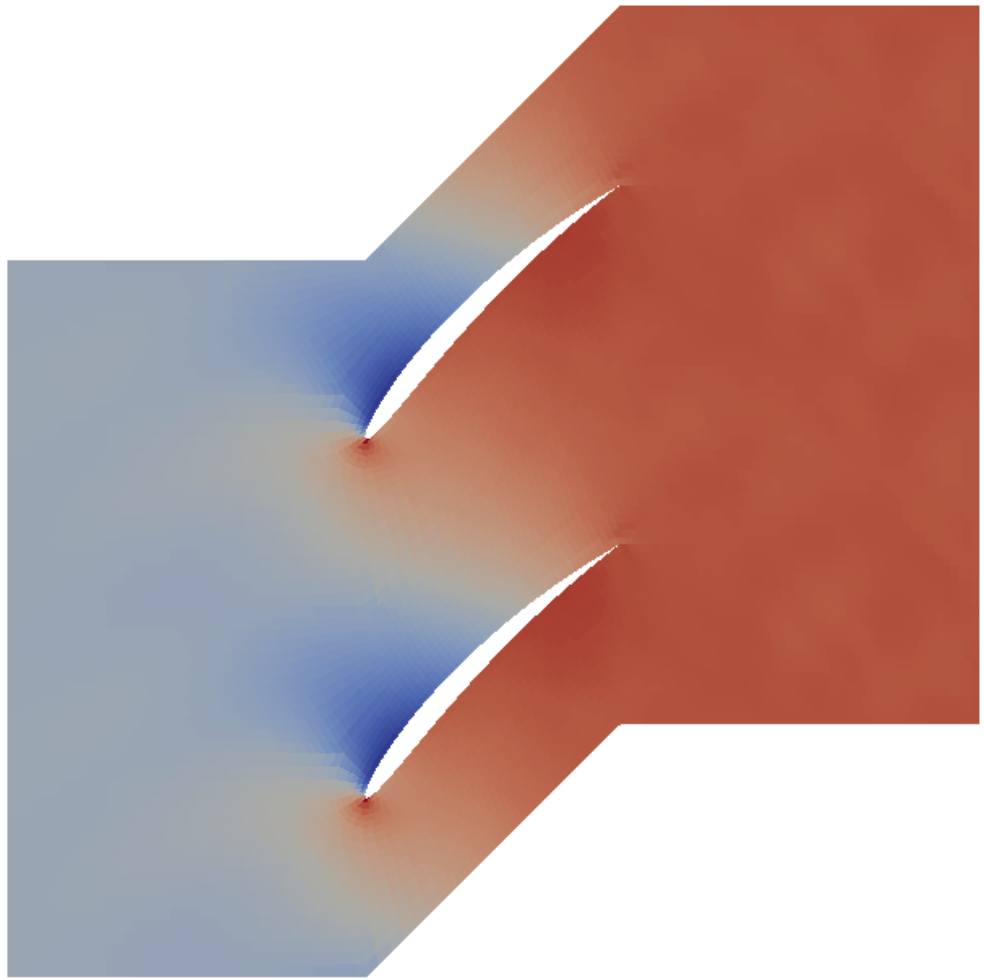


Turbomachinery extensions

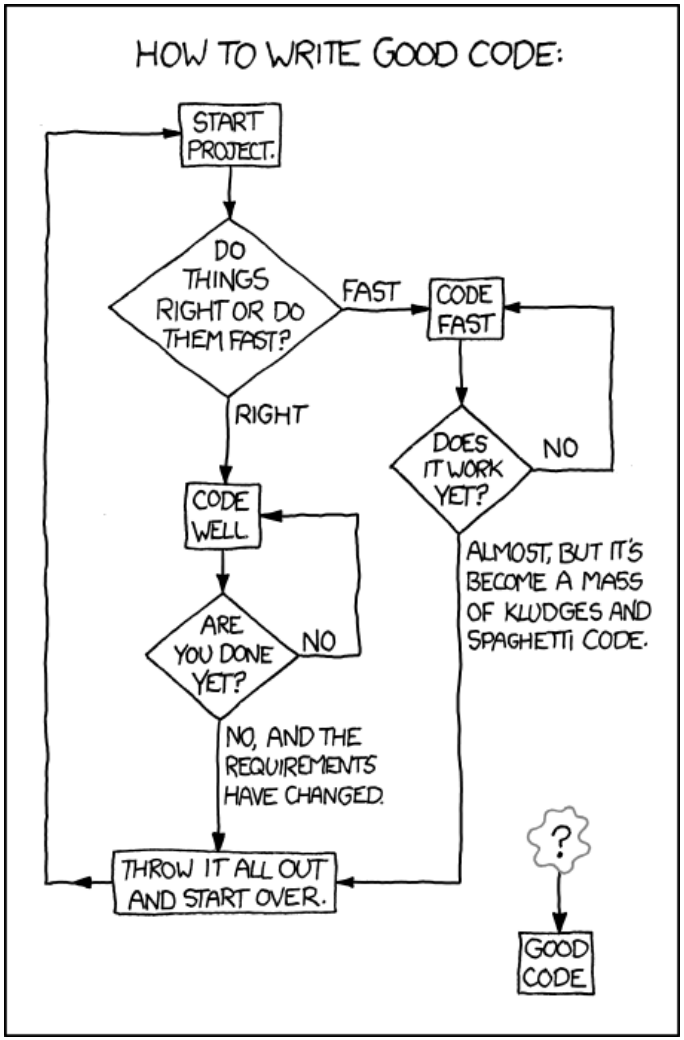
- new flow solver components
  - rotating frame of reference for some blocks
  - body forces: centrifugal and Coriolis
  - energy eqation uses *rothalpy*
- new boundary conditions
  - subsonic inflow and outflow, with swirl (SubsonicInBC)
  - periodic boundaries (MappedCellBC)
  - mixing-plane between rotating and non-rotating blocks (ExchangeWithMixingPlaneBC)

Standard condition 10 compressor blade

- one of many test cases in the turbomachinery literature
- pressure field for an exit Mach number of 0.7



Where to, now?



- http://xkcd.com/844/
- The Compressible Flow CFD Project http://cfcfd.mechmining.uq.edu.au/
- Let's take the best parts and do it again in D.