4

The Toolbox: Special Functions for Dynamic Models

In the process of formulating dynamic models of social systems, certain kinds of relations occur with great frequency. The DYNAMO language has created convenient shorthand versions of a number of such formalisms that speed the process of theory specification. While it is not our purpose here to provide a user's guide to the DYNAMO language, it will be helpful if we briefly introduce some of the most commonly used tools in expressing dynamic relations.[1] This quick overview will both help the reader to understand the models used for illustration in later chapters, and give a sense of the kinds of relations that frequently used in discussing dynamics. Again, the "semimathematical" nature of these functions acts as a bridge between everyday language and mathematics for the theorist—making formalization a more or less normal part of the task of theorizing about the causes of change in social structures.

Describing Relations Among State-Space Elements

A key task in the formalization of any theory is the specification of the forms of the relationships among the elements of the state space. In the early stages of constructing a theory it is sufficient to list the elements of the state space (that is, the "concepts" or "variables") and to map the connectivity of the space—that is, which states are and are not direct causes of change in other states. To say that two variables are connected, or even that one is a cause of the other, however, is hardly sufficient. In order to understand how a variable changes over time, and potentially make predictions about its time course, we must know how variables affect it, not simply that they do.

We have, in fact, a variety of very useful tools for describing the relationships among state space elements. The relations among many state space elements can be rather conveniently expressed using simple mathematical terms: addition, subtraction, division, multiplication, and exponentiation. A large variety of linear and nonlinear relations among quantitative (continuous) states can be captured with such terms. In addition to normal mathematical functions, the DYNAMO language accept trigonometric functions. Thus, for example, one might wish to create a "test pattern" to examine the responsiveness of a model using a sine wave of a certain amplitude and period.

In addition to the normal armament of mathematical and trigonometric functions, the DYNAMO language allows one to "draw" pictures of relationships between continuous variables (and "time," of course is a continuous variable) using a TABLE function. The table function accepts as arguments sets of points (X,Y) that describe any continuous relation, and perform interpolation between the points. It is particularly useful for describing relationships among quantitative variables that are fundamentally nonlinear (e.g., having more than one "bend"), or which have upper or lower limits (as in the case of an S-shaped curve).

For example, an S-shaped relationship between the variable X and time could be expressed like this in DYNAMO:

```
A    X.K = TABLE(XTAB,TIME,0,10,1)
T    XTAB = 0/.5/1.5/3.0/4.0/5.0/6.0/7.0/8.5/9.5/10
```

The first statement says that the value of X is to be defined by a table called XTAB, in which the horizontal dimension is TIME. TIME ranges from 0 to 10, and a value of X will be supplied for each unit increment of TIME. The second statement provides the values of X that correspond to the values of TIME from 0 to 10, respectively.

One reason why many social scientists do not "formalize" their theories is a hesitancy to use mathematical forms to express relations among state variables. Many kinds of relations among states in models of sociological models, however, are fundamentally "qualitative" or logical relations—rather than "quantitative" or mathematical relations. The DYNAMO language (and most others designed for "mixed" or discrete state modeling) provides a useful set of shorthand functions for describing qualitative relations, in addition to the mathematical functions for describing quantitative relations.

The MIN and MAX functions select either the smaller or the larger of a pair of values—providing one mechanism for expressing conditional and limited relationships. For example, the statement:

```
A    Y.K = MAX(0,MIN(X.K,100))
```

first selects the smaller of the current value of X or 100, and then selects the larger of this result or 0 to be the current value of Y. That is, the function sets Y equal to X, except that Y cannot be smaller than 0 nor larger than 100.

The more general form of logical functions is also provided by the CLIP function, which is a simple "if, then, else" type of test (a specialized form the SWITCH statement, tests for arguments being equal to zero). For example, the statement:

```
A      Y.K = CLIP(FUNCT1,FUNCT2,THRESH.K,100)
```

sets the current of value of Y equal to FUNCT1 (which could be a constant, a variable, a conditional statement, or whatever) if the value of THRESH.K is greater than or equal to 100; it sets the current value of Y.K to FUNCT2 if THRESH.K is less than 100.

The clip function is a very useful tool for modeling relationships that change, depending on the values of other processes. In our model of the elements of Marx's theory of capitalism, for example (see Chapter 13), a simple logical test is used to determine whether the strength of the working class exceeds that of the capitalist class. If the workers are stronger, a number of the fundamental relationships in the model "switch": The state is now regarded as controlled by workers, capitalists are eliminated, and profit and exploitation of surplus value are reduced to zero.

It is possible then to use logical functions both to describe the relations among qualitative variables, and to include qualitative change as well as quantitative change in the dynamic theory. This capacity gives much greater flexibility than the language of statistical models which, by and large, are required to have time-invariant coefficients.[2]

A major advantage of such "semimathematical" languages as DYNAMO for formalizing theory lies in the ease and precision with which they allow the statement of hypotheses about the relations among variables. The availability of both logical (qualitative) and mathematical (quantitative) functions make the task of expressing even very complicated ideas about relations among variables rather straightforward. The language requires theory builders be more precise than they might have been had they used everyday language to discuss how variables are connected. This can only be an improvement. Formalized statements about how elements of a state space are connected may be illogical, and they may be inconsistent with empirical evidence, but they are not indeterminate. One of the joys and frustrations of everyday language for describing relations among states is that such statements can be so imprecise as to allow virtually any empirical result to be judged to be consistent (or inconsistent) with the theory.

Semimathematical languages such as DYNAMO provide considerable ease and flexibility in specifying relations. Complex expressions are simplified by breaking them down into sequences of simpler relations: Levels are determined by multiple "rates," each of which is determined by multiple "auxiliaries," each of which, in turn, is determined by other levels, rates, and auxiliaries until, at some point, the circle is closed. The languages allow both qualitative (if-then-else, and/or) relations and quantitative relations (either linear or nonlinear), as well as providing the possibility of creating specialized jargon to express particular forms of relations.[3] The resulting statements about how variables "go together" do not read as well as English—but are far less ambiguous. The statements also tend to be far more intelligible than parallel statements in the language of sets, formal logic, algebra, or differential equations.

Special Functions for Dynamic Analysis: Delays

In formalizing theories about explicitly dynamic relations among variables there is a need for additional specialized vocabulary. The particular nature of theory about dynamics is that we are greatly concerned with processes that occur over time. It follows that we need a vocabulary for describing connections among variables that directly address the "time-shape" of the effect of causal variables on response variables.

The most fundamental "time function" necessary for specifying theories about dynamic relations has already been discussed. It is so well hidden in the basic vocabulary and syntax of DYNAMO and similar languages that the theorist (rightly) has to give them little thought. This "special dynamic function" is integration with respect to time, and is an automatic part of the use of "level" and "rate" equations. "Integration with respect to time" can be thought of as the simplest form of over-time relation. It says, for example, that the amount of change in some (dependent) variable Y over a period of time (e.g., between "J" and "K") is the summing up or accumulation of signals received from some (causal) variable X over the time interval. These "signals" are sent at "rates" that are dependent upon other things (auxiliaries). The "integrating function" that is a part of the syntax of "levels" and "rates," then, is expressing a particular form of relation among states that occurs over time: the continuous accumulation of causal impacts that occur at rates specified by other variables.

In most semimathematical languages for describing dynamic relations, another special function for dealing with the time-shape of effects is provided: the "lag." A "lag" suggests that the "signal" or causal force emitted by a variable at one point in time has an instantaneous effect on the receiving variable after a fixed interval of time has passed. Thus, it might be convenient to specify a dynamic relation between two states as follows: The number of persons who are 25 years old on Nov. 12, 1976 is equal to the number who where 24 years old on Nov. 12, 1975, lagged by one year (assuming no deaths, etc.). Because of its strong bias toward continuous time functions (as opposed to the discrete time function of a lag), the DYNAMO language does not directly and easily provide for the specification of "lags."[4]

The "continuous integration" and "lag" time functions can be thought of as two rather distinctive "time-shapes" of responses to stimuli. In integration, a change in the independent or stimulus variable is responded to in a cumulative and linear fashion until the full impact has been realized; in a lagged response, the dependent variable does not change at all for a specified period, then reaches its full realization instantly. These time shapes are illustrated in the Figure 4.1.

There is no necessity that the time-shapes of the relationships among variables be restricted to these two simple forms. Indeed, some of the most interesting aspects of the over-time behavior of states in models of social action and interaction are consequences of the time-shapes of responses to stimuli. As in modeling static relations a linear approximation is often "good enough" to capture the essentials of a more complex pattern, continuous integration is often "good enough" in dynamic models for rough and ready theory exploration.

The DYNAMO language provides shorthand functions for two other common time shapes: first- and third-order exponential delays. A first-order delay shows immediate response to a stimulus, with exponential decline thereafter. A third-order delay displays an initial "lag" period of little response, followed by a rapid increase and a slow decline.[5] These time shapes are shown in Figure 4.2:

Four particular functions are provided in the DYNAMO language for first- and third-order delays of material and informational quantities: DELAY1, DELAY3, SMOOTH, and DLINF3. Material delays conserve the quantity "in transit" in a delay if the length of the delay changes; informational delays do not conserve the quantities in the delay. This is of consequence only if the average length of the delay is specified to be a variable (for example, the length of the delay in information reaching the top of an organization changes proportional to the number of levels in the organization).

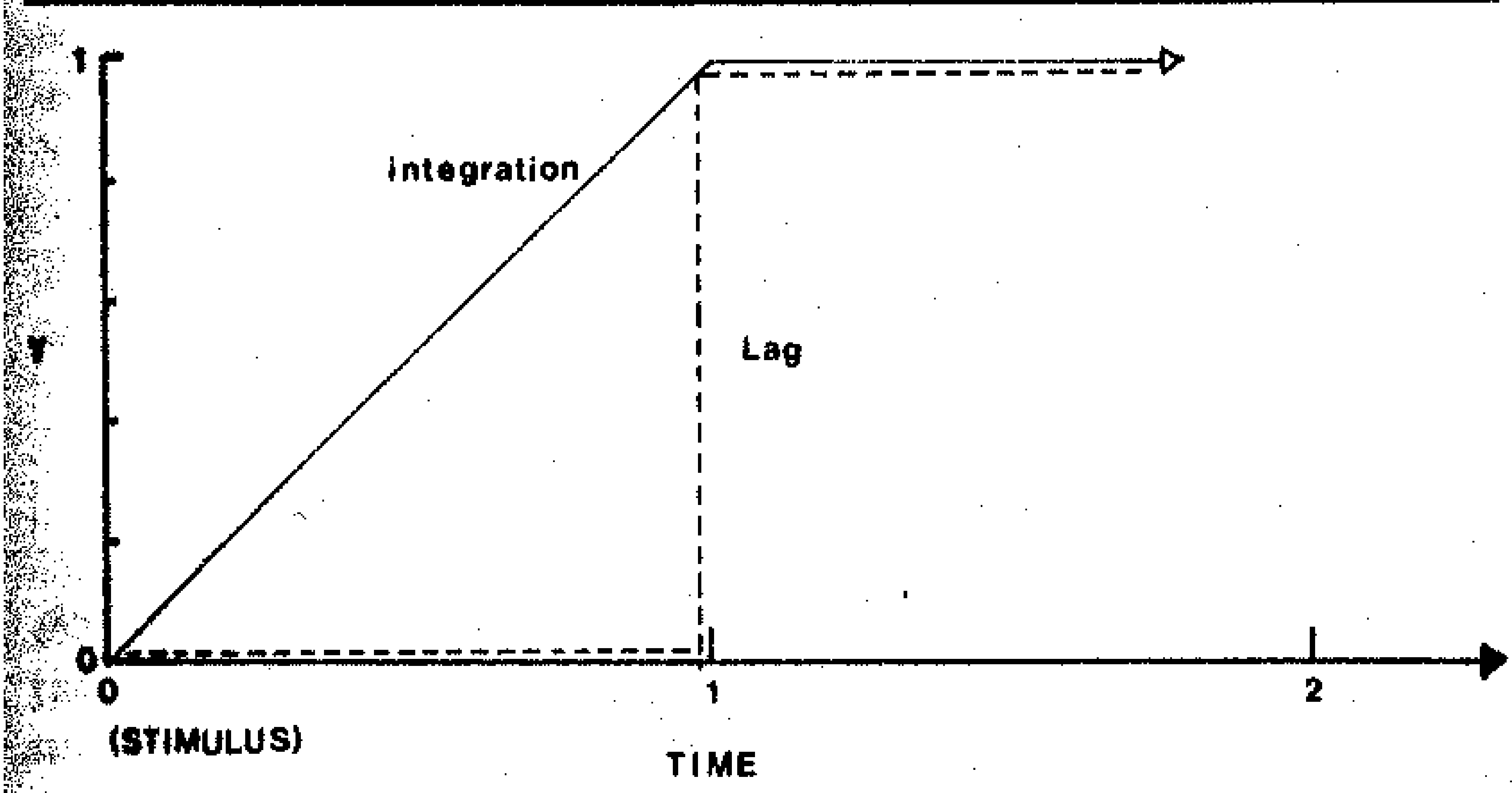

Figure 4.1: Integrating and lag response to unit impulse.

Delays can be used in describing relations among material or informational quantities, and have a quite simple syntax, for example the statement:

A Y.K = DELAY3(X.K,5)

says that the value of Y at time K is responsive to the value of X by a time shape of a third-order exponential delay (that is, S-shaped), with an average delay time of five units. Thus, both the form and the magnitude of the delay are specified in a single statement.

In practice, the distinction between material and informational delays is seldom of great consequence; the choice of the time-shape of response (that is, first, third, infinite, or some other order) can be extremely consequential. Theorists have rarely given attention to the question of the time-shape of the relations among variables because they have been primarily concerned with static or equilibrium analysis. Again, the DYNAMO language helps, requiring that the theory builder ask him/herself specifying questions about the nature of the dynamic relations among the states in the theory: Given that a change in X causes a change in Y, how is this stimulus realized in a response over time? How long does it take for the full effects of each change in X to be realized in responses in Y (that is, what is the period of the delay)? Does Y begin responding immediately and then approach its full realization asymptotically (as in a first-order delay)? Is the response initially low, but increasing at a decreasing rate thereafter (as in a second-order delay)? Is initial response low, but then accelerating and finally decelerating (as in

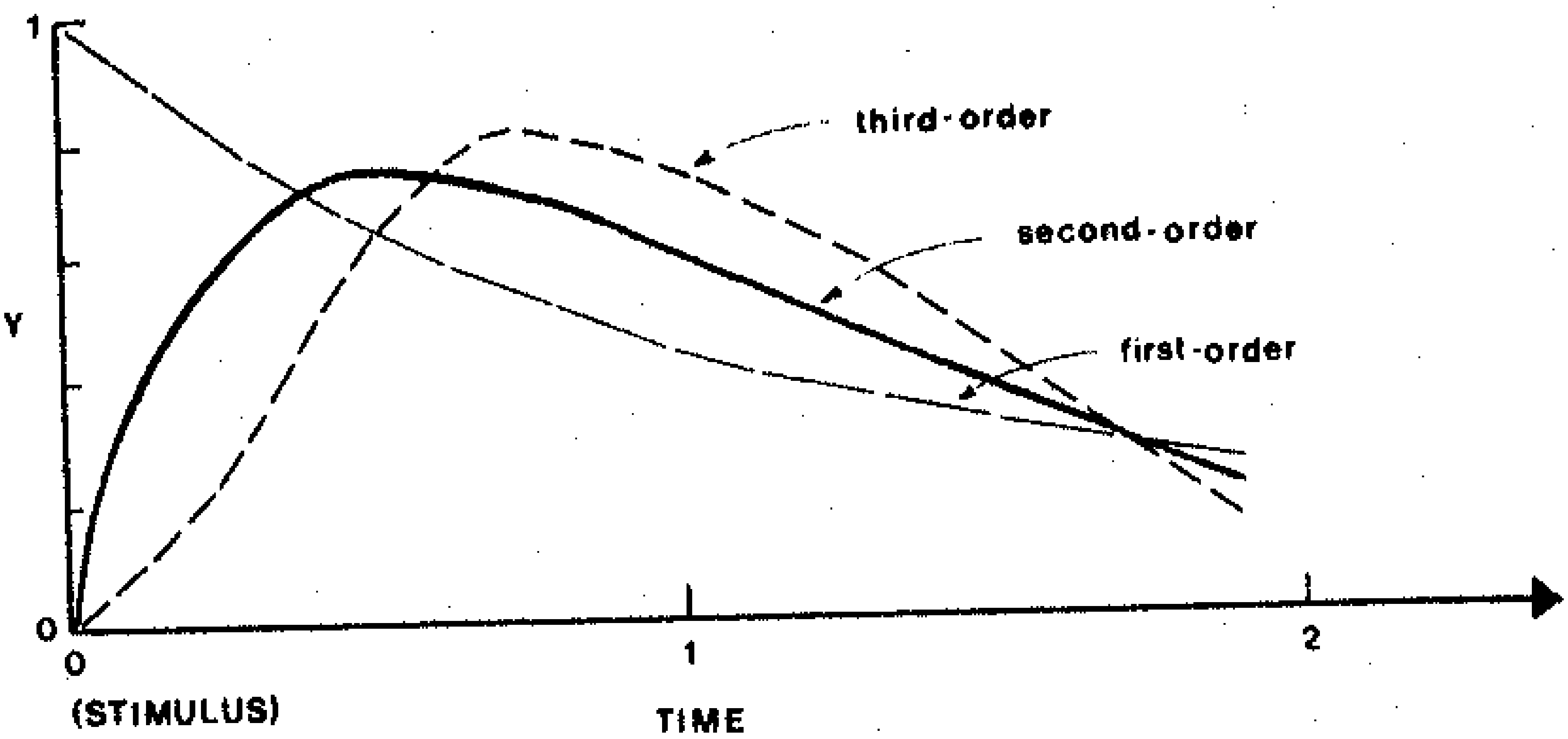

Figure 4.2: Delayed responses to unit impulse.

a third-order delay)? Is there no response at all for a fixed period, and then an instantaneous adjustment (as in a lag or high-order exponential delay)?

In theorizing about dynamics, we must specify not only the functional form of relations among variables (i.e., linear additive, if-then, or whatever) but also the time-shape of the relationship. There are many possible time shapes that can be specified and analyzed using mathematical formalisms, but the vocabularies of time relations in mathematics are very difficult. On the other hand, the everyday language of the theorist is often insufficiently specific in describing the time-shapes of responses. Vagueness will not do, for, as we shall see in later chapters, the time-shapes of responses have very important impacts on over-time behavior. The language of "delays" in DYNAMO is one useful shorthand for specifying the most common forms of time relations.

Ghosts Within the Machine: Noise

Formalizations of theories utilizing statistical terminology (e.g., the general linear model) routinely include "residual" or "error" terms as independent variables. In assessing how well such models can account for observations, we frequently find that the residual term is by far the largest of the variance components. In constructing models of dynamic relations with semimathematical languages, it is also useful to consider the role played by "noise," "error," and "residual sources" The DYNAMO language (and other similar languages) provide shorthand tools for this purpose. Theories that are stated in logical deductive terms have similar caveats. When we say that a change in X produces a change in Y, all else being equal, we are admitting that the relationship between X and Y will not always be realized perfectly.

Like delay, noise plays a very important role in dynamic relations. This role is somewhat different than that of the "residual" in static and nonrecursive models. Consider the static model:

$$Y = a + bX + e$$

This "sentence" says that the score on Y at a given point in time is equal to some constant amount (a), plus some amount that is a function (b) of the score on X at the same time point, plus some random error (usually with an expected value of zero, and a normal distribution). The "error" component here is conceptually simple: It represents a separate source of variations in the scores across realizations (observations) of Y.

Things change a bit when we consider a dynamic (in this case lagged and conserved) relation:

$$Y_{t+1} = a + b_1 (Y_t) + b_2 (X_t) + e$$

That is, the score on Y at the second time point is a linear additive function of a constant, the score at the previous point in time (conservation), the score on an independent variable (X) at the previous point in time, and "error." Now, at each point in time the actual value of the state Y is composed of all of these components. As a result, the score on Y at each later time point is a function of past "errors" as well as past scores on Y and X. If "errors" are truly random with a mean value of zero, this is not a problem for understanding why Y changes the way it does over time. But the particular value of Y at each point in time is somewhat indeterminate due to both current and past errors. That is, "errors" remain in the system over time and may cause the time-track of Y to differ from what we would have expected had there been no error.[6]

If we made our simple statistical model fully recursive (that is, with X causing changes in Y and Y causing changes in X), the role that error plays is still more complex. In that case (and we won't burden you with the equations), the random-error component of each variable at each point in time is "multiplied" through the system, so that scores on both X and Y at later points in time are a function of earlier scores on X and Y and earlier errors in X and Y.

Most of the systems that we will be considering in the later chapters of this volume represent patterns of dynamic relations among states that are at least as complex as this last case of statistical models. The role that noise and error play in systems with nonlinear and feedback relations is often very difficult to deduce directly. Consequently, the normal strategy for dealing with error is to ignore it in the early stages of model formulation, and then add it to the model in a controlled and systematic manner to understand its consequences (more on such simulation approaches to analyzing complex theories below). The DYNAMO language (as do most others) makes this a relatively simple process by providing two functions that generate pseudorandom noise. The NOISE function is a simple white noise generator (that is, it produces random numbers); the NORMRN function generates normally distributed random numbers with a mean and standard deviation selected by the analyst. With some cleverness, other noise patterns can be created if they are needed. For example, one might specify:

```
L     Y.K = (PARM1*Y.J)+(PARM2*X.J)+NORMRN(0,10)
```

That is, the level of Y at time K is equal to the autoregressive effect (PARM1) of its value at the previous point in time, plus the lagged effect (PARM2) of the value of X at the previous time, plus an error component of normally distributed noise having a mean of 0 and a standard deviation of 10 units.

Ghosts Without the Machine: Test Functions

The purpose of constructing formal theories of dynamic processes is to gain an understanding of the dynamic consequences of the relations among the elements of the state space. In models involving large state spaces, with high connectivity and nonlinear, delayed, noisy, and feedback relations among the states, the dynamic behavior of the theory can be very difficult to deduce directly. In such complex cases, understanding and analysis of the theory can best be accomplished by simulation. We will deal with this approach at length in the next chapter.

The last group of shorthand functions provided by most semimathematical languages for formalizing continuous state continuous time dynamics are "test patterns" used (primarily) in simulating the over-time consequences of the theory. In understanding by simulating, the basic procedure is to subject the formal model of the state space and relations among state space elements to "shocks," and to observe the consequences. DYNAMO provides some useful tools for creating such shocks.

It is frequently useful to subject models to shocks that are composed of "white noise," random noise, constants, or simple mathematical functions such as sine waves. We have already considered the vocabulary necessary to create these tests. In addition, three other test patterns are particularly useful: PULSEs, STEPs, and RAMPs.

PULSEs consist of instantaneous shocks of specified magnitude and timing, and are useful for assessing the "transient response" of models (more on transient response in the next chapter). For example, the statement:

```
A     TEST.K = PULSE(10/DT,5,15)
```

creates a test signal of a total magnitude of 10 units (it is necessary to divide the desired magnitude of the shock by the size of the integration interval "DT" to take the fineness of the integration into account; alternatively one could leave the first parameter as a raw number, which would set the value of the "peak" of the shock, but not its total magnitude, which would vary with the size of DT). This shock occurs for the first time at TIME = 5 and occurs every 15 time units thereafter.

We can often learn a good bit about the behavior of our theory by subjecting it to sudden discrete changes in exogenous stimuli by use of STEP functions, or to stimuli that are continuously increasing or decreasing over time, by use of RAMP functions.

The statement

```
A     TEST.K = STEP(10,5)
```

for example, creates a test signal that has a value of 0 (by default) until time point 5, and 10 units thereafter. The statement:

```
A     TEST.K = RAMP(5,10)
```

creates a test signal that is 0 (by default) for the first 10 time points and increases by 5 units for each unit of time thereafter.

Virtually any pattern of stimuli can be generated by clever use of the PULSE, STEP, and RAMP functions. In addition to using such signals for sensitivity experiments, such shocks can be important parts of models in other ways. Exogenous shock patterns can be used to model changes in exogenous stimuli (the model in Chapter 10 does this), and can be used to schedule the occurrence of exogenous events or unusual changes in relations among endogenous variables that are time dependent.

Conclusions

The purpose of this brief chapter has been to introduce some of the vocabulary that can be of assistance in formalizing and assessing dynamic relations. We have discussed a set of mathematical and logical functions that are generally useful for describing relations among state space elements. These functions are quite familiar. We have also described a number of specialized functions that are less familiar, and particularly useful for describing the forms of the "time shapes" of relations among state space elements.

One major and very important group of functions are DELAYs, which can be used to describe how long it takes a change in one state to be realized in a response in another state, and the form that the response takes with respect to time. Responses can be constant with respect to time (by use of the integration function DT), may begin sharply and then decline in intensity (for example, first-order exponential delays), may begin slowly and then accelerate (as in third-order exponential delays), or may be characterized by discrete lags (very high-order delays, or "boxcars"). Theorists have rarely given much attention to the length of time that it takes for causal processes to occur, or to the time shapes of the processes. In dynamic models with feedback, however, these aspects of the relations among variables can be extremely important. Delays allow one fairly easy way of specifying some of the most common patterns.

The other group of time functions that we have examined here are most commonly used in testing models by subjecting them to shocks. PULSEs, RAMPs, and STEPs are particularly useful for generating test patterns for this purpose, though they can also be used to describe (rather unusual) forms of relationships among variables. We have been deliberately a bit vague about how these test functions are used in the process of formulating and testing models of dynamic theories. As we consider the general methodology of utilizing computer assisted simulation in theory building, in the next chapter, their use will become much clearer.

Notes

1. There are a number of excellent sources that are very helpful in learning the basics of the computer language DYNAMO. To work effectively with DYNAMO, readers will also have to become more familiar with the specifics of the program and its use. The most useful introductions are given by Goodman (1974), Pugh (1980), Pugh-Roberts Associates (1982), and Roberts et al. (1983).

2. The DYNAMO language, while it has many useful functions, particularly for describing nonlinear and over-time relations, is rather inelegant with regard to discrete lags and certain logical functions. Particularly annoying are the absence of the "AND" and "OR" functions and the rather peculiar form of the "IF-THEN-ELSE" operation.

3. The macro facility available in most versions of DYNAMO allows the user to create specialized functions of any complexity that can be called upon as specialized "jargon" in model construction.

4. Such a function, however, can be created by use of a macro. In some versions of DYNAMO a specialized "boxcar" or "pipeline" delay can be used to capture discrete delays with some effectiveness. DYNAMO is not particularly friendly to discrete time functions, which are extremely important in large classes of dynamic models; see, for example, Takacs (1962).

5. Integration can also be thought of as a "zero-order" delay; discrete lags can also be thought of as infinite-order delays.

6. Dealing with the complexities of such over-time error processes and their consequences for the statistical estimation of parameters is the "bread and butter" of econometrics. Any of a very large number of econometrics texts can provide an excellent introduction to error processes and their effects on parameter estimation.