Permission to use Chapter 17 from "Introduction to Computer Simulation, A System Dynamics Modeling Approach" in the MIT System Dynamics Group Road Maps series, Chapter 5 has been given by Professor Nancy Roberts, October, 2001. The permission is only for non-profit, educational purposes.

CHAPTER 17

INTRODUCTION TO DELAYS*

Delays occur frequently in social and economic systems. When a business organization orders supplies, the supplies usually arrive only after a delay. When a pollutant is dumped into a river, it takes time to dissipate. When the price of gasoline rises, consumers take time to adjust by driving less or by purchasing more fuel-efficient cars. And, of course, when you mail a letter, the letter will be delivered only after a delay.

Delays are conveniently divided into two types: delays resulting from the time involved in processing physical materials and delays resulting from the time involved in perceiving and acting upon information. As these two types of delays—called *material delays* and *information delays*—abound in social and economic systems, some of their properties are investigated in this chapter.

EXAMPLE I: THE MARTAN CHEMICAL COMPANY

Recall from Example I in Chapter 7 that the Martan Chemical Company, which manufactures the pesticide Nobug, dumps a quantity of Nobug into the Sparkill River once a week. During the course of the week, the pollutant is absorbed by the river's natural clean-up processes. A causal-loop and flow diagram of the Martan case are shown in Figure 17.1.

The equations for the model are similar to the equations for the yeast deaths system, discussed in Chapters 13 through 15.

```
L    NOBUG.K=NOBUG.J+(DT)(DUMP.JK-ABSORB.JK)
N    NOBUG=NOBUGN
C    NOBUGN=0
R    ABSORB.KL=NOBUG.K/NAT
C    NAT=2
```

*Students wishing immediately to try modeling a more complete problem situation might do Chapters 18 and 19 before this chapter. However, the contents of this chapter are critical for the models contained in Chapter 20 and beyond.

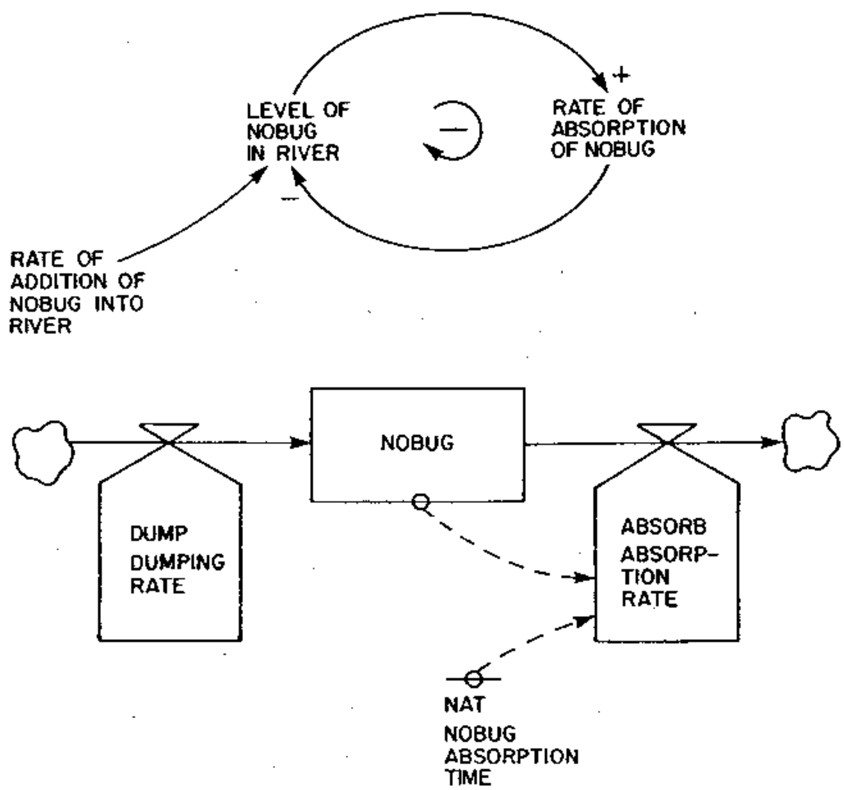

Figure 17.1 Diagram of the Martan Case

These equations indicate that the level of NOBUG in the Sparkill River is influenced by the dumping rate DUMP and the absorption rate ABSORB; and the absorption rate ABSORB is equal to the level of NOBUG divided by the Nobug Absorption Time NAT (2 days).

The only equation that remains to be specified is the dumping rate DUMP. According to the description in Chapter 7, the Martan Company releases Nobug into the river in once-a-week batches, producing a Nobug concentration in the river of about 420 parts per million (ppm). Assuming that the river contains 1 million gallons of water, this amounts to a dumping rate of 420 gallons each week.

For modeling purposes it is easier to begin by assuming that Martan dumps Nobug into the river continuously, at an overall rate of 420 gallons per week. This amounts to a continuous daily dumping rate of $420/7 = 60$ gallons per day.

Exercise 1: Preliminary Nobug Model

a. Write DYNAMO equations for the Nobug case, adding the DUMP equation and other needed specifications. Run the model, setting the initial level of NOBUG = 0, and choosing DT = 0.25 days. What behavior does the model generate?

b. Rerun the model, setting the Nobug Absorption Time NAT = 4. How do the results differ? Rerun the model, setting NAT = 1. How do these results differ?

USING THE PULSE FUNCTION TO REPRESENT THE DUMPING RATE

The model developed so far is somewhat inadequate, because it assumes that Nobug is continuously released into the Sparkill River at a rate of 60 gallons per day. The DYNAMO PULSE function permits modifying the model to represent the dumping of Nobug in once a week batches. The following equation indicates that 420 gallons of Nobug are dumped into the river on day 1 of the simulation, and 420 gallons are dumped again at regular intervals of 7 days.

R DUMP.KL = (1/DT)*PULSE(420,1,7)

Figure 17.2 shows the dumping rate over the first 10 days of the simulation. On day one, the dumping rate rises to 1680 gallons per day for the duration of a time interval of one DT (i.e., during one quarter-day). It then falls to zero, and remains there until day 8, when it again rises to 1680 for a time interval of one DT. It will rise again on day 15, and then once again fall to zero.

One aspect of the equation for the dumping rate may seem puzzling at first glance. Why is the factor (1/DT) included in the formulation? On the surface this may seem odd, since it produces a dumping rate of 1680 gallons per day rather than 420. To understand the use of (1/DT) in the dumping equation, it is necessary to take a closer look at the level equation for NOBUG.

L NOBUG.K = NOBUG.J + (DT)(DUMP.JK − ABSORB.JK)

Note that in the level equation for NOBUG, the dumping rate DUMP is multiplied by DT to produce the amount of NOBUG added to the river during one time interval DT. Thus during the first time interval of one day, the amount dumped is equal to DT*1680 = 0.25*1680 = 420 gallons. In general, the amount dumped during any time interval equals

(DT)*(1/DT)*PULSE(420,1,7) = PULSE(420,1,7)

The factor (1/DT) is necessary in the PULSE rate equation to cancel the factor DT included in the level equation. If the dumping rate that takes place during the first quarter of day one were maintained *for the entire day,* 1680 gallons of NOBUG would be dumped into the river, but the rate is not main-

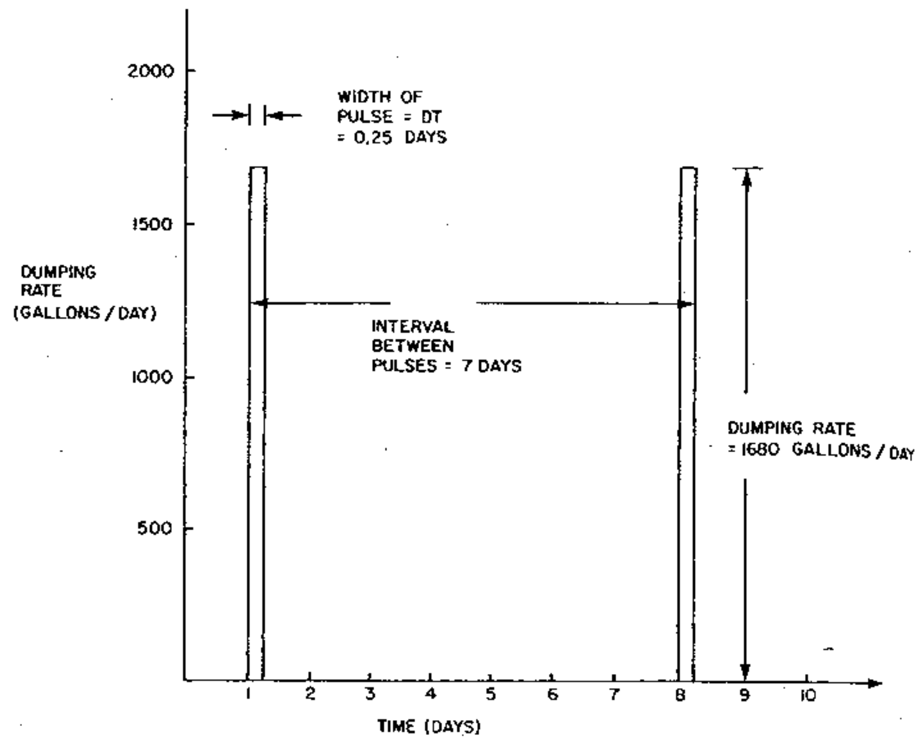

Figure 17.2 Dumping rate

tained for the entire day. It continues for just one time period DT and then it falls to zero. Thus, the *total amount dumped* is 420 gallons—exactly what the formulation is supposed to produce.

The PULSE function can also be used to examine the response of the system to the dumping of just one batch of NOBUG. It is possible to do this by making the following change in the equation for DUMP.

R DUMP.KL = (1/DT)*PULSE(420,1,1000)

This equation indicates that the dumping rate rises from zero to 420/DT or 1680 on day one and every 1000 days thereafter (rather than every 7 days). Thus if the model is run for a period shorter than 1000 days, only one pulse in the dumping rate will occur.

Figure 17.3 shows a simulation of the Nobug system, with one batch of Nobug released into the river on day one. As can be seen, the level of NOBUG in the river rises sharply to 420 gallons on day one, and then drifts slowly toward zero.

The general form of the PULSE function is:

PULSE(AMOUNT,FIRST,INTERVAL)

```
       NOBUG=N,ABSORB=A,DUMP=D

      0.00      150.00      300.00      450.00      600.00  NA
      0.0        600.0      1200.0      1800.0      2400.0  D
0.0000 N------------------------------------------------ NAD
       N           .           .           .           . NAD
       N           .           .           .           . NAD
       N           .           .           .           . NAD
       N           .           .         D .           . NA
       D           .    A      .         N .           .
       D           .  A        .    N      .           .
       D           .A          . N         .           .
       D          A.          N.           .           .
       D         A .       N   .           .           .
2.5000 D--------A-------N-------------------------------
       D       A   .  N        .           .           .
       D      A    .N          .           .           .
       D     A     N           .           .           .
       D    A    N .           .           .           .
       D   A    N  .           .           .           .
       D   A   N   .           .           .           .
       D  A   N    .           .           .           .
       D  A  N     .           .           .           .
       D  A N      .           .           .           .
5.0000 D-A--N-------------------------------------------
       D A N       .           .           .           .
       D AN        .           .           .           .
       D AN        .           .           .           .
       DA N        .           .           .           .
       DAN         .           .           .           .
       DAN         .           .           .           .
       DAN         .           .           .           .
       DAN         .           .           .           .
       DN          .           .           .           . NA
7.5000 DN----------------------------------------------- NA
       DN          .           .           .           . NA
       AN          .           .           .           . AD
       AN          .           .           .           . AD
       AN          .           .           .           . AD
       AN          .           .           .           . AD
       AN          .           .           .           . AD
       N           .           .           .           . NAD
       N           .           .           .           . NAD
       N           .           .           .           . NAD
10.000 N------------------------------------------------ NAD
```

Figure 17.3 Release of one batch of NOBUG

where AMOUNT indicates the amount to be inputted in the pulse; FIRST indicates the time at which the first pulse occurs; and INTERVAL indicates the time interval between pulses.

Exercise 2: The Halving Time for NOBUG

Revise your DYNAMO equations to include a PULSE function for the dumping rate. Choose a time between pulses of 1000 days, in order to examine the effects of just one pulse.

a. What is the halving time for the amount of NOBUG in the Sparkill River?

b. Experiment with various values of NAT. How does the choice of NAT influence the halving time?

c. Using your results for part (b), select a value of NAT to produce a halving time that corresponds to the data shown in Chapter 7, Figure 7.1.

Exercise 3: Simulating Repeated Batches

a. Select NAT equal to the value you determined in Exercise 2, part (c). Use the PULSE function to simulate the effect of dumping 420 gallons of Nobug into the river at 7-day intervals. Compare your results with the data shown in Chapter 7, Figure 7.1.

b. Reread Example II in Chapter 7, "Martan Chemical—Part II". According to that example, Martan Chemical changed the chemical composition of Nobug, resulting in an increase in the Nobug absorption time NAT. Change the value of NAT in your model to reflect the change in the chemical composition of Nobug. How does the system respond? Compare your results with the data shown in Figure 7.3, Chapter 7.

NOBUG AND MATERIAL DELAYS

In the Nobug case, Nobug is dumped into the river in once-a-week batches. The absorption of Nobug does not occur immediately, however. Instead, it occurs over a period of time. In fact, the absorption of Nobug by the river can be viewed as a *time delay* process. The absorption rate is the river's delayed response to the dumping rate.

The Nobug model structure is called a *first-order material delay*, because it describes the flow of a material substance into and out of a single level. Figure 17.4 shows a general flow diagram for a first-order material delay,

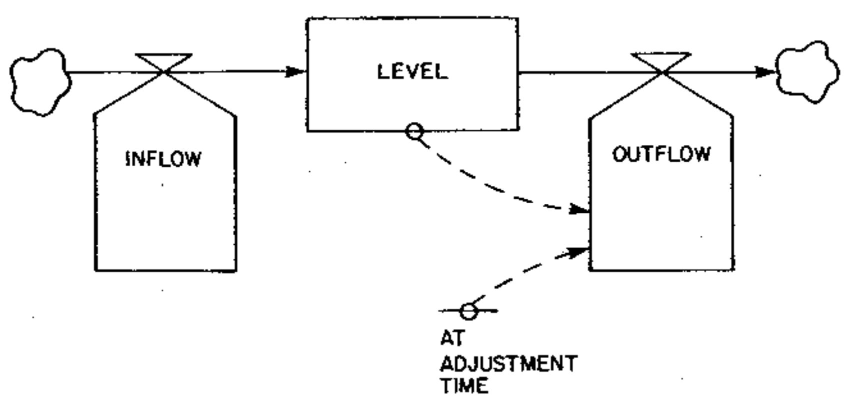

Figure 17.4 First-order material delay

along with the corresponding equations. The general form of a first order material delay is exactly analogous to the Nobug structure. An inflow rate accumulates in a level; and an outflow rate is equal to the amount in the level, divided by a time constant for adjustment of the level.

Because first-order material delays are widely used in system dynamics models, they are often given a special flow diagram notation. Figure 17.5 shows the usual flow diagram symbol for a first order material delay, and Figure 17.6 shows how the symbol can be used to represent the Nobug case.

When a first-order material delay is used in a model, there are two ways to write the equations. One approach is simply to write out individual equations, exactly as was done in the Nobug case. But because first-order material delays are frequently used, a special DYNAMO function is available that can be used to replace the set of individual equations. The following equation can be used to indicate that the OUTFLW rate is a first-order delayed response to the INFLOW rate, with an adjustment time AT.

R OUTFLW.KL = DELAY1(INFLOW.JK,AT)

The DELAY1 function is simply a shorthand notation. When the model is run, DYNAMO will substitute the full level and rate formulation for a first-order delay, whenever the DELAY1 function appears in the model.

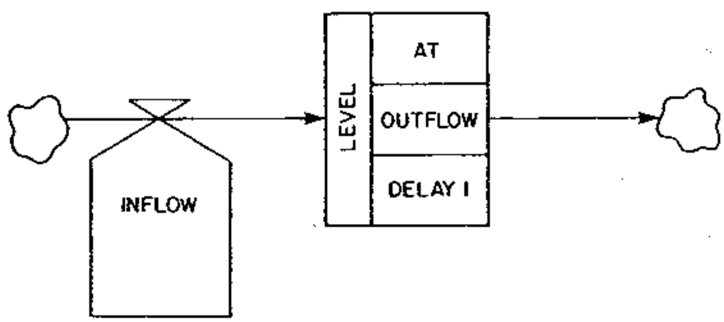

Figure 17.5 First-order delay symbol

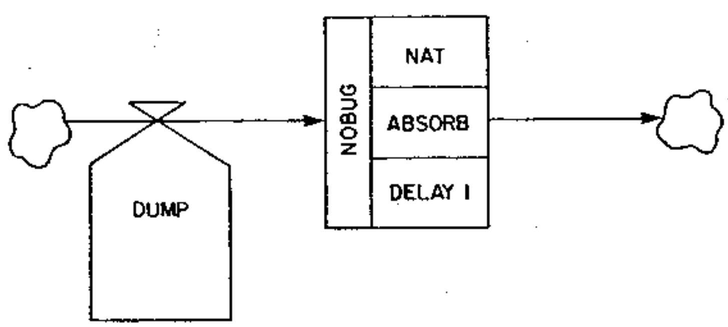

Figure 17.6 Delay for NOBUG

The DELAY1 function can be used quite easily to represent the Nobug case. The full set of equations for the Nobug case becomes:

```
R    ABSORB.KL=DELAY1(DUMP.JK,NAT)
C    NAT=2
R    DUMP.KL=(1/DT)*PULSE(420,1,7)
```

This indicates that the absorption rate is a delayed response to the dumping rate, with an absorption time NAT = 2.[1] Note that in this formulation the NOBUG level equation is not needed, and is also not available for use elsewhere, such as output printing or plotting.

Exercise 4: Nobug and the DELAY1 Function

a. Modify your equations for the Nobug model, using the DYNAMO DELAY1 function to replace the level of NOBUG and the Nobug absorption rate.

b. Run the model. It should behave exactly as it did in the previous Exercises 2 and 3.

c. Experiment with various values of NAT. How does the model respond?

Exercise 5: The Mail Delay

The Nifty Department Store sends out bills to its charge-card customers once a month, and the credit department has learned that, on the average, it takes about three days for the bills to arrive in the mail.

a. Draw a causal-loop diagram, flow diagram, and equations for the Nifty Department Store case. (*Hint:* See Figure 7.5 in Chapter 7.) Assume that NIFTY has 1000 charge customers.

b. Run the model and examine the results.

c. How many bills take more than six days to arrive?

EXAMPLE II: THE GOAL-GAP FORMULATION, DELAYS, AND CYCLES IN APARTMENTS

Seemingly simple models can often generate surprising behavior when a delay is introduced. One interesting example is the construction of apartment buildings in a large city. Suppose builders construct apartments in response to the gap between the total number of apartments desired by people in the city and the total number of apartments available. A causal-loop diagram and flow diagram for this system are shown in Figure 17.7. The system described by the flow diagram is exactly analogous to the coffee cooling system in Chapter 15.

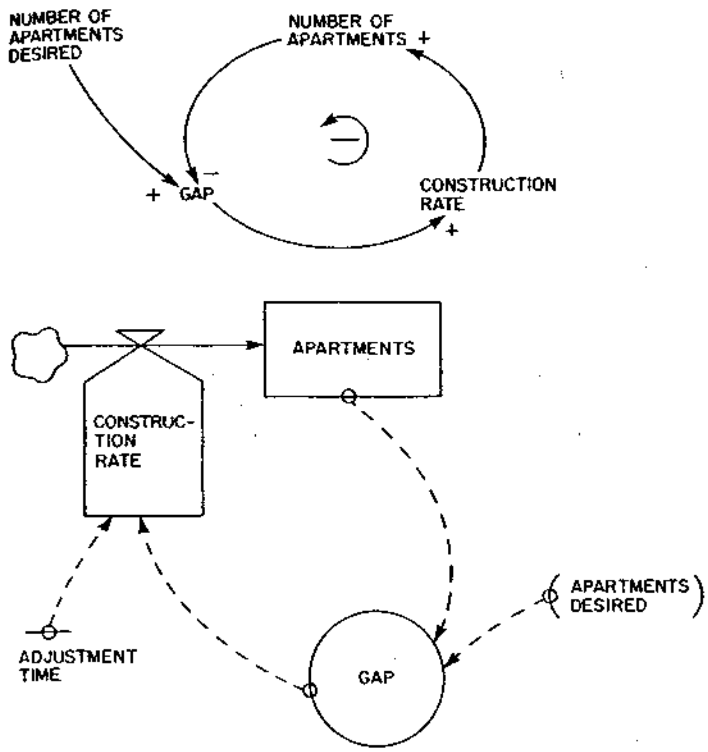

Figure 17.7 Diagrams expressing a housing gap

Exercise 6: Apartments—Part I

a. Write DYNAMO equations for the apartment model shown in Figure 17.7. Assume that the desired number of apartments is 10,000 and the time required to respond to the gap is one year. (Do not include an explicit construction delay in your formulation—it will be added in the next exercise.)

b. Run the model and examine the behavior. Determine the equilibrium value for the number of apartments.

c. Start the model in equilibrium, and use a STEP function to test the response of the system to an increase in the number of apartments desired from 10,000 to 15,000.

The apartment model developed so far has ignored an important delay: it takes time to construct apartments. Once an apartment builder makes the decision to build a new apartment house, it may take roughly four years to purchase appropriate land, obtain building permits, complete architectural drawings, and build the apartments.

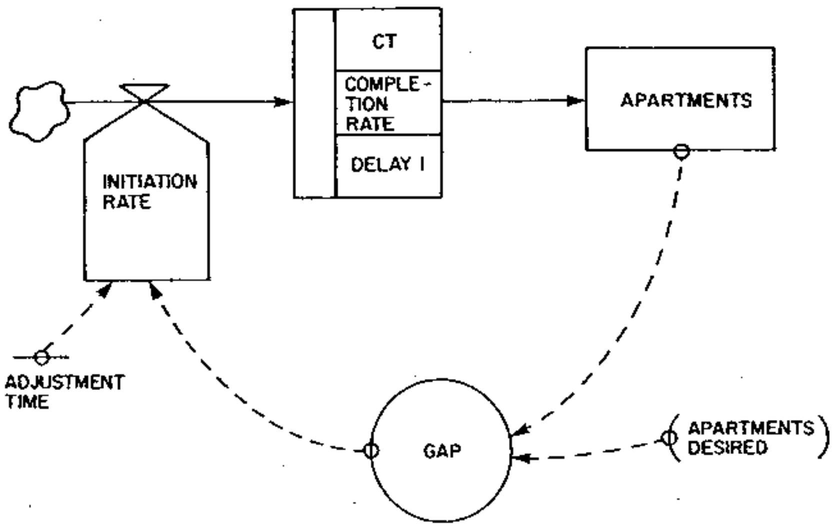

Figure 17.8 Apartment model with completion delay

This suggests that a delay should be added to the model. One way to do this is shown in Figure 17.8. The diagram indicates that the apartment completion rate is a delayed response to the apartment initiation rate. A delay exists between the times apartments are initiated and completed.

Exercise 7: Apartments—Part II

a. Modify your model of apartment construction to include a first-order material delay in the construction completion rate.

b. Start the model in equilibrium.

c. Examine the response of the system to a STEP increase in desired apartments from 10,000 to 15,000. What behavior does the system generate?

d. What do you think accounts for this behavior?

e. Experiment with the model, choosing various values for the time required to construct apartments. What happens when this value is made longer than 4 years? What happens when it is made shorter than 4 years?

f. At times, the apartment construction rate may become negative. What does this mean? Is it realistic? Why or why not? Predict how the system would behave for a period twice as long as your run. Run the model to check your prediction. If your prediction does not fit the run, try to formulate a new prediction for the long-term behavior.

INFORMATION DELAYS

Delays are frequently involved in transmitting and acting on information. One common information delay occurs in the labor market. It takes time for information about the availability of jobs to reach people who are seeking them. Similarly, information delays occur in organizational management. For example, the information is delayed from the time sales occur in a department store until the manager learns of the sales and uses the information to reorder depleted stock.

Information delays also occur in individual and organizational decision-making for another, more subtle reason. Often, individuals and organizations make decisions on the basis of information that has been *averaged*, and averaging implicitly involves delays. For example, data on U.S. unemployment are often presented on an annual basis, which is generally an average of the percent of the work-force that has been unemployed over the last twelve months. Thus a rise in current monthly unemployment must persist for several months before it has a large impact on the average annual unemployment figure.

Decisions based on averaged information are widespread. For example, upon filling up the tank with gasoline, the driver notices that her car has not traveled as far on a tank-full as it usually does. She is unlikely to immediately conclude that something is the matter with the car. Instead, she probably waits to see what happens on the next few fill-ups before sending the car to the shop. Thus the car-owner has informally computed an average. Similarly, the owner of a baseball team does not fire the manager when he loses the first game. The owner usually waits to see what the balance of wins and losses looks like over the longer run.

While it is certain that delays and information averaging are frequently involved in individual and organizational decision-making, it is less clear how to represent these processes in a model. After all, some business organizations use complex information averaging formulas in making decisions, while individuals often simply weigh whatever information is available in an informal, intuitive manner. One simple formulation that might be used to represent the delays involved in perceiving and acting on information is the moving average, which is considered in the following example.

EXAMPLE III: MARINA'S BAKE SHOP

Marina's Bake Shop bakes and sells sourdough French bread Monday through Friday. Each morning Marina has to decide how many loaves to bake, and she relies on the average sales over the past five days in making her decision. She collects the daily sales from the previous five days, adds them together, and divides by five.

Sales data for Marina's Bake Shop for a twenty-five-day period are shown in Table 17.1. Since it takes five days of data to compute an average, the first

Table 17.1 Marina's sales data and moving average sales

Day	*Moving average sales*	*Sales*
1	—	94
2	—	116
3	—	87
4	—	104
5	—	107
6	101.6	90
7	100.8	102
8		96
9		108
10		121
11		123
12		130
13		135
14		113
15		117
16		117
17		128
18		97
19		109
20		123
21		116
22		117
23		121
24		128
25		122

morning on which average sales can be computed is day six. The average on day six is just the sum of the first five days' sales, divided by five. The average for day seven is the sum of the sales for days two through six, divided by five; and so on.

Another way to state the formula for the moving average is:

$$\text{AVERAGE SALES(today)} = \text{AVERAGE SALES(yesterday)} + \text{SALES(yesterday)}*(1/5) - \text{SALES(six days ago)}*(1/5)$$

This simply means that to compute the average sales, take the value of the average sales computed yesterday, subtract the portion of the average contributed by the sales six days ago, and add on the portion of the average contributed by the most recent day.

Exercise 8: Computing the Moving Average

Calculate the moving average for Marina's sales for days 8–25, based on the data in Table 17.1.

The moving average is fairly simple, conceptually, and easy to calculate by hand. Unfortunately, however, to represent it precisely in a simulation model requires that data be maintained on each past event included in an average. The longer the period of averaging, the more data to be retained. Furthermore, the moving average has the additional defect that all data over the time period of the average are weighed equally. However, when individuals and organizations make decisions, they tend often to rely more heavily on recent information, and less heavily on older data.

A second type of average, called the exponential average, resolves this latter problem by explicitly weighing recent information more heavily. Suppose Marina, on the morning of day two, wants to compute average sales using an exponential average. To compute the average, she has only one piece of information: sales on day one. Thus Marina sets the average on day two equal to the sales on day one:

AVERAGE SALES(day two) = SALES(day one)

If Marina wants to compute the average sales on the morning of day three, she has one more piece of information. She now has the average sales she computed on day two, along with the new sales figure for day two. To combine these two pieces of information to produce a five-day exponential average, Marina should use the following formula:

AVERAGE SALES(day three) = (4/5)*AVERAGE SALES(day two)
+ (1/5)*SALES(day two)

The new sales figure should be weighted by a factor of one-fifth, and the previous average should be weighted by a factor of four-fifths.

Similarly, to calculate the average sales on the morning of day four, Marina should use the formula:

AVERAGE SALES(day four) = (4/5)*AVERAGE SALES(day three)
+ (1/5)SALES(day three)

Once again, the new sales figure is weighted by a factor of one-fifth, and the previous average is weighted by a factor of four-fifths.

The formula for a five-day exponential average of sales in Marina's Bake Shop is:

AVERAGE SALES(today) = (4/5)*AVERAGE SALES(yesterday)
+ (1/5)*SALES(yesterday)

Hence, in computing today's average, yesterday's average is given a weight of four-fifths, and yesterday's new sales figure is given a weight of one-fifth. Implicitly, this procedure weights recent data more heavily than old data. By the twenty-first day, for example, the twentieth day's sales figure is given a weight of one-fifth; and all nineteen previous days' sales are given a total weight of only four-fifths.[2] Table 17.2 shows the exponential averages calculated for the first several days.

Exercise 9: Computing the Exponential Average

a. Calculate the exponential average for Marina's sales for days five through twenty-five, based on the data in Table 17.2.

b. Compare the exponential average and the moving average you obtained for each day. In what ways are they similar? How do they differ?

Table 17.2 Marina's exponentially averaged sales

Day	*Exponential average sales*	*Sales*
1	—	94
2	94.0	116
3	98.4	87
4	96.1	104
5		107
6		90
7		102
8		96
9		108
10		121
11		123
12		130
13		135
14		113
15		117
16		117
17		128
18		97
19		109
20		123
21		116
22		117
23		121
24		128
25		122

AVERAGING AS A FIRST-ORDER DELAY

The formula used to calculate the exponential average of the sales in Marina's Bake Shop can be rewritten in a form that clarifies the structure involved. The equation:

AVERAGE(today) = (4/5)*AVERAGE(yesterday) + (1/5)SALES(yesterday)

can be rewritten:

AVERAGE(today) = AVERAGE(yesterday) − (1/5)*AVERAGE(yesterday)
+ (1/5)SALES(yesterday)

or:

AVERAGE(today) = AVERAGE(yesterday) + (1/5)SALES(yesterday)
− (1/5)*AVERAGE(yesterday)

This indicates that the average computed today is simply the average computed yesterday, plus one-fifth of yesterday's sales figure, minus one-fifth of the average computed yesterday.

The structure can be clarified further by noticing that the expression

(1/5)*SALES(yesterday) − (1/5)*AVERAGE(yesterday)

is simply one-fifth of the gap between yesterday's sales and yesterday's average. Thus the formula for today's average might be written:

AVERAGE(today) = AVERAGE(yesterday) + (1/5)*GAP(yesterday)

where GAP(yesterday) represents the gap between yesterday's sales and yesterday's average.

The formulation should now look quite familiar: it is simply a goal-gap structure. This can be seen more easily by writing the equations in DYNAMO:

```
L     AVG.K = AVG.J + (DT)(ADJ.JK)
NOTE     AVERAGE SALES (loaves)
R     ADJ.KL = GAP.K/5
NOTE     ADJUSTMENT RATE (loaves/day)
A     GAP.K = SALES.K − AVG.K
NOTE     GAP (loaves)
```

Average sales AVG is a level, and the adjustment rate tends to move the average toward the daily sales figure. But the average does not adjust immediately to daily sales. Instead, the gap is closed over a period of time.

Figure 17.9 shows a flow diagram for the exponential averaging process, corresponding to the DYNAMO equations. The flow diagram is identical in

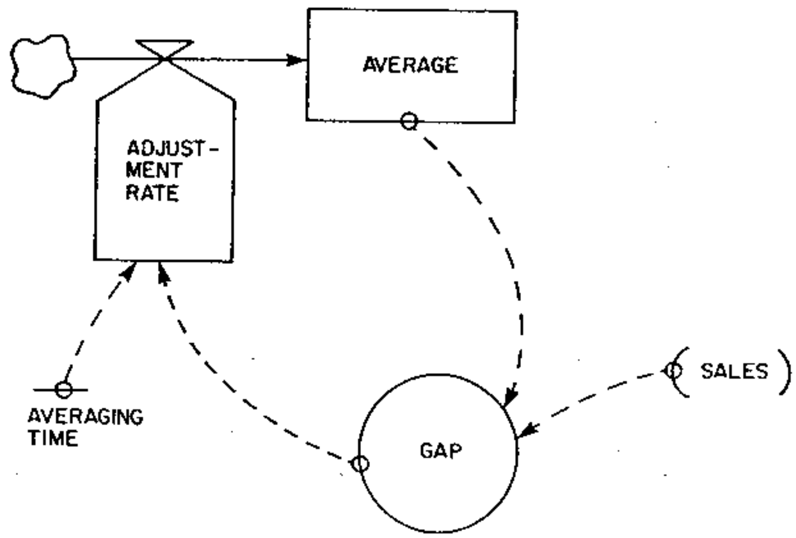

Figure 17.9 Flow diagram for exponential averaging

structure to the coffee cooling model discussed in Chapter 15. Coffee cools to room temperature over a period of time T. Similarly, the average sales in Marina's Bake Shop adjusts toward the daily sales over a period of time AT = 5. Thus the 5-day exponential average is simply a goal-gap formulation with an adjustment time of 5 days.

One striking difference exists between the coffee cooling example and that for Marina's Bake Shop. The coffee cooling case represents the adjustment of the coffee temperature to a constant outside temperature. Marina's average sales is adjusting to a fluctuating sales rate, but the structures are identical.

To demonstrate the similarity of the averaging structure and the coffee cooling structure, it is helpful to examine how the averaging process responds to a simple step-change in the sales rate. Table 17.3 shows sales figures for Evelyn's Bake Shop. As can be seen, sales in Evelyn's shop are much more regular than sales in Marina's Bake Shop. In fact, the only change in sales occurs on day ten, when sales jump from 100 to 120 loaves per day. Table 17.3 also shows computations for the five-days exponential average for days 1–25; and Figure 17.10 shows a plot of both actual sales and average sales. As expected, the exponential average responds to a step change exactly as a simple goal-gap structure.

Table 17.3 Evelyn's bake shop sales and average

Day	*Exponential average sales*	*Sales*
1	100	100
2	100	100
3	100	100
4	100	100
5	100	100
6	100	100
7	100	100
8	100	100
9	100	100
10	100	120
11	104	120
12	107	120
13	109.8	120
14	111.8	120
15	113.5	120
16	114.8	120
17	115.8	120
18	116.6	120
19	117.3	120
20	117.9	120
21	118.3	120
22	118.6	120
23	118.9	120
24	119.1	120
25	119.3	120

Exercise 10: Equations for the Exponential Average

a. Write DYNAMO equations to compute the average sales in Evelyn's Bake Shop, using a five-day exponential average.

b. Start the model in equilibrium.

c. Use a STEP function to represent the rise in sales from 100 to 120 on day 10.

d. How long does it take for average sales to rise half-way from 100 to 120?

e. Rerun the model, using as averaging time of 10 days rather than 5. How do the results differ?

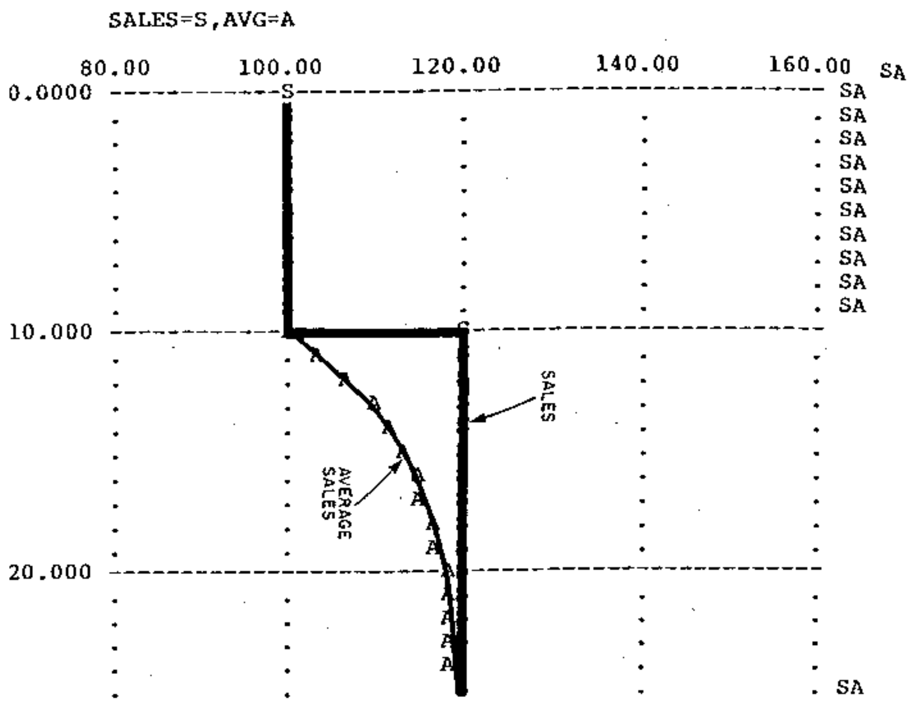

Figure 17.10 Evelyn's bake shop plotted simulation results

THE SMOOTH FUNCTION

The exponential average can be used whenever it is necessary to represent information averaging in a model. Figure 17.9 also shows the general structure involved in exponential averaging, with the input being averaged substituting for SALES in that diagram. The average to be computed is a level, which adjusts toward an input value over an averaging time.

Like first-order material delays, information averages are widely used in system dynamics modeling, and it is convenient to have a special flow diagram symbol for them. Figure 17.11 shows the symbol that is used to represent the exponential averaging process. Because this averaging process "smooths" the disturbances in the input, the function is often called a smoothing equation.

Whenever an exponential average is used in a model, it can be computed explicitly, using the level and rate equations previously described. In addition, DYNAMO includes a special function called SMOOTH, which can be used to

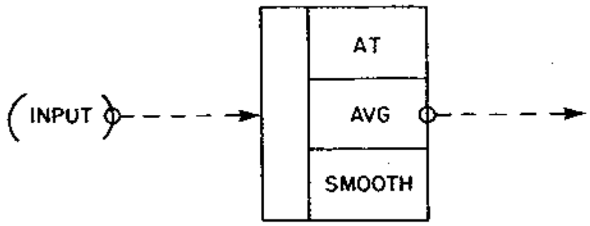

Figure 17.11 Flow diagram symbol for exponential averaging or smoothing

calculate the exponential average directly. The SMOOTH function takes the following form:

A AVG.K = SMOOTH(INPUT.K,AT)

where AVG represents the average to be computed, INPUT is the variable (rate, level, or auxiliary) to be averaged, and AT is the adjustment time to be used.[3]

Exercise 11: The SMOOTH Function

a. Use the SMOOTH function to represent the averaging process in Evelyn's Bake Shop.

b. Run the model and compare your results with your model in the previous exercise. The results should be identical.

c. Rerun the model, setting the adjustment time AT = 10. How do the results differ?

NEGATIVE LOOPS, INFORMATION DELAYS, AND CYCLES

Like material delays, information delays can often cause surprising behavior in seemingly simple structures. For example, consider the negative loop shown in Figure 17.12, describing the relationship between jobs and urban migration. (This loop was discussed in more detail in Exercises 16 and 22, Chapter 15.)

According to the loop, the availability of job openings influences people to migrate into the city; and as people migrate into the city, they fill the available openings. The simplest level and rate formulation for this structure involves just one level and one rate, shown in Figure 17.13. It should be evident that this simple structure will generate goal-seeking behavior: the number of people in the city will adjust smoothly to the number of jobs.

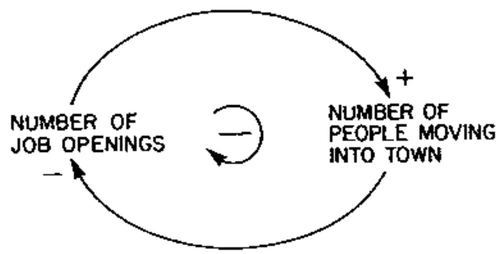

Figure 17.12 Relationship between jobs and migration

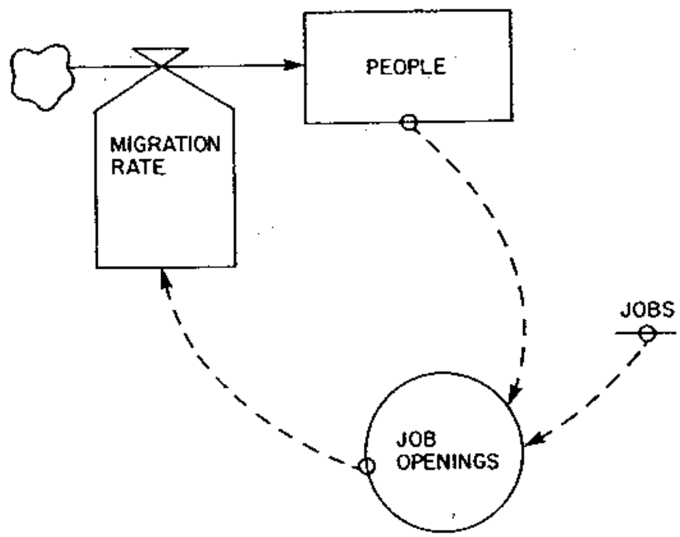

Figure 17.13 Flow diagram for negative feedback loop

While this formulation provides a rough description of the relationship between jobs and migration in a city, it ignores one important aspect of the problem. Undoubtedly, it takes time for people to learn about new job openings, and it takes even more time for them to relocate. Furthermore, people probably respond to the average number of job openings in a city, not to short-run increases or decreases. Thus the structure shown in Figure 17.14 is probably a more sensible representation of the relationship between jobs and migration.

The introduction of an information delay can cause the behavior of the model to change substantially. If people respond instantaneously to information about job openings, the adjustment process is smooth. Migration declines until any gap between the number of jobs and the number of workers is closed. But, if it takes time for people to respond to information about jobs, the model can generate cycles, as the following exercise indicates.

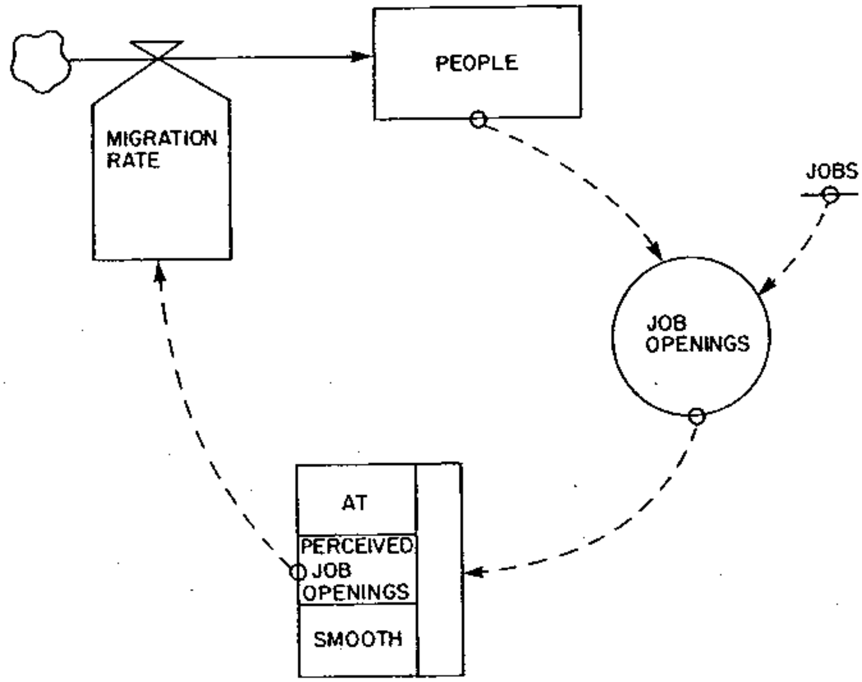

Figure 17.14 Negative loop with delay

Exercise 12: Information Delays, Migration, and Jobs

a. Review your model of migration and jobs developed in Exercise 21, Chapter 15. (It should correspond to the flow diagram structure shown in Figure 17.13.) Run the model and note the behavior it generates.

b. Revise the model to include a first-order information delay, as indicated in Figure 17.14. Choose an averaging time that you believe might represent the time required for people to respond to information about jobs.

c. Run the model and examine the results. (Be sure to examine the value of DT you have chosen. Make certain it is no more than one-third the value of the averaging time in your delay. See the discussion of DT at the end of Chapter 15.)

d. Rerun the model, choosing various values for the information averaging time. How do the results differ?

e. What do you believe causes the model to behave as it does?

EXAMPLE IV: STEVE'S ICE CREAM PARLOR—SEEMINGLY POSITIVE LOOPS THAT CONTAIN HIDDEN INFORMATION DELAYS

On occasion, a loop appearing in a causal-loop diagram may seem not to contain any level variables. Consider, for example, the loop shown in Figure 17.15, which represents the relationship between advertising, revenues, and sales in Steve's Ice Cream Parlor. According to the diagram, the amount Steve spends on advertising influences the number of ice cream cones sold; the number sold influences revenues; and the amount Steve earns in revenues influences the amount he spends on advertising.

At first glance, all the variables around the loop seem to be auxiliaries. This would produce the flow diagram shown in Figure 17.16. Equations for the model might be written:

```
A   AD.K=FRSA*REV.K
NOTE        ADVERTISING (DOLLARS/MONTH)
C   FRSA=0.1
NOTE        FRACTION OF REVENUE SPENT ON ADVER-
NOTE        TISING (DIMENSIONLESS)
A   SALES.K=SALESN+ADEFF*AD.K
NOTE        SALES (CONES/MONTH)
C   SALESN=1000
NOTE        SALES, NORMAL VALUE (CONES/MONTH)
C   ADEFF=5
NOTE        ADVERTISING EFFECTIVENESS (ADDITIONAL
NOTE        CONES PURCHASED PER DOLLAR
NOTE        SPENT ON ADVERTISING)
A   REV.K=SALES.K*PRICE
NOTE        REVENUE (DOLLARS/MONTH)
C   PRICE=1
NOTE        PRICE (DOLLARS/CONE)
```

The equations indicate that the amount spent on advertising each month is equal to one-tenth of the sales revenue earned per month. Furthermore, the

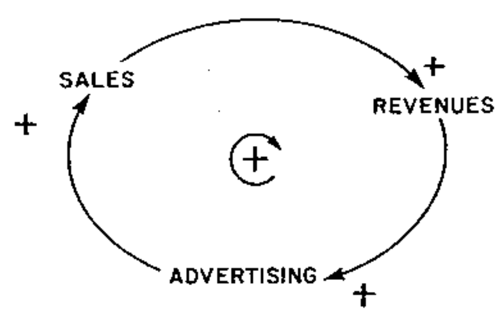

Figure 17.15 Causal relationships for Steve's Ice Cream Parlor

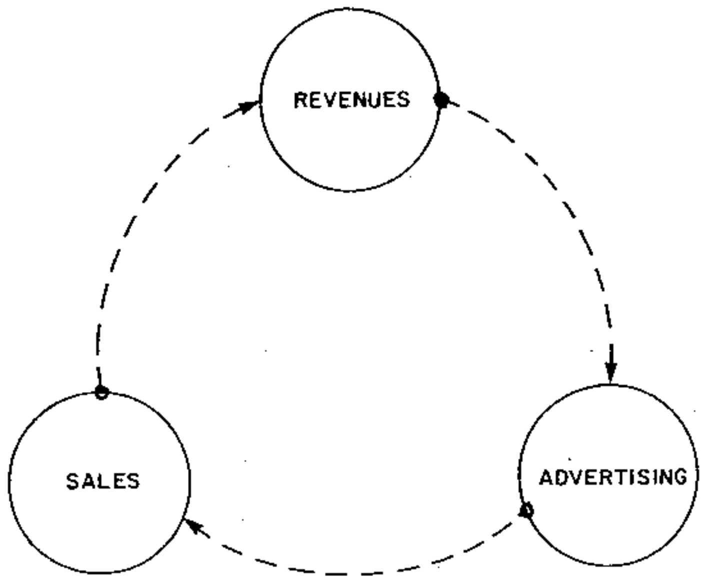

Figure 17.16 Flow diagram for Steve's Ice Cream Parlor

number of ice cream cones sold is a function of the amount Steve spends on advertising. For example, if Steve spends \$100 on advertising, the number of cones sold = 1000 + 5*100 = 1500. Finally, the revenue earned is equal to the number of cones sold, multiplied by the price of a cone.

While this information seems sensible, it suffers from an important difficulty, each variable has an instantaneous effect on the next. According to the equations, advertising has an instantaneous effect on sales; sales has an instantaneous effect on revenue; and revenue has an instantaneous effect on the amount spent on advertising.

Because all of the relationships in the loop are instantaneous, it is not possible to draw any conclusions about behavior over time. Technically, the model is written as a set of simultaneous equations. It is possible to analyze the equations algebraically, to determine whether there are values of advertising, sales, and revenue that are consistent with the entire set of equations. But, since none of the equations include an explicit formulation for a level and a rate of change, the equations cannot be used to generate behavior over time.[4]

Loops such as this one, which seem at first glance not to contain any level variables, often involve hidden information delays. For example, in the advertising loop, two information delays might be involved. First, Steve undoubtedly does not respond immediately to increases in sales by increasing the amount he spends on advertising. Instead, he probably uses average sales over a month or two to determine how much should be spent. And, as previous sections have indicated, averaging implicitly introduces a delay. Second, Steve's advertising probably does not have an immediate effect on the number of ice cream cones purchased. It probably takes consumers some time to notice and to react with purchases.

The following equations might be added to the model to represent the fact that Steve uses a monthly average of the number of ice cream cones sold to determine the amount he spends on advertising:

```
A   AVGREV.K=SMOOTH(REV.K,RAT)
NOTE        AVERAGE  REVENUE  (DOLLARS/MONTH)
C   RAT=1
NOTE        REVENUE  AVERAGING  TIME  (MONTHS)
A   AD.K=FRSA*AVGREV.K
NOTE        ADVERTISING  (DOLLARS/MONTH)
```

The equations indicate that average revenue AVGREV is a first-order exponential average, with an averaging time or time constant of one month; and the amount spent on advertising is one-tenth of the average revenue each month.

As usual, it is necessary to provide an initial value to begin the simulation. In this case, however, since the only level in the loop is contained in the SMOOTH function, it is necessary to choose an initial value for the SMOOTH, which, in this case, is REVN = 2000.

```
N     AVGREV=REVN
C     REVN=2000
```

Figure 17.17 is a modified flow diagram for the advertising and sales model, showing the first-order information delay involved in determining the average revenue. The flow diagram indicates that the model now contains two loops: one positive and one negative. As the following exercise demonstrates, the advertising and sales model can generate either exponential decay or exponential growth, depending on the values of the parameters chosen. Thus a structure that initially seemed to be a simple positive loop is actually more complex. It contains a hidden negative loop, and this has a critical influence on the model's behavior.

Exercise 13: Steve's Ice Cream Parlor

a. Write DYNAMO equations for the version of the advertising and sales model that does *not* include an information delay.

b. Try to run the model on the computer. What error message does DYNAMO generate?

c. Add the formulation for average revenue to your model. (As mentioned in the text, it is necessary to choose an initial value for the SMOOTH

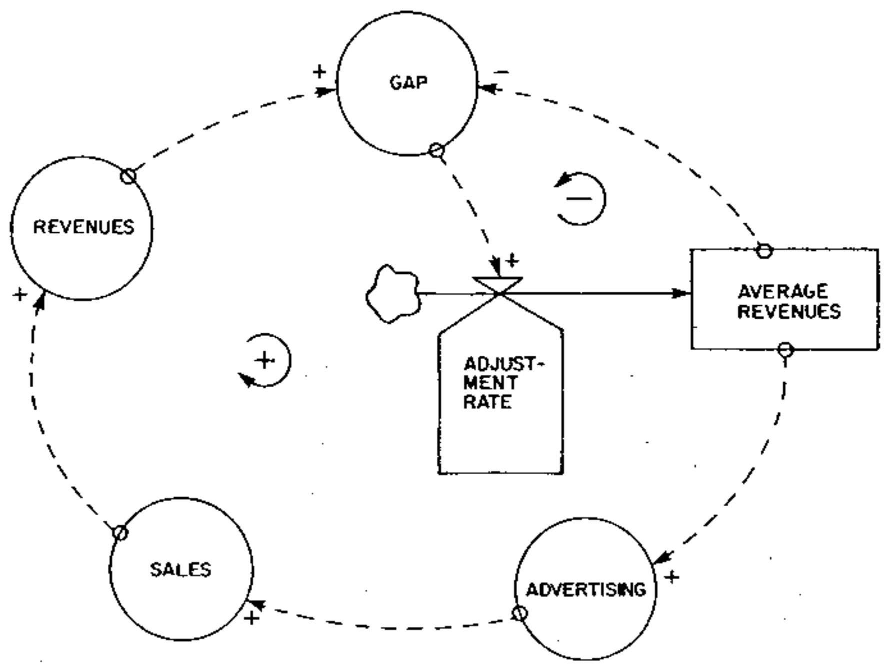

Figure 17.17 Modified flow diagram for Steve's Ice Cream Parlor

function. Choose an initial value that will place the system in equilibrium.) Test the response of the model to a STEP increase in normal sales, from 1000 cones per month to 2000 cones per month.

d. Repeat part (c), choosing a revenue averaging time RAT = 2 months. How do the results differ? Rerun the model, setting RAT = 0.5 months. How do the results differ?

e. Choose a value of zero for normal sales, and choose an initial value for the input to the SMOOTH function to start the system in equilibrium. How does the system respond to a STEP increase in normal sales, from zero to 1000 cones per month? (Compare your results with those in part (c).)

f. Repeat part (e), setting the fraction of revenue spent on advertising FRSA = 0.05. How do the results differ? Set FRSA = 0.2. How do the results differ? Rerun the model, setting FRSA = 0.3. Which loop dominates if FRSA is less than 0.2? Which loop dominates if FRSA is larger than 0.2. Which loop dominates if FRSA is *exactly* 0.2?

EXAMPLE V: TREE HARVESTING—HIGHER-ORDER DELAYS

The material and information delays considered so far have all been first-order delays: that is, they have involved only one level. Although first-order delays provide a useful representation of the delays involved in many social and economic systems, delay processes are frequently encountered that do not seem to resemble the behavior of a first-order delay. Recall, for example, the tree harvesting case in Chapter 7 (Example III). According to the example, Lester Splintz planted 10,000 saplings in 1930, and the particular species of trees he chose was supposed to reach harvesting size after an average of twenty years. Figure 17.18 (copied from Figure 7.6) indicates the harvest rate Lester Splintz obtained.

It seems somewhat plausible to represent the tree harvesting process as a first-order delay, as shown in the following causal-loop diagram and flow diagram (Figure 17.19), and equations (Figure 17.20). The behavior generated by this model is shown in Figure 17.21. Unfortunately, however, the behavior obtained does not resemble the behavior shown in Figure 17.18.

As the discussion in Chapter 7 indicated, the problem lies in the causal-loop representation. The initial causal-loop diagram fails to take into account the fact that trees come in different ages and sizes; for example, saplings (trees that are 0 to 1 inch in diameter); small trees (1 to 3 inches in diameter); medium-sized trees (3 to 6 inches in diameter); and harvestable trees (6-plus inches in diameter). Ordinarily, trees are not harvested until they reach

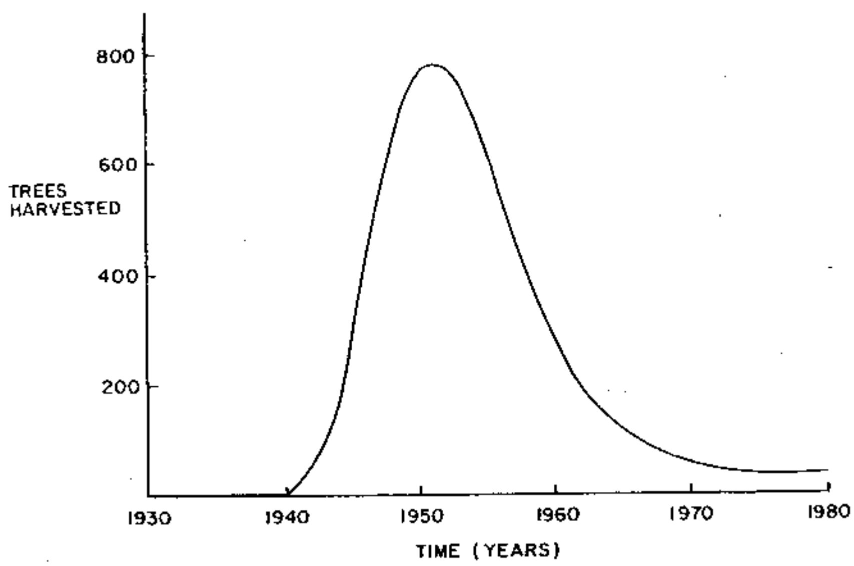

Figure 17.18 Harvest rate

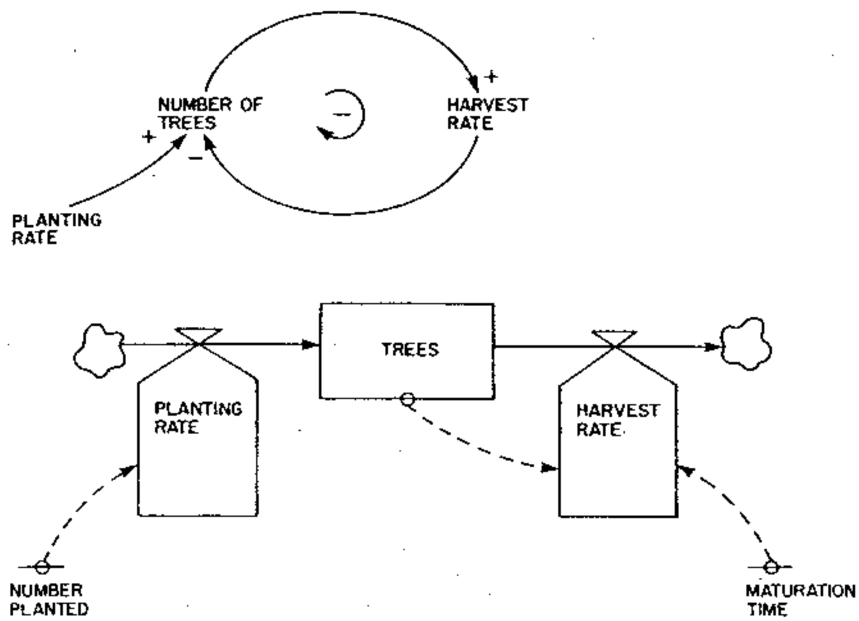

Figure 17.19 Causal-loop and flow diagrams for Splintz's farm

harvestable size. This distinction is incorporated in the causal-loop diagram in Figure 17.22 (copied from Figure 7.9).

The following flow diagram and equations, Figures 17.23 and 17.24, are based on the causal-loop diagram in Figure 17.22. This new model contains four levels, each with an adjustment time of 5 years (4*5 = 20). The behavior generated by the model is shown in Figure 17.25.

The harvest rate results (Figure 17.25) are quite similar to the data shown in Figure 17.18. The number of trees harvested remains at zero for the first 10 years. It then begins to rise, reaching a peak shortly after 1950. It then falls to zero by about 1975.

Exercise 14: Harvesting Model

a. Enter the DYNAMO equations for the model as shown in Figure 17.24.

b. Run the model and examine the behavior. It should be identical to the behavior in Figure 17.25.

c. Rerun the model, using a tree growth time of 40 years. Rerun the model, choosing a growth time of 10 years. How do the results differ?

```
*       TREE HARVESTING MODEL
NOTE
NOTE
L       TREES.K=TREES.J+(DT)(PLANT.JK-HRVSTR.JK)
N       TREES=TREESN
NOTE            TREES (TREES)
C       TREESN=0
NOTE            TREES, INITIAL (TREES)
R       PLANT.KL=(1/DT)*PULSE(NPLANT,PTIME,TBP)
NOTE            PLANTING RATE (TREES/YEAR)
C       NPLANT=10000
NOTE            NUMBER PLANTED DURING EACH PULSE (TREES)
C       PTIME=1930
NOTE            PLANTING TIME (YEARS)
C       TBP=1000
NOTE            TIME BETWEEN PULSES (YEARS)
R       HRVSTR.KL=TREES.K/MT
NOTE            HARVESTING RATE (TREES/YEAR)
C       MT=20
NOTE            MATURATION TIME (YEARS)
NOTE
NOTE            SIMULATION SPECIFICATIONS
NOTE
SPEC    DT=0.5/PLTPER=1/LENGTH=1980
N       TIME=1930
PLOT    TREES=T(0,10000)/HRVSTR=H(0,800)
RUN
```

Figure 17.20 Equations for Splintz's Farm

d. Modify your model to include only two levels: young trees and harvestable trees. Set the adjustment time for each level equal to half the total growth time (20 years). Run the model. How do the results differ from your results in part (b)?

e. Modify your model again, this time choosing three levels rather than two. How do the results differ?

f. Modify your model, using 5 levels rather than three. What happens?

g. What do you think would happen if you chose 10 levels? 20 levels? 100 levels?

DYNAMO FUNCTIONS FOR THIRD-ORDER DELAYS

Many social and economic processes resemble the tree-growing process previously discussed, in that they can be represented by a sequence of smaller delays. For example, while the apartment construction delay discussed earlier in this chapter was represented as a first-order delay, it is probably more accurate to consider it a higher-order delay, representing a sequence of smaller

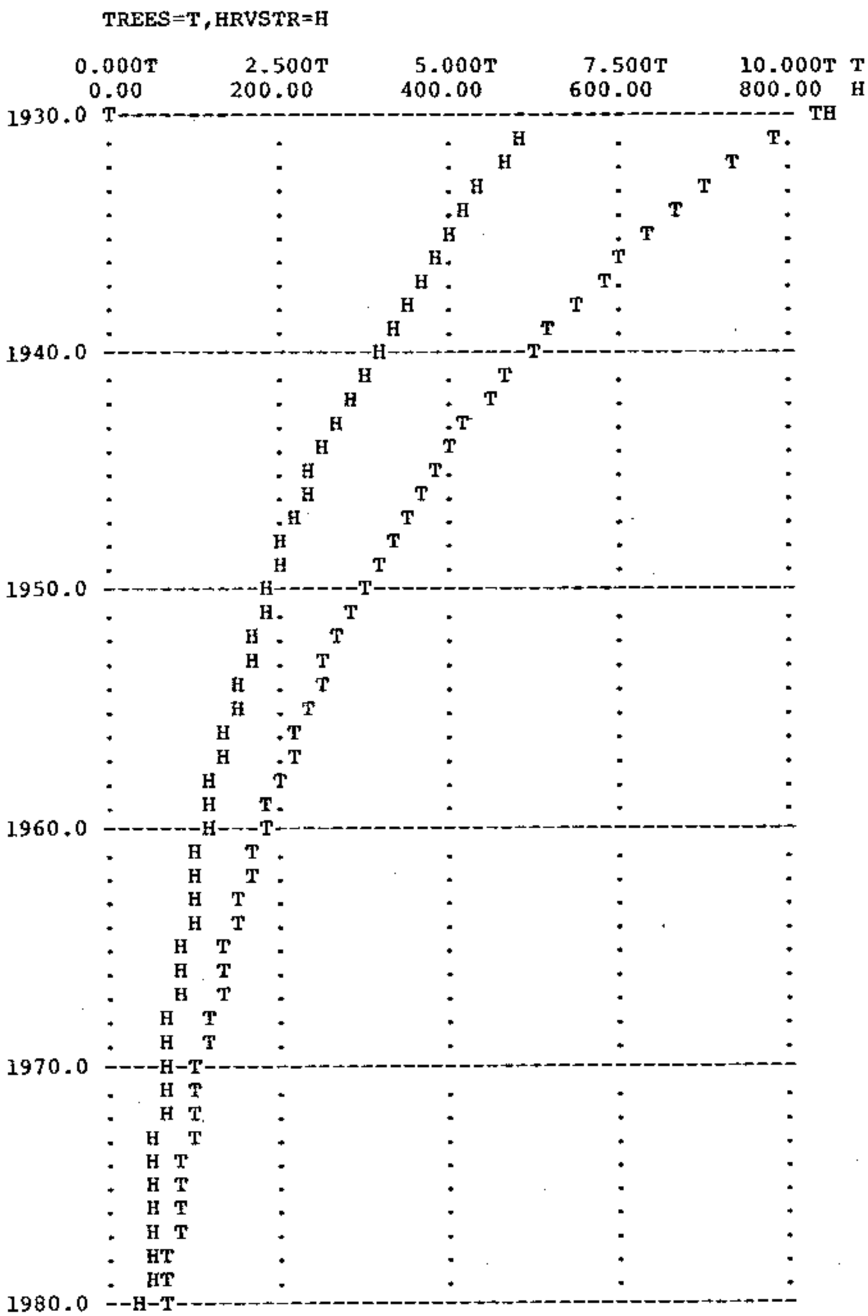

Figure 17.21 Plot for Splintz's farm

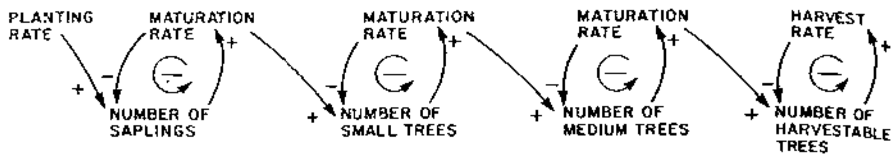

Figure 17.22 Causal-loop diagram showing levels of tree growth

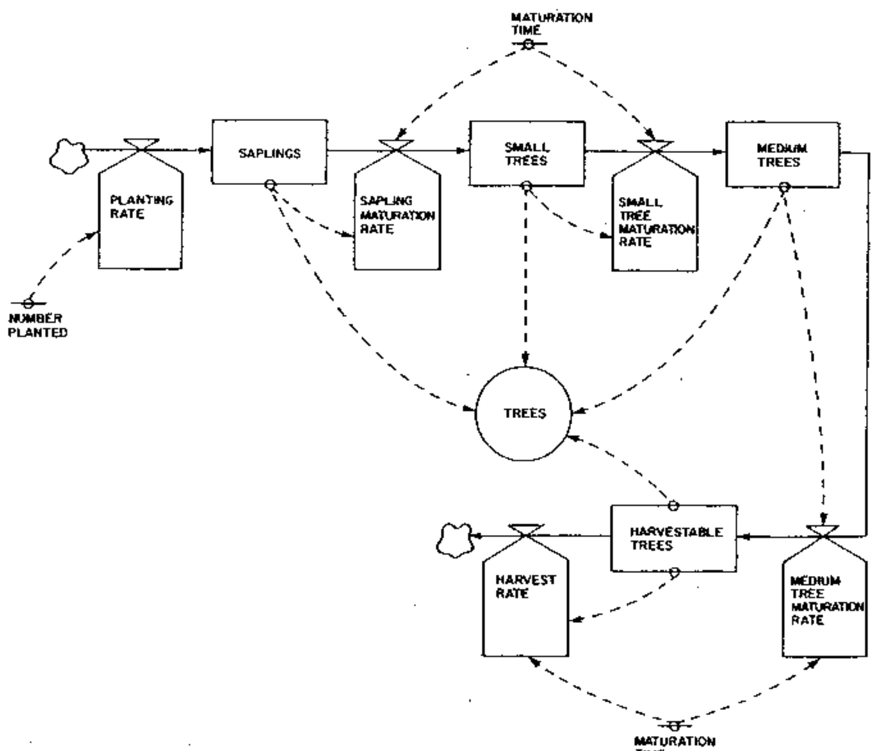

Figure 17.23 Flow diagram showing levels of tree growth

```
*         TREE HARVESTING MODEL
NOTE
NOTE
L         SAPLNG.K=SAPLNG.J+(DT)(PLANT.JK-SAPMR.JK)
N         SAPLNG=SPLNGN
NOTE              SAPLINGS (TREES)
C         SPLNGN=0
NOTE              SAPLINGS, INITIAL (TREES)
L         SMALL.K=SMALL.J+(DT)(SAPMR.JK-SMAMR.JK)
N         SMALL=SMALLN
NOTE              SMALL TREES (TREES)
C         SMALLN=0
NOTE              SMALL TREES, INITIAL (TREES)
L         MEDIUM.K=MEDIUM.J+(DT)(SMAMR.JK-MEDMR.JK)
N         MEDIUM=MDIUMN
NOTE              MEDIUM TREES (TREES)
C         MDIUMN=0
NOTE              MEDIUM TREES, INITIAL (TREES)
L         HRVST.K=HRVST.J+(DT)(MEDMR.JK-HRVSTR.JK)
N         HRVST=HRVSTN
NOTE              HARVESTABLE TREES (TREES)
C         HRVSTN=0
NOTE              HARVESTABLE TREES, INITIAL (TREES)
A         TREES.K=SAPLNG.K+SMALL.K+MEDIUM.K+HRVST.K
NOTE              TOTAL TREES (TREES)
R         PLANT.KL=(1/DT)*PULSE(NPLANT,PTIME,TBP)
NOTE              PLANTING RATE (TREES/YEAR)
C         NPLANT=10000
NOTE              NUMBER PLANTED DURING EACH PULSE (TREES)
C         PTIME=1930
NOTE              PLANTING TIME (YEARS)
C         TBP=1000
NOTE              TIME BETWEEN PULSES (YEARS)
R         SAPMR.KL=SAPLNG.K/MT
NOTE              SAPLING MATURATION RATE (TREES/YEAR)
R         SMAMR.KL=SMALL.K/MT
NOTE              SMALL TREE MATURATION RATE (TREES/YEAR)
R         MEDMR.KL=MEDIUM.K/MT
NOTE              MEDIUM TREE MATURATION RATE (TREES/YEAR)
R         HRVSTR.KL=HRVST.K/MT
NOTE              HARVEST RATE (TREES/YEAR)
N         MT=TMT/4
NOTE              MATURATION TIME (YEARS)
C         TMT=20
NOTE              TOTAL MATURATION TIME (YEARS)
NOTE
NOTE              SIMULATION SPECIFICATIONS
NOTE
SPEC      DT=0.5/PLTPER=1/LENGTH=1980
N         TIME=1930
PLOT      TREES=T(0,10000)/HRVSTR=H(0,800)
RUN
```

Figure 17.24 Model of tree growth

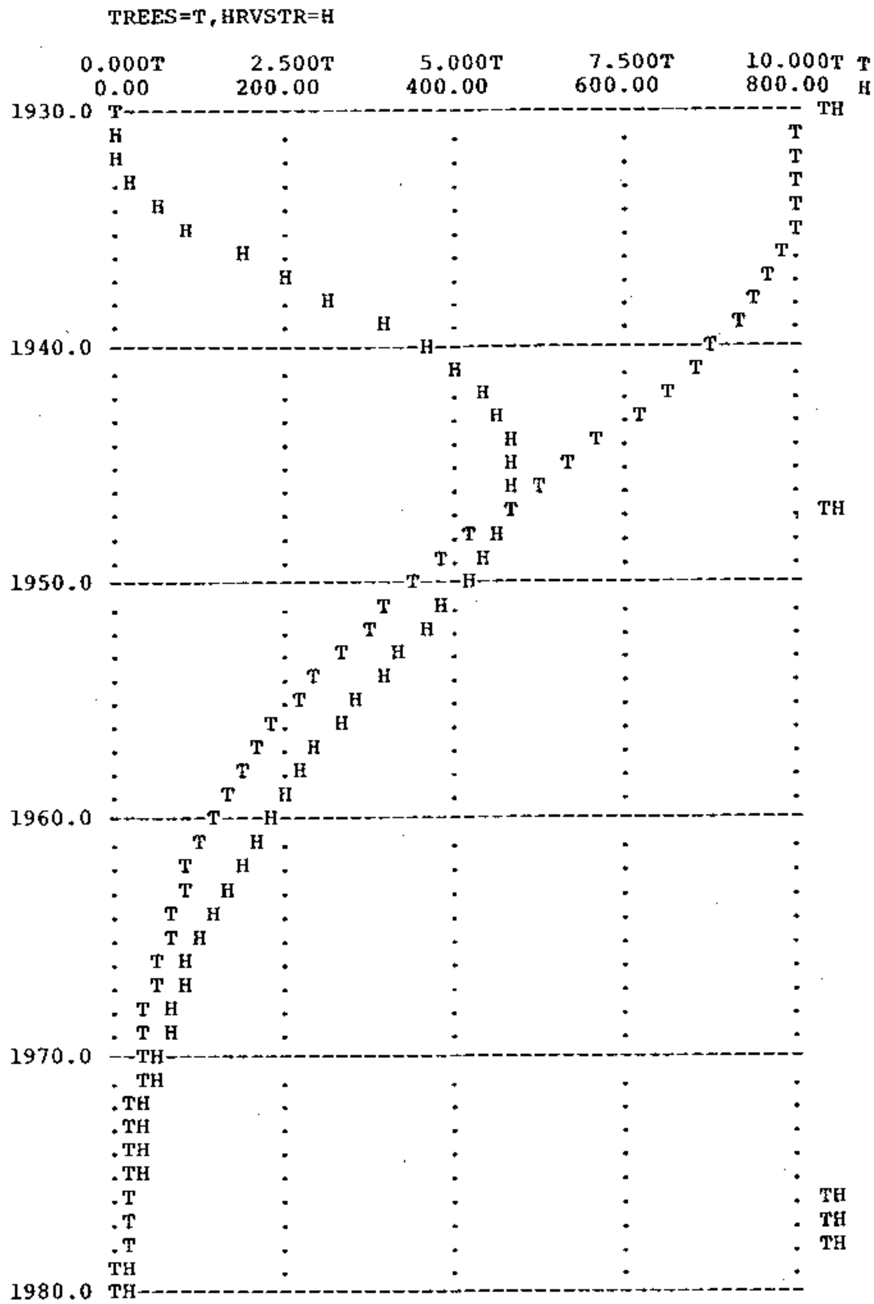

Figure 17.25 Tree growth model

delays (the delays involved in getting permits, hiring an architect, negotiating loans, and so forth). Similarly, the mail delay discussed in Exercise 5 might more realistically be considered a higher-order delay, since it undoubtedly involves a sequence of smaller delays (the delays involved in sorting the mail, getting it from post-office to post-office, and delivering it).

When modeling a delay process, one immediate question arises: How many levels should be included in the delay? The answer, of course, depends on the exact process being modeled. However, to a rough approximation, it is often sufficient to choose either a first-order delay or a third-order delay (that is, a delay with only one level or a delay with three).

As the preceding exercise indicates, for example, the behavior of a third-order delay is reasonably similar to the behavior of a fourth-order delay. Thus either might be a reasonable choice in modeling the tree-harvesting case.

DYNAMO includes special functions to represent third-order material and information delays, analogous to the DELAY1 and SMOOTH first-order delays. The DELAY3 function can be used to represent third-order material delays, and the DLINF3 function can be used to represent third-order information delays. The DELAY3 and DLINF3 functions are simply shorthand notations, exactly identical to the full set of equations for third-order delays. For example, the following equations might be used to represent the tree-harvesting case:

```
R    HARVST.KL=DELAY3(PLANT.JK,GT)
C    GT=20
```

This indicates that the harvest rate HARVST is a third-order material delay of the planting rate PLANT, with a total growth time GT = 20 (years). The following equations might be used to indicate that perceived job openings PJO is a third-order information delay of actual job openings JO, with an adjustment time of two years.

```
A    PJO.K=DLINF3(JO.K,AT)
C    AT=2
```

Exercise 15: Using the DELAY3 and DLINF3 Functions

a. Use the third-order material delay function (DELAY3) to represent the tree-harvesting case.

b. Reread Exercise 2 in Chapter 7 (Tree Harvesting—Part II). Use a third-order delay function to represent Warren's tree harvesting process. (*Hint:* Use a STEP function to represent the planting rate.)

c. Modify your model of the Warren Splintz case, using a first-order delay rather than a third-order delay. How do the results differ? Revise the model, using a fourth-order delay. (One way to do this is to combine a third-order and first-order delay. What delay times should you choose for each?)

d. Modify your apartment construction model (Exercise 7), using a third-order material delay, rather than a first-order delay. How do the results differ?

e. Modify your model of job-migration cycles (Exercise 12), using a third-order information delay (DLINF3) rather than a first-order delay. How do the results differ?

ENDNOTES

1. When the DELAY1 function is used, DYNAMO automatically calculates an initial value for the level equation in the delay, using the formula: LEVELN = INFLOW*AT. This produces an initial outflow rate exactly equal to the inflow rate. For example, in the Nobug case, DYNAMO selects an initial value of 0, since the initial value of the dumping rate is 0. It is possible to initialize the level at whatever value is desired, by using an initial value equation for the inflow rate. For example, the following equations could be used to set the initial value of the level of NOBUG equal to 100:

```
R    ABSORB.KL=DELAY1(DUMP.JK,NAT)
C    NAT=2
N    DUMP=100/NAT
R    DUMP.KL=(1/DT)*PULSE(420,1,7)
```

   DYNAMO will calculate the initial value of the level equation LEVELN = DUMP*NAT = (100/NAT)*NAT = 100.

2. For an infinite sequence of observations, it is not hard to show that in a five-day exponential average the most recent day's sales receives a weight of 1/5; the second most recent, a weight of 4/25; the third, 16/125; and in general, the *n*th most recent previous observation receives a weight of

$$\frac{4^{n-1}}{5^n}$$

3. As previously explained, all averages really are levels. A peculiarity of DYNAMO processing requires that the average be treated as an auxiliary equation (A) when the SMOOTH function is used. Using the proper L (for Level) with the SMOOTH equation will generate a strange DYNAMO error message. In order to avoid problems, the model-builder should also provide an initial condition (N equation) for AVG, as would be done with any other level equation.

4. DYNAMO checks to make sure there is a level equation in all loops. If one or more loops occur without levels, DYNAMO generates an error message similar to the following:

   "SIMULTANEOUS EQUATIONS IN THE AUX EQUATIONS FOR AD" which indicates that the auxiliary variable "AD" is part of a simultaneous equation loop.

   If the equations of the advertising model are analyzed as a set of simultaneous equations, the solution is:

   SALES = 2000 cones/month; REV = 2000 dollars/month; and AD = 200 dollars/month.