STAT 280 Sept. 26, 2001

Our next topic is Markov Chains (Ch 4 in your text):

This is an example of a Discrete Time Stochastic Process: a sequence, of random variables, usually with some dependence from one time to the next.

{X(t): t=1,2,3, ....}

Markov Chains have a particular kind of dependence - Given the present, the future is independent of the past. A more formal definition is given on p 142.

For example, the random walk is such a process.

An important class of these processes is Queues: or "waiting lines".

Queues are defined by an arrival process and a service process.

Think of supermarkets, transportation systems, traffic, ...

But some processes which may not seem to be queues are probabilistically the same as queues.

Maintenance: 5 computer example on pp 136-139.

"Queue" is the number of computers waiting to be repaired.

Arrivals occur when a computer needs repair
Service fixes the computer but takes time.

If we make certain specifications about the service time distribution, and how long a working computer will stay working, we will have defined a queuing process.

The introductory example in the book specifies as follows:

Each day, a working computer will fail with probability 0.2.
Each day, the technician can service only one failed computer.

(Even if the computer fails on say, Monday, we assume it will be fixed before Tuesday morning.)

Does this system tend to build up a backlog?

While we might be interested in the long term behaviour of this Maintenance program, in this example we just look at one week(5 days).

How do we simulate this queuing system for one week by computer, so
We can study the system probabilistically from many realizations of the
week's experience?

The book defines a program in a pseudocode, and I want to show what
this looks like in MINITAB.

Note the focus on the "queue length", not the "arrivals" or "services".

```
Gmacro
Q
name k1 'DAY'*****c1 'BACKLOG'*****k2 'GOOD'*****k3 'FAIL'*****k4 'IDLEDAYS'
#DAY is the day of the week being simulated
#BACKLOG(I) is the backlog at the beginning of day I+1  (Monday =2)
#GOOD is the number of computers 'up' at the end of the day
#FAIL is the number of computers failing today
# C1 is the column of backlogs, with 6 rows including sunday=0
# C2(1)  is the simulated number of failures (It is only a column to
# satisfy MINITAB syntax for the Binomial subcommand.
#In this example, the technician can deal with one failure per day
# so if a computer fails, it is fixed same day.
erase c1 c2     #initializiation
let k3=0
let k4=0
let k2=5
let c1(1)=0      # yesterday's backlog is set to 0 (an assumption)
do k1=1:5        # start week on day 1, and repeat each day
print  k1-k4 c1
if k2=0           # if yesterday had no good ones, don't look for failures
let k3=0          # 0 failures
goto 1             # skip look for failures (because binomial n=0 will crash)
endif
random 1 c2;    #how many failures among the good ones?
bino k2 .2.
Let k3=c2(1)    # label k3 for convenience
mlabel 1           # this is destination of above goto, if needed
if ((k3=0) and (c1(k1)=0))  # if no failures and no backlog yesterday
let c1(k1+1)=0                   # then set backlog = 0 for today
let k4=k4+               # also today is idle and increase idledays by 1
else
let c1(k1+1)=c1(k1)+k3-1  # otherwise increase backlog by failures less
endif                                  # the one that was fixed today
let k2=5-c1(k1+1)  # the good ones left for tomorrow are the ones not backlogged
enddo                                 # end of do-loop (repeat loop until do-
loops finished).
Endmacro
```

Sample Output:

```
MTB > %q
Executing from file: q.MAC

Data Display

DAY      1.00000
GOOD     5.00000
FAIL     0
IDLEDAYS 0

C1
   0


Data Display

DAY      2.00000
GOOD     5.00000
FAIL     0
IDLEDAYS 1.00000

C1
   0    0


Data Display

DAY      3.00000
GOOD     5.00000
FAIL     1.00000
IDLEDAYS 1.00000

C1
   0    0    0


Data Display

DAY      4.00000
GOOD     2.00000
FAIL     4.00000
IDLEDAYS 1.00000

C1
   0    0    0    3
```

Data Display

DAY      5.00000
GOOD     3.00000
FAIL     0
IDLEDAYS 1.00000

C1
   0    0    0    3    2

Note that the Markov Property will often hold for queues:

Future independent of past given present.

More general approach to Markov Chains:

Define State Space:
Define Starting State:
Define one-step transition matrix:

Markov chain depends only on these things.

P transition matrix:

What do the elements of P2 look like?

Pn turns out to be the n-step transition matrix.

A weather example:

1 - clear
2 - cloudy
3 - rainy

discret time - by days

1-step transition matrix

.5 .4 .1
.1 .7 .2
.05 .35 .6

We can produce the following powers of this matrix:

MTB > copy c1-c3 m1
MTB > mult m1 m1 m2
MTB > mult m2 m2 m3
MTB > mult m3 m3 m4

```
MTB > mult m4 m4 m5
MTB > print m1-m5

Data Display


 Matrix M1

  0.50   0.40   0.10
  0.10   0.70   0.20
  0.05   0.35   0.60


 Matrix M2

  0.295   0.515   0.190
  0.130   0.600   0.270
  0.090   0.475   0.435


 Matrix M3

 0.171075  0.551175  0.277750
 0.140650  0.555200  0.304150
 0.127450  0.537975  0.334575


 Matrix M4

 0.142189  0.549727  0.308084
 0.140914  0.549395  0.309691
 0.140111  0.548924  0.310965


 Matrix M5

 0.140848  0.549297  0.309855
 0.140845  0.549296  0.309859
 0.140843  0.549295  0.309862
```

A conclusion is that it is clear 14% of the time, cloudy 55%, and rainy 31 %.

More theory on this to come.