DISCRETE EVENT SIMULATION

Discrete Event Simulation Overview

- Events: we need descriptions of possible events and
  the probabilitity distributions for the event times;
  the "event list" is maintained and updated as system changes.
- Variables: time t, counters, system state and output variable(s).
- Examples: single and multiple server queues, inventory stocking models, insurance risk models, multimachine repair models, stock option models.

SINGLE-SERVER QUEUES

Single-Server Queues

- Assumptions:
  a) customer arrivals $\sim$ a homogeneous Poisson process with rate λ,
     so inter-arrival times are $\sim Exp(\lambda)$;
     could also use nonhomogeneous $\lambda(t)$ Poisson process;
  b) sequential customer service from one server;
  c) random service time Y with distribution specified by some G;
  d) no customers are allowed to enter system after a final time T.
- Variables:
  a) time t;
  b) counters $N_A(t) = \#$ arrivals by time t, and $N_B(t) = \#$ departures by time t;
  c) state variable $n(t) = \#$ of customers in system at t.
- Events: arrivals or departures; the event list contains
  time for next arrival t_A and time for next departure t_D.
- Example output variables are:
  a) total customers served N;
  b) arrival times $A(i)$ for customers with $i = 1, .., N$;
  c) departure times $D(i)$ for customers with $i = 1, .., N$;
  d) $T_p =$ time past final allowed arrival T, for last service.

SINGLE-SERVER QUEUES CONTINUED

- Single Server Queue Simulation Algorithm:
  Initialize: $t = N_A = N_D = 0$, $n = 0$;
  generate T_0, set $t_A = T_0$, $t_D = \infty$.
  Advance the system state using four cases:
  1. $t_A \leq t_D$, and $t_A \leq T$ (a new arrival at time t_A, before next departure)
     a) reset $t = t_A$, $n = n + 1$, $N_A = N_A + 1$;
     b) generate T_t and reset $t_A = T_t$ (next arrival time);
     c) if $n = 1$ generate $Y \sim G$ and reset $t_D = t + Y$;
     d) collect output data $A(N_A) = t$.
  2. $t_D < t_A$, $t_D \leq T$ (a departure at time t_D)
     a) reset $t = t_D$, $n = n - 1$, $N_D = N_D + 1$;
     b) if $n > 0$ generate $Y \sim G$ and reset $t_D = t + Y$;
     otherwise set $t_D = \infty$ (queue is empty);
     c) collect output data $D(N_D) = t$.
  3. $min(t_A, t_D) > T$, $n > 0$ (try to empty the queue)
     a) reset $t = t_D$, $n = n - 1$, $N_D = N_D + 1$;
     b) if $n > 0$ generate $Y \sim G$ and reset $t_D = t + Y$;
     c) collect output data $D(N_D) = t$.
  4. $min(t_A, t_D) > T$, $n = 0$ (queue empty)
     a) collect output data $T_p = \max(t - T, 0)$.

SINGLE-SERVER QUEUES CONTINUED

- End of "run" results
  - Times $(A(1), D(1)), (A(2), D(2)), \ldots, (A(N_A), D(N_A))$;
    $(D - A)$ average provides average of customer times for each run.
  - Time T_p is server overtime.
  - (Event,Time) pairs $(n_1, t_1), (n_2, t_2), \ldots$ provide an arrival/departure history:
    $n = 0, \quad 0 \leq t < t_1,$
    $n = n_1, \; t_1 \leq t < t_2,$
    $n = n_2, \; t_2 \leq t < t_3,$
    $\vdots$
- Averages over many runs give expected service time, server overtime, and other statistics.
- Example: Assume arrivals for a single server queue follow a homogeneous Poisson process with rate 3/hour, and the service time is uniformly distributed between 12 and 18 minutes.
  Estimate the average time a customer spends in the system and the average amount of overtime put in by the server if $T = 8$ hours.

SINGLE-SERVER QUEUE EXAMPLE

```
function [A D Tp] = snglsv( T, lam )
% Returns arrival times A, departure times D, and overtime Tp
%  Single-Server Q Simulation
  t = 0; na = 0; nd = 0; n = 0; ta = -log(rand)/lam; td = inf;
  while ta <= T % time left for more arrivals
    if ta <= td, t = ta; n = n + 1; % new arrival
      na = na + 1; A(na) = t; ta = t - log(rand)/lam;
      if n == 1, td = t + G; end,
    else, t = td; n = n - 1; % departure
      nd = nd + 1; D(nd) = t;
      if n >  0, td = t + G; else, td = inf; end
    end
  end % no more arrivals, empty the Q
  while n > 0, t = td; nd = nd + 1; D(nd) = t;
    n = n - 1; td = t + G;
  end,
  Tp = max(t-T,0); % Overtime
% end snglsv
function Y = G;  % Uniform between .2 and .3
   Y =  .2 + .1*rand;
% end % G = uniform(.2,.3)
```

SINGLE-SERVER QUEUE EXAMPLE

Sample Output:

```
[ A, D, OT ] = snglsv(8,3); disp( [ A; D ] ); disp(OT)
0.42035  0.65946  0.87783  0.91594  1.0987
0.66493  0.91209  1.1605   1.3804   1.6201
... 27 total arrivals and departures ...
6.3914   6.4858    6.608   7.9071  7.9328
6.643    6.9398    7.1588  8.1947  8.4424
0.44236
for K = [100 1000] % collect ave customer times for each run
  for i = 1:K, [A,D,OT(i)] = snglsv(8,3); C(i) = mean(D-A); end
  disp([K mean(C) mean(OT)])
end
 100      0.54095      0.37947
1000      0.52367      0.37052
clear
for K = [100 1000] % change lambda
  for i = 1:K, [A,D,OT(i)] = snglsv(8,4); C(i) = mean(D-A); end
  disp([K mean(C) mean(OT)])
end
 100      0.89534        1.058
1000      0.90772       1.0916
```