Performance Modelling — Lecture 14: Random Variables and Simulation

Jane Hillston
School of Informatics
The University of Edinburgh
Scotland

9th March 2017

Introduction

- Random variables play two important roles in simulation models.

Introduction

- Random variables play two important roles in simulation models.
  1. We assume that within our models some delays will not have deterministic values, but instead will be represented by random variables; and

Introduction

- Random variables play two important roles in simulation models.
  1. We assume that within our models some delays will not have deterministic values, but instead will be represented by random variables; and
  2. when a choice must be made within the behaviour of an entity we will sometimes want the decision to be made probabilistically.

Introduction

- Random variables play two important roles in simulation models.
  1. We assume that within our models some delays will not have deterministic values, but instead will be represented by random variables; and
  2. when a choice must be made within the behaviour of an entity we will sometimes want the decision to be made probabilistically.
- Both cases will involve sampling a probability distribution to extract a value each time this part of the entity's behaviour is reached.

Introduction

- Random variables play two important roles in simulation models.
  1. We assume that within our models some delays will not have deterministic values, but instead will be represented by random variables; and
  2. when a choice must be made within the behaviour of an entity we will sometimes want the decision to be made probabilistically.
- Both cases will involve sampling a probability distribution to extract a value each time this part of the entity's behaviour is reached.
- Both cases rely on the random number generator.

Random variables

- We also assume that the variables characterising the behaviour of the system/model, the performance measures or output parameters, are also random variables.

Random variables

- We also assume that the variables characterising the behaviour of the system/model, the performance measures or output parameters, are also random variables.
- In general, each run of the simulation model provides a single estimate for these random variables.

Simulation and steady-state

- If we want steady state values, the longer we run a simulation the better our estimate will be. However, it still remains a single observation in the sample space.

Simulation and steady-state

- If we want steady state values, the longer we run a simulation the better our estimate will be. However, it still remains a single observation in the sample space.
- We need more than a single estimate in order to draw conclusions about the system.

Simulation and steady-state

- If we want steady state values, the longer we run a simulation the better our estimate will be. However, it still remains a single observation in the sample space.
- We need more than a single estimate in order to draw conclusions about the system.
- We use output analysis techniques to improve the quality of an estimates and to develop ways of gaining more observations without excessive computational cost.

Simulation and steady-state

- If we want steady state values, the longer we run a simulation the better our estimate will be. However, it still remains a single observation in the sample space.
- We need more than a single estimate in order to draw conclusions about the system.
- We use output analysis techniques to improve the quality of an estimates and to develop ways of gaining more observations without excessive computational cost.
- Realistic simulation models take a long time to run—there is always a trade-off between accuracy of estimates and execution time.

Random number generation

- Generating random values for variables with a specified random distribution, such as an exponential or normal distribution, involves two steps.

Random number generation

- Generating random values for variables with a specified random distribution, such as an exponential or normal distribution, involves two steps.
  1. A sequence of random numbers distributed uniformly between 0 and 1 is obtained.

Random number generation

- Generating random values for variables with a specified random distribution, such as an exponential or normal distribution, involves two steps.
  1. A sequence of random numbers distributed uniformly between 0 and 1 is obtained.
  2. The sequence is transformed to produce a sequence of random values which satisfy the desired distribution.

Random number generation

- Generating random values for variables with a specified random distribution, such as an exponential or normal distribution, involves two steps.
  1. A sequence of random numbers distributed uniformly between 0 and 1 is obtained.
  2. The sequence is transformed to produce a sequence of random values which satisfy the desired distribution.
- This second step is called random variate generation.

Generating uniform random numbers

- To obtain a sequence of uniform random numbers between 0 and 1 in fact we generate a sequence X_k of integers in the range $[0, M-1]$

Generating uniform random numbers

- To obtain a sequence of uniform random numbers between 0 and 1 in fact we generate a sequence X_k of integers in the range $[0, M-1]$
- The sequence $X_k/(M-1)$ will then be approximately uniformly distributed over $(0, 1)$.

Generating uniform random numbers

- To obtain a sequence of uniform random numbers between 0 and 1 in fact we generate a sequence X_k of integers in the range $[0, M-1]$
- The sequence $X_k/(M-1)$ will then be approximately uniformly distributed over $(0, 1)$.
- In 1951, D.H. Lehmer discovered that the residues of successive powers of a number have good randomness properties.

Lehmer generators

- Lehmer obtained the kth number in the sequence by dividing the kth power of an integer a by another integer M and taking the remainder.

$$X_k = a^k \bmod M$$

Lehmer generators

- Lehmer obtained the kth number in the sequence by dividing the kth power of an integer a by another integer M and taking the remainder.

$$X_k = a^k \bmod M$$

This can be expressed as an iteration:

$$X_k = (a \times X_{k-1}) \bmod M$$

Lehmer generators

- Lehmer obtained the kth number in the sequence by dividing the kth power of an integer a by another integer M and taking the remainder.

$$X_k = a^k \bmod M$$

This can be expressed as an iteration:

$$X_k = (a \times X_{k-1}) \bmod M$$

- The parameters a and M are called the multiplier and the modulus respectively.

Lehmer generators

- Lehmer obtained the kth number in the sequence by dividing the kth power of an integer a by another integer M and taking the remainder.

$$X_k = a^k \bmod M$$

This can be expressed as an iteration:

$$X_k = (a \times X_{k-1}) \bmod M$$

- The parameters a and M are called the multiplier and the modulus respectively.
- Random number generators of this form are called Lehmer generators, or multiplicative linear-congruential generators.

Desirable properties for a random number generator

- It should be efficiently computable
  Simulations typically require several thousand random numbers in each run so processor time to generate these should be kept small.

Desirable properties for a random number generator

- It should be efficiently computable
  Simulations typically require several thousand random numbers in each run so processor time to generate these should be kept small.
- It should be pseudo-random
  Given the same seed, the random number generator should produce exactly the same sequence of numbers. (Good for reproducibility of experiments.)

Desirable properties for a random number generator

- It should be efficiently computable
  Simulations typically require several thousand random numbers in each run so processor time to generate these should be kept small.
- It should be pseudo-random
  Given the same seed, the random number generator should produce exactly the same sequence of numbers. (Good for reproducibility of experiments.)
- The cycle should be long
  A short cycles may lead to repeated event sequences. This may limit the useful length of simulation runs.

Desirable properties for a random number generator

- It should be efficiently computable
  Simulations typically require several thousand random numbers in each run so processor time to generate these should be kept small.
- It should be pseudo-random
  Given the same seed, the random number generator should produce exactly the same sequence of numbers. (Good for reproducibility of experiments.)
- The cycle should be long
  A short cycles may lead to repeated event sequences. This may limit the useful length of simulation runs.
- Independent and uniformly distributed succesive values
  The correlation between successive numbers should be small.

Problems with random number generators

- Research has shown that Lehmer generators obey these properties provided a and M are carefully chosen. However care is needed.
- In the early 1970s most university mainframes were using a linear-congruence generator known as `RANDU`.
- It used the values $a = 65539$ and $M = 2^{31}$.
- Although the output looked random, detailed statistical analysis showed that there was significant correlation in the output.

Efficiency of random number generators

- This form of generator continues to be used, if somewhat more warily.
- These generators are particularly efficient if M is chosen to be a power of 2.
- In this case finding the residue amounts to simply truncating the result of the multiplication.
- However a modulus of the form 2^k results in a shorter cycle: 2^{k-2} at best.

Mersenne Twister

- One of the best families of random number generators for simulation is that based on the Mersenne Twister algorithm.
- It is used by default in python, R, MATLAB and several other languages.
- It comes in a number of variants, but the commonly used MT19937 variant produces a sequence of 32-bit integers, and has the following desirable properties:
  - It has a very long period of $2^{19937} - 1$.
  - It passes numerous tests for statistical randomness, including some stringent tests which are failed by linear congruential random number generators.

Random variate generation algorithms

Random variate generation algorithms for values of the commonly used probability distributions, based on a uniformly distributed stream of values between 0 and 1, can be found in many books on simulation and performance modelling.

Random variate generation algorithms

Random variate generation algorithms for values of the commonly used probability distributions, based on a uniformly distributed stream of values between 0 and 1, can be found in many books on simulation and performance modelling.

A good example is the book by Raj Jain:

The Art of Computer Systems Performance Analysis, Wiley, 1991.

Inverse transformations

- Inverse transformation algorithms are based on the observation that for any probability distribution with distribution function $F(x)$, the value of $F(x)$ is uniformly distributed between 0 and 1.

Inverse transformations

- Inverse transformation algorithms are based on the observation that for any probability distribution with distribution function $F(x)$, the value of $F(x)$ is uniformly distributed between 0 and 1.
- Thus, using values from the random number stream, $u = X_k$, the function is inverted to find the next value of x: $x = F^{-1}(u)$.

Exponential distributions

For example, given a random number u, we generate the next value in an exponential distribution with parameter λ as

$$x = -\frac{1}{\lambda} \ln(u)$$

Exponential distributions

For example, given a random number u, we generate the next value in an exponential distribution with parameter λ as

$$x = -\frac{1}{\lambda} \ln(u)$$

Note

Strictly speaking, the equation should be

$$x = -\frac{1}{\lambda} \ln(1 - u)$$

but since u is uniformly distributed between 0 and 1, $1 - u$ will be uniformly distributed between 0 and 1 and the generation algorithm can be simplified.

Boolean-valued distributions

- Boolean-valued distributions which are used to make decisions within a model take a single real parameter, p, such that $0 \leq p \leq 1$.
- This represents the probability of a "positive" outcome.
- Then each time the branching point in the model is reached, the next random number in the stream is generated $u = X_k$.
- If $u \leq p$ the positive branch is taken;
- If $u > p$ the other branch is selected.

Simulation packages

- One of the benefits of using a simulation package is that at least some of these algorithms are provided for us.
- Each time that a distribution is instantiated the seed for the random number generator can be set explicitly.
- If seeds are not well-spaced there may be overlap between the sequences of random numbers used by the generators resulting in correlation between the samples used in the simulation.
- Some simulation packages provide an automatic seeding mechanism which will seed each distribution with a distinct seed which is far in the cycle from other seeds currently in use.

Simulation output analysis

- Our objective in constructing a simulation model is to generate one or more performance measures for the system.

Simulation output analysis

- Our objective in constructing a simulation model is to generate one or more performance measures for the system.
- In the Markov models such measures were derived from the steady state probability distribution, after the model solution.

Simulation output analysis

- Our objective in constructing a simulation model is to generate one or more performance measures for the system.
- In the Markov models such measures were derived from the steady state probability distribution, after the model solution.
- In contrast, in a simulation model measures are observed or evaluated directly during the execution of the model.

Simulation output analysis

- Our objective in constructing a simulation model is to generate one or more performance measures for the system.
- In the Markov models such measures were derived from the steady state probability distribution, after the model solution.
- In contrast, in a simulation model measures are observed or evaluated directly during the execution of the model.
- It is part of model construction to make sure that all the necessary counters and updates are in place to allow the measures to be collected as the model runs.

Simulation output analysis

- Our objective in constructing a simulation model is to generate one or more performance measures for the system.
- In the Markov models such measures were derived from the steady state probability distribution, after the model solution.
- In contrast, in a simulation model measures are observed or evaluated directly during the execution of the model.
- It is part of model construction to make sure that all the necessary counters and updates are in place to allow the measures to be collected as the model runs.
- This is sometimes called instrumentation of a model as it is analogous to inserting probes and monitors on a real system.

Simulation trajectories

It is important to remember that each run of a model constitutes a single trajectory over the state space.

Simulation trajectories

It is important to remember that each run of a model constitutes a single trajectory over the state space.

So, in general, any estimate for the value of a performance measure generated from a single run constitutes a single observation in the possible sample space.

Simulation and long-term averages

- To gain an accurate measure of the performance of the system we should not base our results on a single observation.

Simulation and long-term averages

- To gain an accurate measure of the performance of the system we should not base our results on a single observation.
- For steady state analysis the averages we calculate from data collected during execution will always be an approximation of the unknown true long-term averages that characterise the system performance.

Simulation and long-term averages

- To gain an accurate measure of the performance of the system we should not base our results on a single observation.
- For steady state analysis the averages we calculate from data collected during execution will always be an approximation of the unknown true long-term averages that characterise the system performance.
- Important issues are:

Simulation and long-term averages

- To gain an accurate measure of the performance of the system we should not base our results on a single observation.
- For steady state analysis the averages we calculate from data collected during execution will always be an approximation of the unknown true long-term averages that characterise the system performance.
- Important issues are:
  - choosing the starting state of the simulation;

Simulation and long-term averages

- To gain an accurate measure of the performance of the system we should not base our results on a single observation.
- For steady state analysis the averages we calculate from data collected during execution will always be an approximation of the unknown true long-term averages that characterise the system performance.
- Important issues are:
  - choosing the starting state of the simulation;
  - choosing the warm-up period that is allowed to elapse before data collection begins;

Simulation and long-term averages

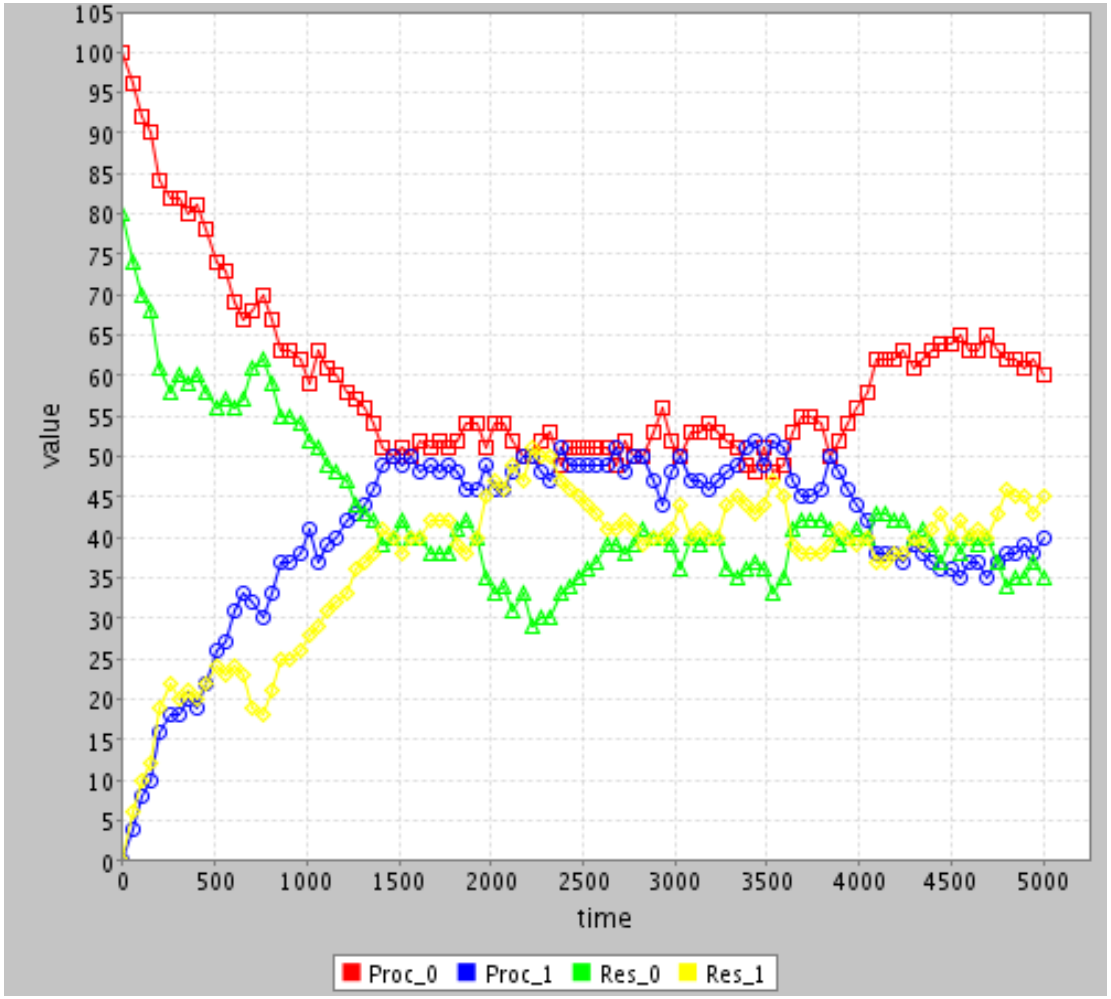
- To gain an accurate measure of the performance of the system we should not base our results on a single observation.
- For steady state analysis the averages we calculate from data collected during execution will always be an approximation of the unknown true long-term averages that characterise the system performance.
- Important issues are:
  - choosing the starting state of the simulation;
  - choosing the warm-up period that is allowed to elapse before data collection begins;
  - choosing a run length that ensures that the calculated averages are representative of the unknown true long term average.

Example

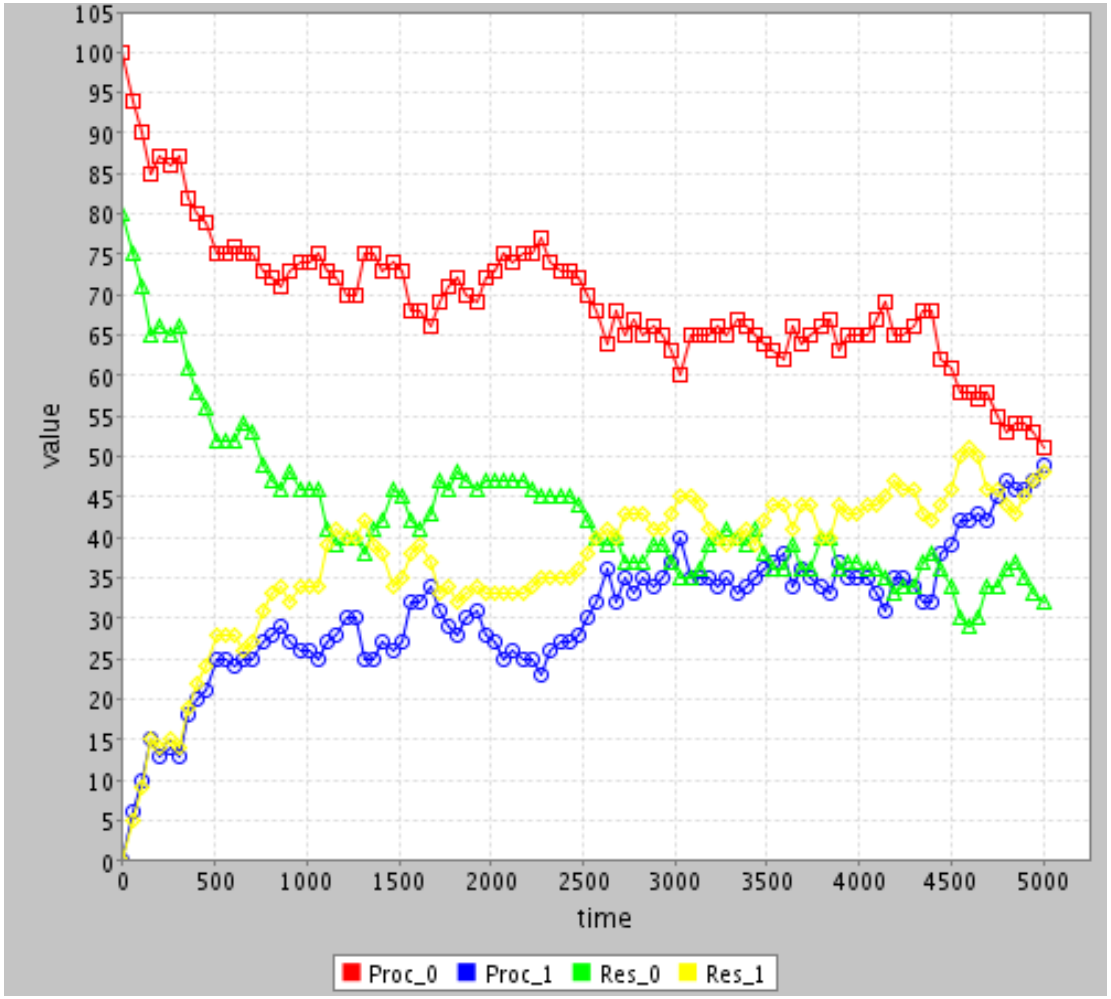
$$
\begin{aligned}
Proc_0 &\stackrel{def}{=} (task1, r_1).Proc_1 \\
Proc_1 &\stackrel{def}{=} (task2, r_2).Proc_0 \\
Res_0 &\stackrel{def}{=} (task1, r_3).Res_1 \\
Res_1 &\stackrel{def}{=} (reset, r_4).Res_0
\end{aligned}
$$

$$
Proc_0[N_P] \underset{\{task1\}}{\bowtie} Res_0[N_R]
$$

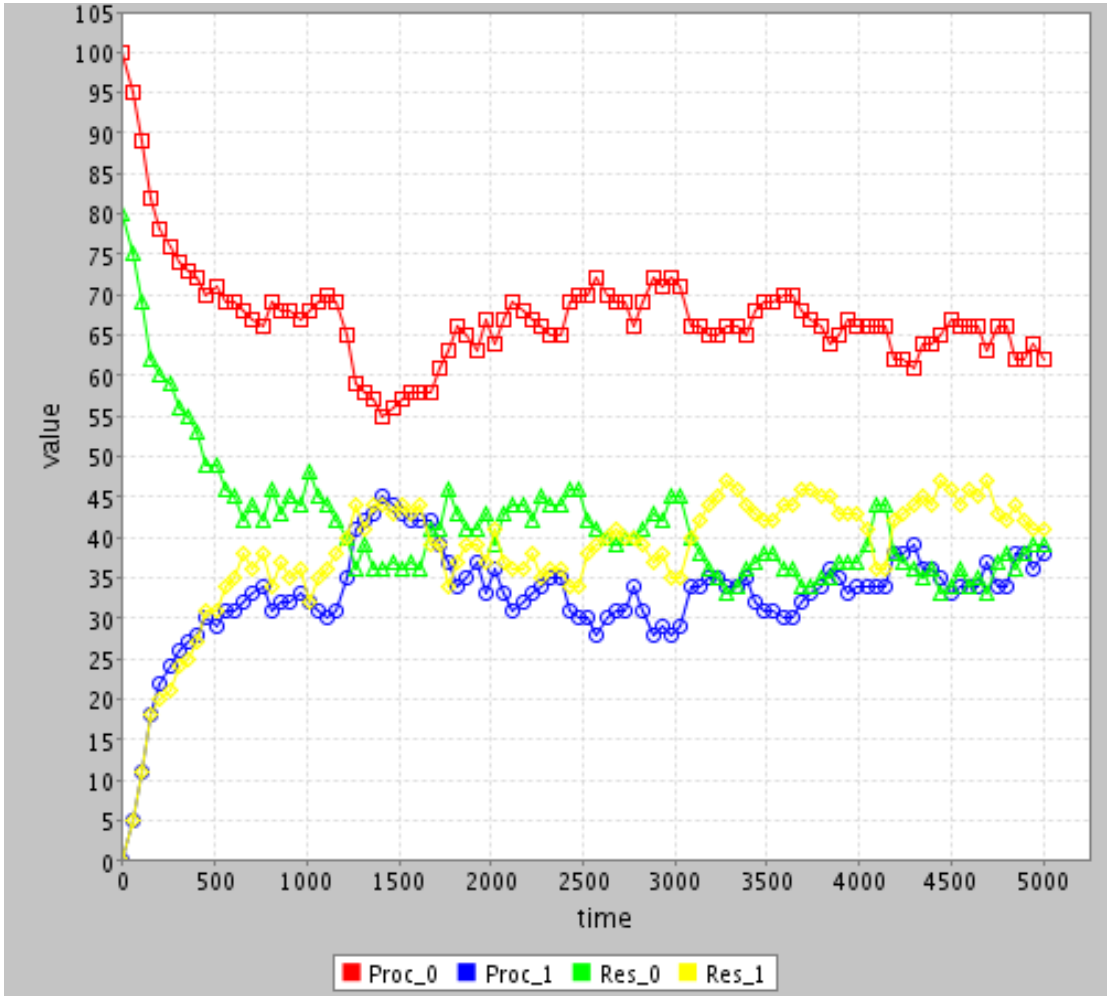
100 processors and 80 resources (simulation run A)



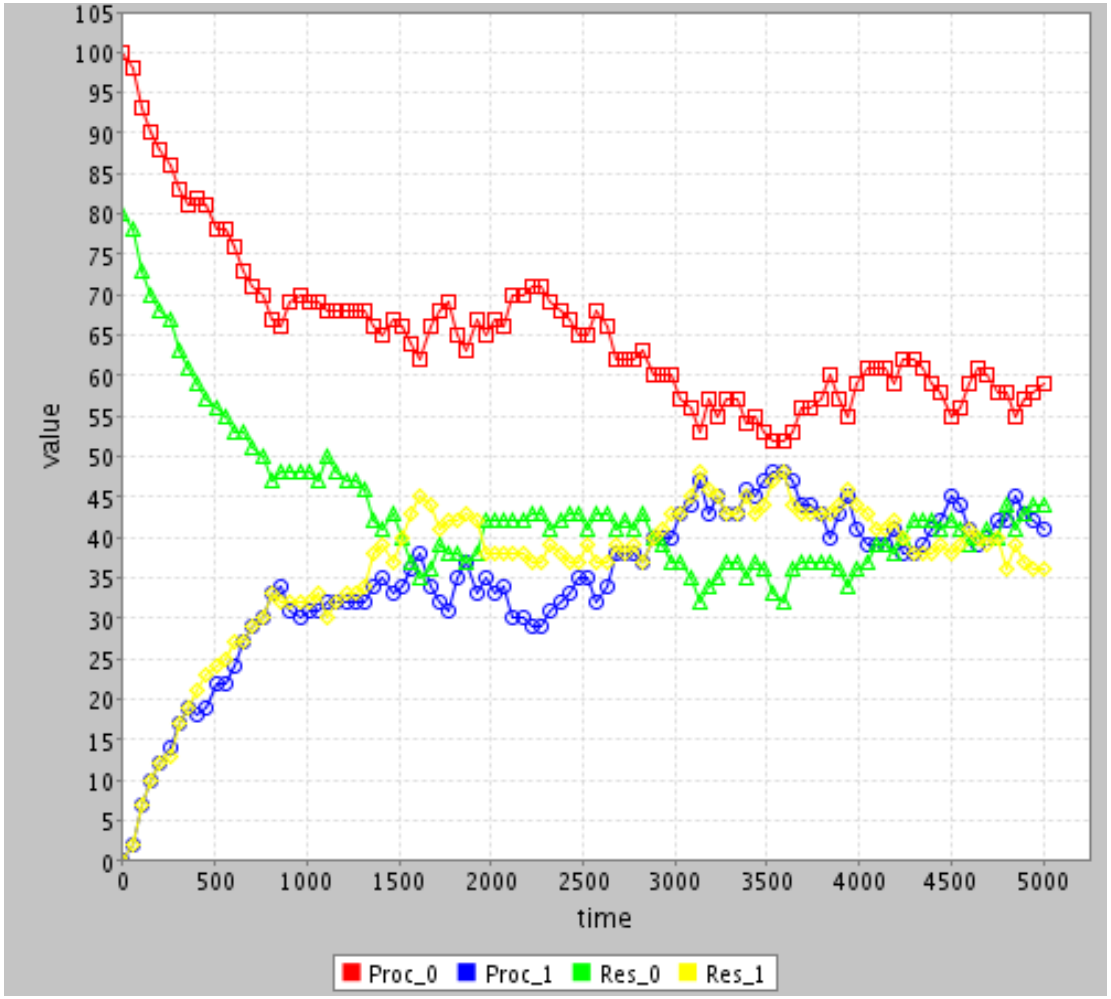
100 processors and 80 resources (simulation run B)



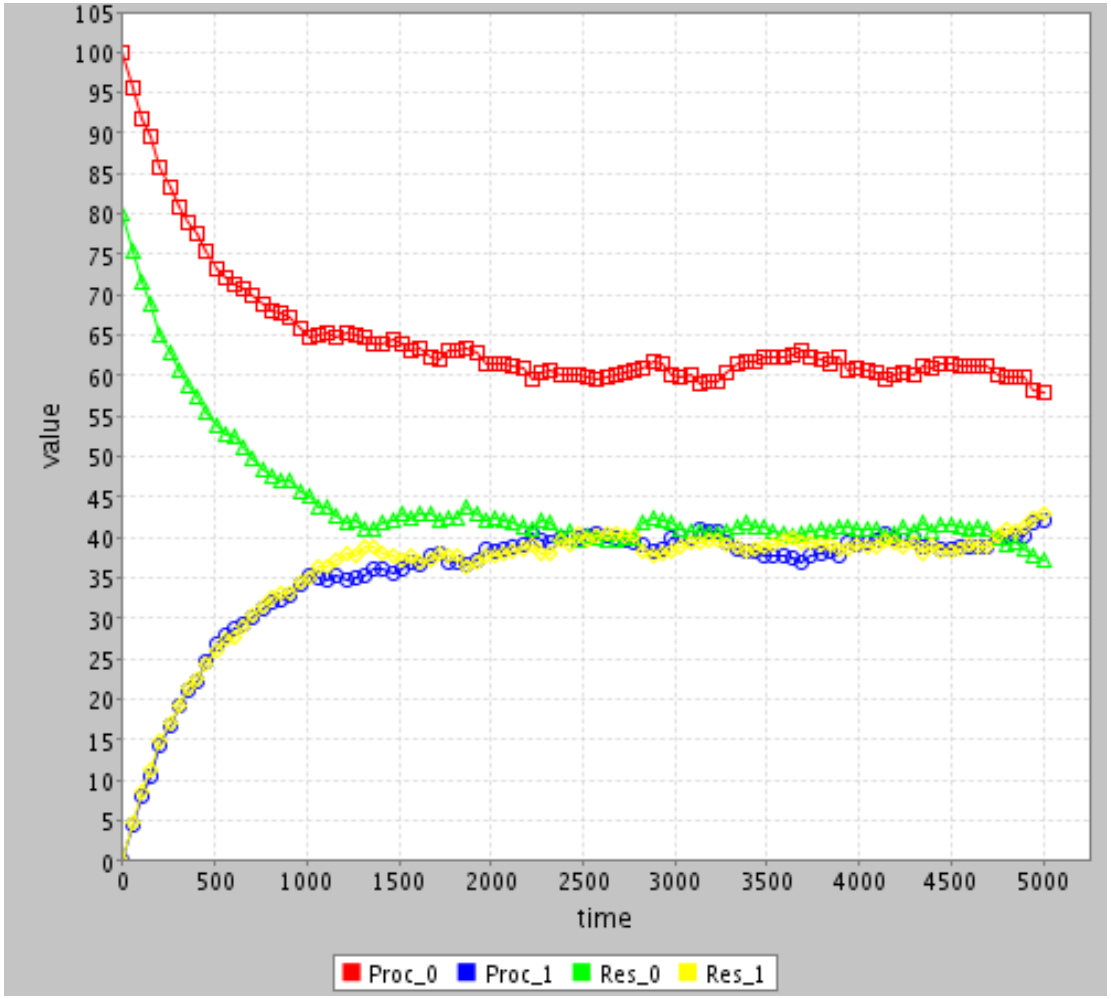
100 processors and 80 resources (simulation run C)



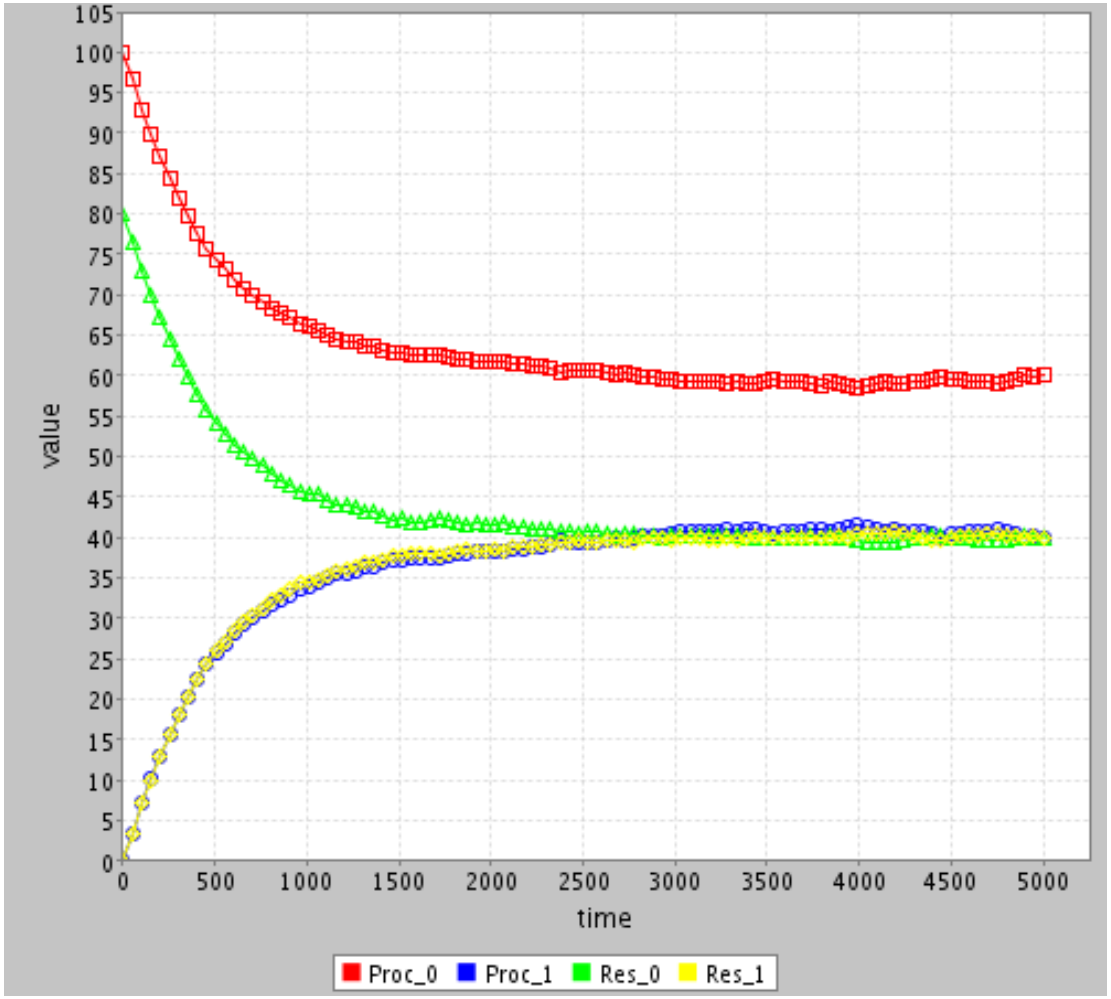
100 processors and 80 resources (simulation run D)



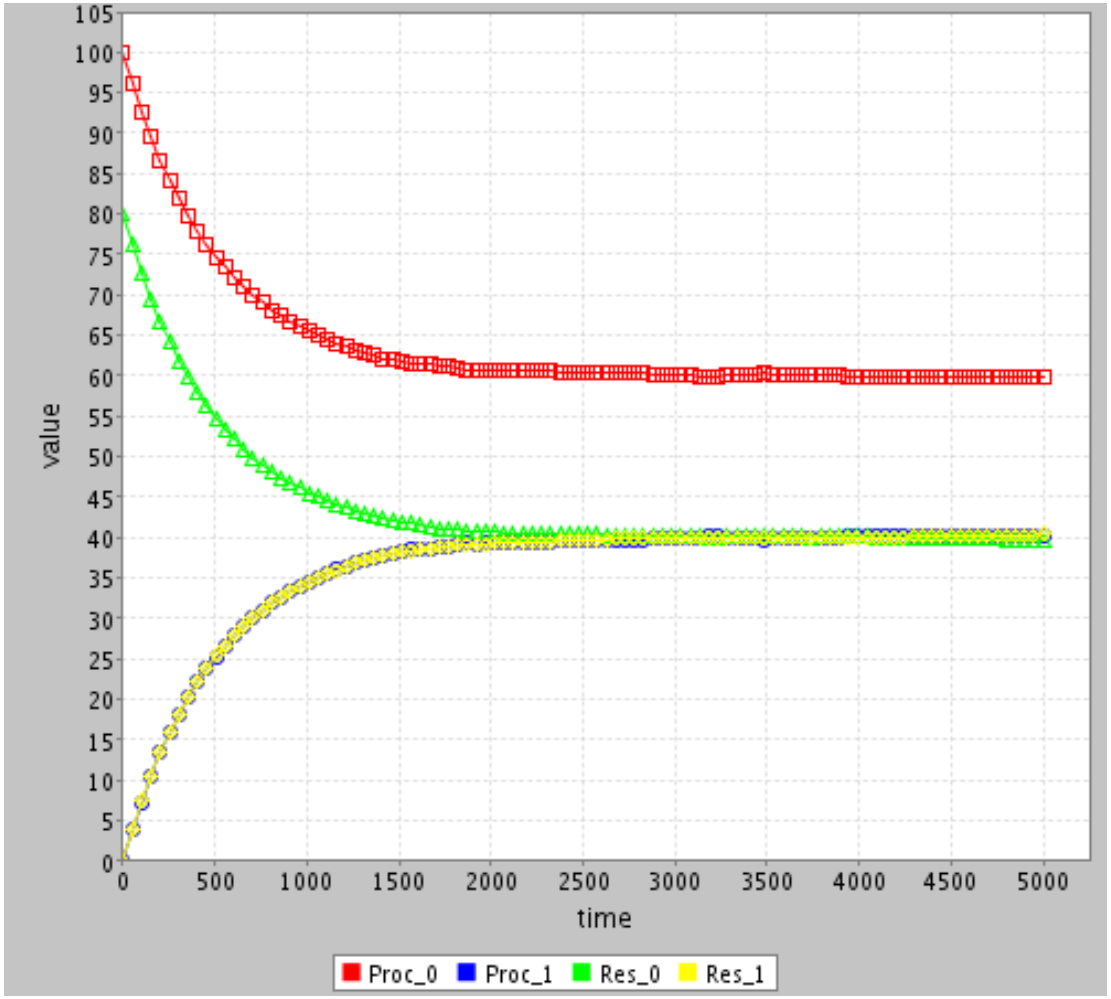
100 processors and 80 resources (average of 10 runs)



100 Processors and 80 resources (average of 100 runs)



100 processors and 80 resources (average of 1000 runs)



Statistical techniques

- Statistical techniques can be used to assess how and when the calculated averages approximate the true average, i.e. to analyse the accuracy of our current estimate.
- This is often done in terms of a confidence interval.
- A confidence interval expresses probabilistic bounds on the error of our current estimate.

Confidence intervals

A confidence interval (c_1, c_2) with confidence level $X\%$, means that with probability $X/100$ the real value v lies between the values c_1 and c_2, i.e.

$$\Pr(c_1 \leq v \leq c_2) = X/100$$

Confidence intervals

A confidence interval (c_1, c_2) with confidence level $X\%$, means that with probability $X/100$ the real value v lies between the values c_1 and c_2, i.e.

$$\Pr(c_1 \leq v \leq c_2) = X/100$$

$X/100$ is usually written in the form $1 - \alpha$, and α is called the significance level, and $(1 - \alpha)$ is called the confidence coefficient.

Confidence intervals and variance

Usually performance modellers will run their simulation models until their observations give them confidence levels of 90% or 95% and a confidence interval which is acceptably tight.

Confidence intervals and variance

Usually performance modellers will run their simulation models until their observations give them confidence levels of 90% or 95% and a confidence interval which is acceptably tight.

Calculation of the confidence interval is based on the variance within the observations which have been gathered.

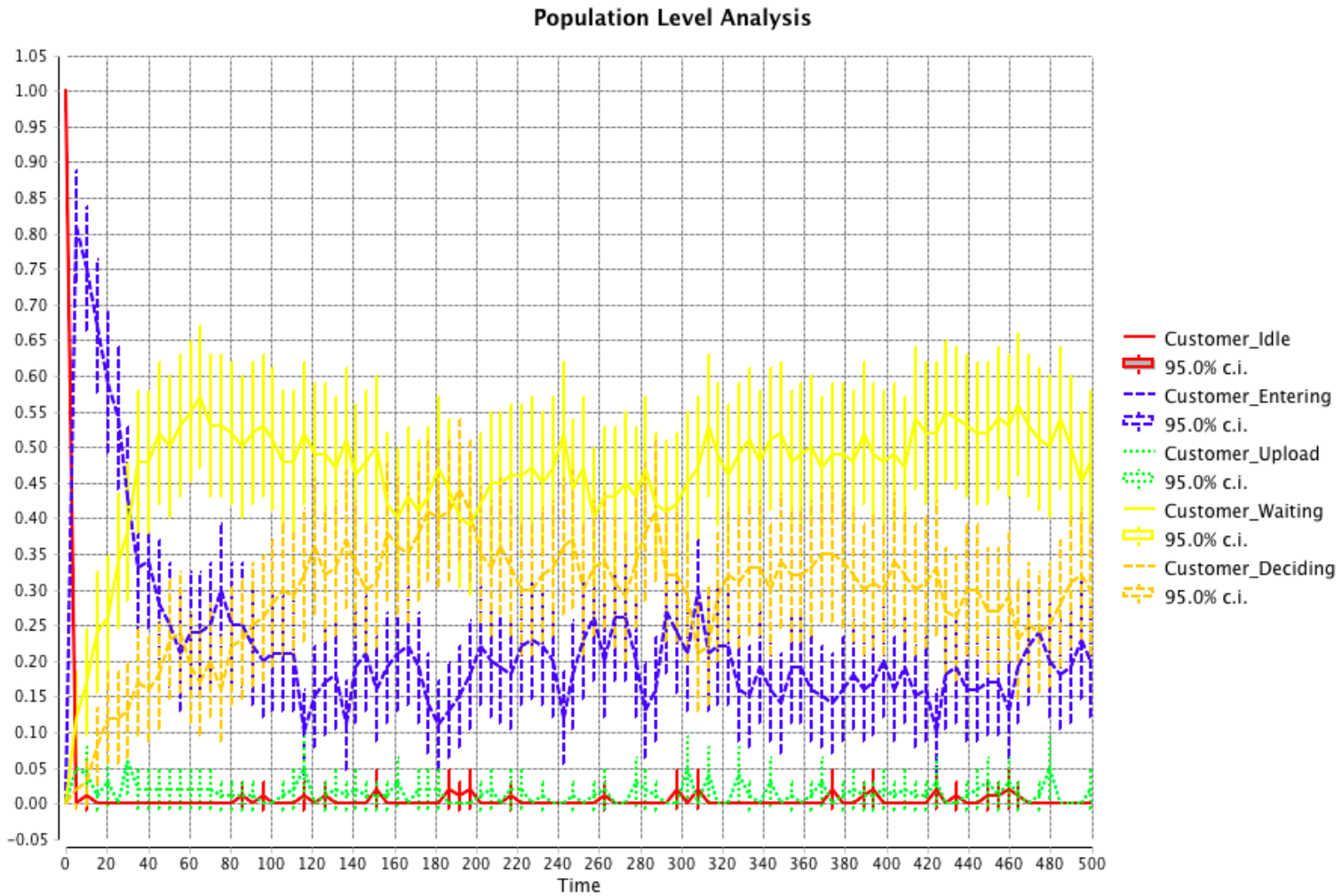
Confidence intervals and variance

Usually performance modellers will run their simulation models until their observations give them confidence levels of 90% or 95% and a confidence interval which is acceptably tight.

Calculation of the confidence interval is based on the variance within the observations which have been gathered.

The greater the variance, the wider the confidence interval; the smaller the variance, the tighter the bounds.

Confidence interval example



Length of simulation runs

For some modelling studies the length of time for which a simulation model should be run is defined by the problem itself.

Length of simulation runs

For some modelling studies the length of time for which a simulation model should be run is defined by the problem itself.

For example, if we wish to investigate how many messages can be processed by a dealers' transaction processing system in the first hour of trading then it makes sense to run the model for 3600 seconds.

Length of simulation runs

For some modelling studies the length of time for which a simulation model should be run is defined by the problem itself.

For example, if we wish to investigate how many messages can be processed by a dealers' transaction processing system in the first hour of trading then it makes sense to run the model for 3600 seconds.

However, if the question is how many messages can be processed in an average hour then running the model for 3600 seconds is unlikely to be enough.

Terminating simulations and cold-start

The first question ("the first hour") identifies the simulation as a transient or terminating simulation.

Terminating simulations and cold-start

The first question ("the first hour") identifies the simulation as a transient or terminating simulation.

It is said to have a cold-start: the system is initially empty which is not its usual state but we still include this data in the observation period.

Terminating simulations and cold-start

The first question ("the first hour") identifies the simulation as a transient or terminating simulation.

It is said to have a cold-start: the system is initially empty which is not its usual state but we still include this data in the observation period.

For this type of simulation the question becomes how many times the simulation must be repeated (with different random number streams) to achieve a required confidence interval.

Steady-state behaviour

In the second scenario ("an average hour") we are interested in the steady state behaviour of the system.

Steady-state behaviour

In the second scenario ("an average hour") we are interested in the steady state behaviour of the system.

As in Markovian modelling we associate steady state behaviour with long term behaviour.

Steady-state behaviour

In the second scenario ("an average hour") we are interested in the steady state behaviour of the system.

As in Markovian modelling we associate steady state behaviour with long term behaviour.

In other words we are theoretically interested in the observations obtained from runs of the model which are infinitely long.

Steady-state behaviour

In the second scenario ("an average hour") we are interested in the steady state behaviour of the system.

As in Markovian modelling we associate steady state behaviour with long term behaviour.

In other words we are theoretically interested in the observations obtained from runs of the model which are infinitely long.

However, in practice we are interested in finite run lengths and estimating the steady state distribution of the measures we are interested in from finitely many samples.

Initial conditions, bias

The initial conditions or starting state of a model influence the sequence of states seen in the simulation, especially early in a run.

Initial conditions, bias

The initial conditions or starting state of a model influence the sequence of states seen in the simulation, especially early in a run.

In a steady state distribution the output values should be independent of the starting state.

Initial conditions, bias

The initial conditions or starting state of a model influence the sequence of states seen in the simulation, especially early in a run.

In a steady state distribution the output values should be independent of the starting state.

Thus the modeller must make some effort to remove the effect of the starting state, sometimes termed bias, from the sample data used for estimating the performance measure of interest.

Initial conditions, bias

The initial conditions or starting state of a model influence the sequence of states seen in the simulation, especially early in a run.

In a steady state distribution the output values should be independent of the starting state.

Thus the modeller must make some effort to remove the effect of the starting state, sometimes termed bias, from the sample data used for estimating the performance measure of interest.

Unfortunately there is no precise procedure for this as we cannot generally detect when the model has moved from transient behaviour (the warm-up period) to steady state behaviour.

Heuristics for reducing bias

The common techniques are

1. Long runs.
2. Proper initialisation.
3. Truncation.
4. Initial data deletion.
5. Moving average of independent replications.
6. Batch means.

Heuristics for reducing bias

The common techniques are

1. Long runs.
2. Proper initialisation.
3. Truncation.
4. Initial data deletion.
5. Moving average of independent replications.
6. Batch means.

The last four techniques are all based on the assumption that variability is less during steady state behaviour than during transient behaviour.

Options for terminating a simulation

Option 1

- begin the simulation at time 0
- begin data collection at specified time $w \geq 0$
- complete data collection at specified time $w + t$
- terminate execution of the simulation at time $w + t$
- calculate summary statistics based on sample path data collected in the time interval $(w, w + t)$.

Options for terminating a simulation

Option 2

- begin the simulation at time 0
- begin data collection when the Mth event completes
- complete data collection when the $(M+N)$th event completes
- terminate execution of the simulation when the $(M+N)$th event completes
- calculate summary statistics based on sample path data collected in the time interval (t_M, t_{M+N}), where t_j is the time at which the jth event completes.

Advantages and disadvantages

- Option 1 implies that the simulated time $(w, w + t)$ for data collection is predetermined but the number of event completions is random.
- Conversely, Option 2 implies that the time period for data collection is random but the number of event completions is predetermined.
- In queueing networks, Option 1 is preferable for calculating queue lengths and resource utilisations, whereas Option 2 is preferable for calculating waiting times.

Variance reduction techniques

- Assume that we are running a simulation model in order to estimate some performance measure M.
- During the ith execution of the model we make observations of M, o_{ij} and at the end of the run we calculate the mean value of the observations O_i.
- Note that the observations o_{ij} in most simulations are not independent. Successive observations are often correlated.

Example of correlation

- If we are interested in the delay of messages in a packet-switching network, if the delay of one message is long because the network is heavily congested, the next message is likely to be similarly delayed.

Example of correlation

- If we are interested in the delay of messages in a packet-switching network, if the delay of one message is long because the network is heavily congested, the next message is likely to be similarly delayed.
- Thus the two observations are not independent.

Example of correlation

- If we are interested in the delay of messages in a packet-switching network, if the delay of one message is long because the network is heavily congested, the next message is likely to be similarly delayed.
- Thus the two observations are not independent.

Note

This is why, in general, a simulation model must be run several times.

Independent replications

- If independent replications are used the model is run m times in order to generate m independent observations.

Independent replications

- If independent replications are used the model is run m times in order to generate m independent observations.
- For the runs to be independent, the random number generator seeds must be carefully chosen.

Independent replications

- If independent replications are used the model is run m times in order to generate m independent observations.
- For the runs to be independent, the random number generator seeds must be carefully chosen.
- If steady state or long term behaviour is being investigated the data relating to the warm-up period must be discarded.

Independent replications

- If independent replications are used the model is run m times in order to generate m independent observations.
- For the runs to be independent, the random number generator seeds must be carefully chosen.
- If steady state or long term behaviour is being investigated the data relating to the warm-up period must be discarded.
- Let O denote the mean value of the retained observations, O_i, after m runs.

Independent replications

- If independent replications are used the model is run m times in order to generate m independent observations.
- For the runs to be independent, the random number generator seeds must be carefully chosen.
- If steady state or long term behaviour is being investigated the data relating to the warm-up period must be discarded.
- Let O denote the mean value of the retained observations, O_i, after m runs.
- The variance over all observations is calculated as:

$$V = \frac{1}{m-1} \sum_{i=1}^{m} (O_i - O)^2$$

Independent replications and steady-state

For steady-state analysis independent replication is an inefficient way to generate samples, since for each sample point, O_i, k observations, $\{o_{i1}, \ldots, o_{ik}\}$, must be discarded.

Batch means

- In the method of batch means the model is run only once but for a very long period.

Batch means

- In the method of batch means the model is run only once but for a very long period.
- The run is divided into a series of sub-periods of length ℓ, and measures over each sub-run form a single point estimate.

Batch means

- In the method of batch means the model is run only once but for a very long period.
- The run is divided into a series of sub-periods of length ℓ, and measures over each sub-run form a single point estimate.
- If the observations made during the run form a set $\{o_i\}$, the set is partitioned into subsets

$$S_i = \{o_j \mid o_j \text{ observed between } (i-1) \times \ell \text{ and } i \times \ell\}$$

Batch means

- In the method of batch means the model is run only once but for a very long period.
- The run is divided into a series of sub-periods of length ℓ, and measures over each sub-run form a single point estimate.
- If the observations made during the run form a set $\{o_i\}$, the set is partitioned into subsets

$$S_i = \{o_j \mid o_j \text{ observed between } (i-1) \times \ell \text{ and } i \times \ell\}$$

- Each sample point O_i is the mean generated from a subset of observations S_i, and O is the mean generated from the O_i.

Batch means

- In the method of batch means the model is run only once but for a very long period.
- The run is divided into a series of sub-periods of length ℓ, and measures over each sub-run form a single point estimate.
- If the observations made during the run form a set $\{o_i\}$, the set is partitioned into subsets

$$S_i = \{o_j \mid o_j \text{ observed between } (i-1) \times \ell \text{ and } i \times \ell\}$$

- Each sample point O_i is the mean generated from a subset of observations S_i, and O is the mean generated from the O_i.
- Variance is calculated as above.

Batch means and independence

This method is unreliable since the sub-periods are clearly not independent.

Batch means and independence

This method is unreliable since the sub-periods are clearly not independent.

However it has the advantage that only one set of observations $\{o_i \ldots o_k\}$ needs to be discarded to overcome the warm-up effects in steady state analysis.

Regeneration

- It is sometimes possible within the run of a simulation model to identify points in the trajectory where the model returns to exactly equivalent states: so-called regeneration points.

Regeneration

- It is sometimes possible within the run of a simulation model to identify points in the trajectory where the model returns to exactly equivalent states: so-called regeneration points.
- Periods between regeneration points are genuinely independent sub-runs, e.g. a queue which empties.

Regeneration

- It is sometimes possible within the run of a simulation model to identify points in the trajectory where the model returns to exactly equivalent states: so-called regeneration points.
- Periods between regeneration points are genuinely independent sub-runs, e.g. a queue which empties.
- The behaviour of the model (queue length, waiting time etc) after a visit to such a state does not depend on the previous history of the model in any way.

Regeneration

- It is sometimes possible within the run of a simulation model to identify points in the trajectory where the model returns to exactly equivalent states: so-called regeneration points.
- Periods between regeneration points are genuinely independent sub-runs, e.g. a queue which empties.
- The behaviour of the model (queue length, waiting time etc) after a visit to such a state does not depend on the previous history of the model in any way.
- The duration between two successive regeneration points is called a regeneration cycle.

Variance computation and regeneration

- The variance computation using regeneration cycles is a bit more complex than that in the method of batch means or the method of independent replications.
- This is because the regeneration cycles are of different lengths, whereas in the other two methods the batches or replications are all of the same length.

Regeneration considered

Unlike the previous two methods, the method of regeneration does not require any transient observations to be removed.

Regeneration considered

Unlike the previous two methods, the method of regeneration does not require any transient observations to be removed.

Unfortunately not all models have easily defined regeneration states, and even when they exist they can be computationally expensive to identify.

Regeneration considered

Unlike the previous two methods, the method of regeneration does not require any transient observations to be removed.

Unfortunately not all models have easily defined regeneration states, and even when they exist they can be computationally expensive to identify.

Another disadvantage is that it is not possible to define the length of a simulation run beforehand.