

4.1 Sample Statistics

Discrete-event simulations generate experimental data — frequently *a lot* of experimental data. To facilitate the analysis of all this data, it is conventional to compress the data into a handful of meaningful statistics. We have already seen examples of this in Sections 1.2 and 3.1, where job averages and time averages were used to characterize the performance of a single-server service node. Similarly, in Sections 1.3 and 3.1, averages were used to characterize the performance of a simple inventory system. This kind of “within-the-run” data generation and statistical computation (see programs `ssq2` and `sis2`, for example) is common in discrete-event simulation. Although less common, there is also a “between-the-runs” kind of statistical analysis commonly used in discrete-event simulation. That is, a discrete-event simulation program can be used to simulate the same stochastic system repeatedly — all you need to do is change the initial seed for the random number generator from run to run. This process is known as *replication*.

In either case, each time a discrete-event simulation program is used to generate data it is important to appreciate that this data is only a *sample* from that much larger *population* that would be produced, in the first case, if the program were run for a much longer time or, in the second case, if more replications were used. Analyzing a sample and then inferring something about the population from which the sample was drawn is the essence of statistics. We begin by defining the sample mean and standard deviation.

4.1.1 SAMPLE MEAN AND STANDARD DEVIATION

Definition 4.1.1 Given the sample x_1, x_2, \dots, x_n (either continuous or discrete data) the *sample mean* is

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i.$$

The *sample variance* is the average of the squared differences about the sample mean

$$s^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2.$$

The *sample standard deviation* is the positive square root of the sample variance, $s = \sqrt{s^2}$.

The sample mean is a measure of *central tendency* of the data values. The sample variance and standard deviation are measures of *dispersion* — the spread of the data about the sample mean. The sample standard deviation has the same “units” as the data and the sample mean. For example, if the data has units of *sec* then so also does the sample mean and standard deviation. The sample variance would have units of *sec*². Although the sample variance is more amenable to mathematical manipulation (because it is free of the square root), the sample standard deviation is typically the preferred measure of dispersion since it has the same units as the data. (Because \bar{x} and s have the same units, the ratio s/\bar{x} , known as the *coefficient of variation*, has no units. This statistic is commonly used only if the data is inherently non-negative, although it is inferior to s as a measure of dispersion because a common shift in the data results in a change in s/\bar{x} .)

For statistical reasons that will be explained further in Chapter 8, a common alternative definition of the sample variance (and thus the sample standard deviation) is

$$\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2 \quad \text{rather than} \quad \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2.$$

Provided one of these definitions is used consistently for *all* computations, the choice of which equation to use for s^2 is largely a matter of taste, based on statistical considerations listed below. There are three reasons that the $1/(n-1)$ form of the sample variance appears almost universally in introductory statistics books:

- The sample variance is undefined when $n = 1$ when using the $1/(n-1)$ form. This is intuitive in the sense that a single observed data value indicates nothing about the spread of the distribution from which the data value is drawn.
- The $1/(n-1)$ form of the sample variance is an *unbiased estimate* of the population variance when the data values are drawn independently from a population with a finite mean and variance. The designation of an estimator as “unbiased” in statistical terminology implies that the sample average of many such sample variances converges to the population variance.

- The statistic

$$\sum_{i=1}^n (x_i - \bar{x})^2$$

has $n-1$ “degrees of freedom”. It is common practice in statistics (particularly in a sub-field known as *analysis of variance*) to divide a statistic by its degrees of freedom.

Despite these compelling reasons, why do we still use the $1/n$ form of the sample variance as given in Definition 4.1.1 consistently throughout the book? Here are five reasons:

- The sample size n is typically *large* in discrete-event simulation, making the difference between the results obtained by using the two definitions small.
- The unbiased property associated with the $1/(n-1)$ form of the sample variance applies only when observations are independent. Observations within a simulation run are typically not independent in discrete-event simulation.
- In the unlikely case that only $n = 1$ observation is collected, the algorithms presented in this text using the $1/n$ form are able to compute a numeric value (zero) for the sample variance. The reader should understand that a sample variance of zero in the case of $n = 1$ does *not* imply that there is no variability in the population.
- The $1/n$ form of the sample variance enjoys the status as a “plug-in” (details given in Chapter 6) estimate of the population variance. It also is the “maximum likelihood estimate” (details given in Chapter 9) when sampling from bell-shaped populations.

- Authors of many higher-level books on mathematical statistics prefer the $1/n$ form of the sample variance.

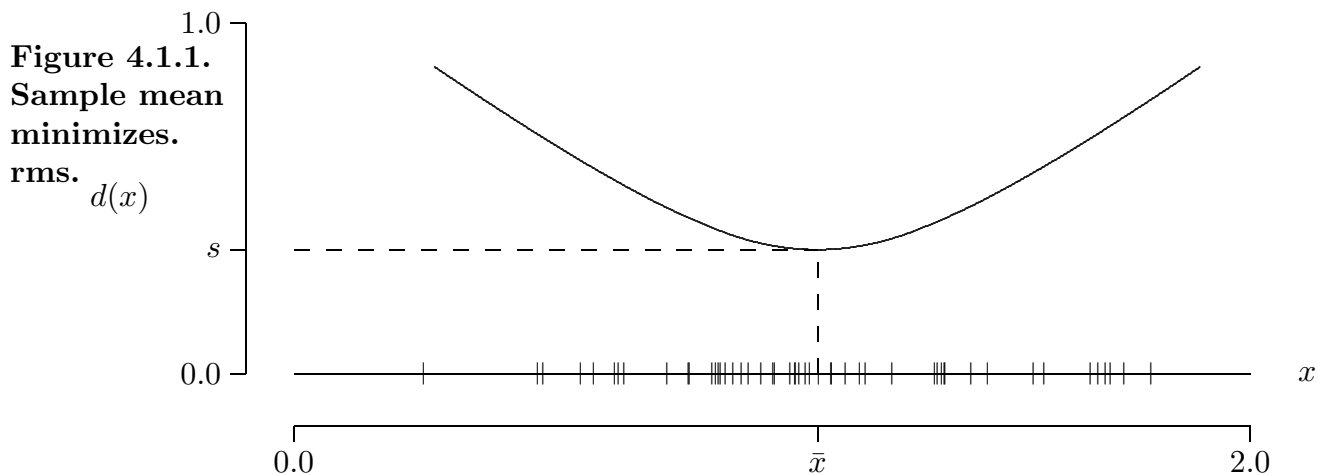
The relationship between the sample mean and sample standard deviation is summarized by Theorem 4.1.1. The proof is left as an exercise.

Theorem 4.1.1 In a *root-mean-square* (rms) sense, the sample mean is that unique value that best fits the sample x_1, x_2, \dots, x_n and the sample standard deviation is the corresponding smallest possible rms value. In other words, if the rms value associated with any value of x is

$$d(x) = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - x)^2}$$

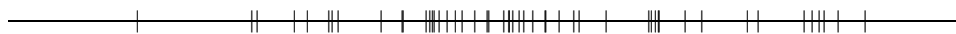
then the smallest possible rms value of $d(x)$ is $d(\bar{x}) = s$ and this value is achieved if and only if $x = \bar{x}$.

Example 4.1.1 Theorem 4.1.1 is illustrated in Figure 4.1.1 for a random variate sample of size $n = 50$ that was generated using a modified version of program `buffon`. The data corresponds to 50 observations of the x -coordinate of the righthand endpoint of a unit-length needle dropped at random (see Example 2.3.10). The 50 points in the sample are indicated with hash marks.



For this sample $\bar{x} \cong 1.095$ and $s \cong 0.354$. Consistent with Theorem 4.1.1 the smallest value of $d(x)$ is $d(\bar{x}) = s$ as illustrated.

The portion of the figure in Figure 4.1.1 that looks like



is known as a univariate *scatter diagram* — a convenient way to visualize a sample, provided the sample size is not too large. Later in this chapter we will discuss *histograms* — a superior way to visualize a univariate sample if the sample size is sufficiently large.

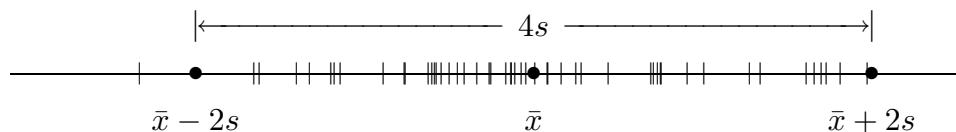
Chebyshev's Inequality

To better understand how the sample mean and standard deviation are related, appreciate that the number of points in a sample that lie within k standard deviations of the mean can be bounded by Chebyshev's inequality. The derivation of this fundamental result is based on a simple observation — the points that make the largest contribution to the sample standard deviation are those that are most distant from the sample mean.

Given the sample $\mathcal{S} = \{x_1, x_2, \dots, x_n\}$ with mean \bar{x} and standard deviation s , and given a parameter $k > 1$, define the set*

$$\mathcal{S}_k = \{x_i \mid \bar{x} - ks < x_i < \bar{x} + ks\}$$

consisting of all those $x_i \in \mathcal{S}$ for which $|x_i - \bar{x}| < ks$. For $k = 2$ and the sample in Example 4.1.1, the set \mathcal{S}_2 is defined by the hash marks that lie within the $2ks = 4s$ wide interval centered on \bar{x} as shown by the portion of Figure 4.1.1 given below.



Let $p_k = |\mathcal{S}_k|/n$ be the *proportion* of x_i that lie within $\pm ks$ of \bar{x} . This proportion is the probability that an x_i selected at random from the sample will be in \mathcal{S}_k . From the definition of s^2

$$ns^2 = \sum_{i=1}^n (x_i - \bar{x})^2 = \sum_{x_i \in \mathcal{S}_k} (x_i - \bar{x})^2 + \sum_{x_i \in \bar{\mathcal{S}}_k} (x_i - \bar{x})^2$$

where the set $\bar{\mathcal{S}}_k$ is the complement of \mathcal{S}_k . If the contribution to s^2 from all the points close to the sample mean (the points in \mathcal{S}_k) is ignored then, because the contribution to s^2 from each point in $\bar{\mathcal{S}}_k$ (the points far from the sample mean) is at least $(ks)^2$, the previous equation becomes the following inequality

$$ns^2 \geq \sum_{x_i \in \bar{\mathcal{S}}_k} (x_i - \bar{x})^2 \geq \sum_{x_i \in \bar{\mathcal{S}}_k} (ks)^2 = |\bar{\mathcal{S}}_k|(ks)^2 = n(1 - p_k)k^2s^2.$$

If ns^2 is eliminated the result is *Chebyshev's inequality*

$$p_k \geq 1 - \frac{1}{k^2} \quad (k > 1).$$

(This inequality says *nothing* for $k \leq 1$.) From this inequality it follows that, in particular, $p_2 \geq 0.75$. That is, *for any sample* at least 75% of the points in the sample must lie within $\pm 2s$ of the sample mean.

* Some values in the sample may occur more than once. Therefore, \mathcal{S} is actually a *multiset*. For the purposes of counting, however, the elements of \mathcal{S} are treated as distinguishable so that the cardinality (size) of \mathcal{S} is $|\mathcal{S}| = n$.

For the data illustrated by the scatter diagram the true percentage of points within $\pm 2s$ of the sample mean is 98%. That is, for the sample in Example 4.1.1, as for most samples, the 75% in the $k = 2$ form of Chebyshev's inequality is very conservative. Indeed, it is common to find approximately 95% of the points in a sample within $\pm 2s$ of the sample mean. (See Exercise 4.1.7.)

The primary issue here is not the accuracy (or lack thereof) of Chebyshev's inequality. Instead, Chebyshev's inequality and practical experience with actual data suggest that the $\bar{x} \pm 2s$ interval defines the "effective width" of a sample.* As a rule of thumb, most but not all of the points in a sample will fall within this interval. With this in mind, when analyzing experimental data it is common to look for *outliers* — values so far from the sample mean, for example $\pm 3s$ or more — that they must be viewed with suspicion.

Linear Data Transformations

It is sometimes the case that the output data generated by a discrete-event simulation, for example the wait times in a single-server service node, are statistically analyzed in one system of units (say seconds) and later there is a need to convert to a different system of units (say minutes). Usually this conversion of units is a *linear* data transformation and, if so, the change in system statistics can be determined directly, without any need to re-process the converted data.

That is, if $x'_i = ax_i + b$ for $i = 1, 2, \dots, n$ then how do the sample mean \bar{x} and standard deviation s of the x -data relate to the sample mean \bar{x}' and standard deviation s' of the x' -data? The answer is that

$$\bar{x}' = \frac{1}{n} \sum_{i=1}^n x'_i = \frac{1}{n} \sum_{i=1}^n (ax_i + b) = \frac{a}{n} \left(\sum_{i=1}^n x_i \right) + b = a\bar{x} + b$$

and

$$(s')^2 = \frac{1}{n} \sum_{i=1}^n (x'_i - \bar{x}')^2 = \frac{1}{n} \sum_{i=1}^n (ax_i + b - a\bar{x} - b)^2 = \frac{a^2}{n} \sum_{i=1}^n (x_i - \bar{x})^2 = a^2 s^2.$$

Therefore

$$\bar{x}' = a\bar{x} + b \quad \text{and} \quad s' = |a|s.$$

Example 4.1.2 Suppose the sample x_1, x_2, \dots, x_n is measured in seconds. To convert to minutes, the transformation is $x'_i = x_i/60$ so that if, say $\bar{x} = 45$ (seconds) and $s = 15$ (seconds), then

$$\bar{x}' = \frac{45}{60} = 0.75 \text{ (minutes)} \quad \text{and} \quad s' = \frac{15}{60} = 0.25 \text{ (minutes)}.$$

* There is nothing magic about the value $k = 2$. Other values like $k = 2.5$ and $k = 3$ are sometimes offered as alternatives to define the $\bar{x} \pm ks$ interval. To the extent that there is a standard, however, $k = 2$ is it. This issue will be revisited in Chapter 8.

Example 4.1.3 Some people, particularly those used to analyzing so-called “normally distributed” data, like to *standardize* the data by subtracting the sample mean and dividing the result by the sample standard deviation. That is, given a sample x_1, x_2, \dots, x_n with sample mean \bar{x} and standard deviation s , the corresponding standardized sample is computed as

$$x'_i = \frac{x_i - \bar{x}}{s} \quad i = 1, 2, \dots, n.$$

It follows that the sample mean of the standardized sample is $\bar{x}' = 0$ and the sample standard deviation is $s' = 1$. Standardization is commonly used to avoid potential numerical problems associated with samples containing very large or very small values.

Nonlinear Data Transformations

Although nonlinear data transformations are difficult to analyze in general, there are times when data is used to create a Boolean (two-state) outcome. That is, the *value* of x_i is not as important as the *effect* — does x_i cause something to occur or not? In particular, if \mathcal{A} is a fixed set and if for $i = 1, 2, \dots, n$ we define the nonlinear data transformation

$$x'_i = \begin{cases} 1 & x_i \in \mathcal{A} \\ 0 & \text{otherwise} \end{cases}$$

then what are \bar{x}' and s' ? To answer this question let the proportion of x_i that fall in \mathcal{A} be

$$p = \frac{\text{the number of } x_i \text{ in } \mathcal{A}}{n}$$

then

$$\bar{x}' = \frac{1}{n} \sum_{i=1}^n x'_i = p$$

and

$$(s')^2 = \frac{1}{n} \sum_{i=1}^n (x'_i - p)^2 = \dots = (1-p)(-p)^2 + p(1-p)^2 = p(1-p).$$

Therefore

$$\bar{x}' = p \quad \text{and} \quad s' = \sqrt{p(1-p)}.$$

Example 4.1.4 For a single-server service node, let $x_i = d_i$ be the delay experienced by the i^{th} job and let \mathcal{A} be the set of positive real numbers. Then x'_i is 1 if and only if $d_i > 0$ and p is the proportion of jobs (out of n) that experience a non-zero delay. If, as in Exercise 1.2.3, $p = 0.723$ then $\bar{x}' = 0.723$ and $s' = \sqrt{(0.723)(0.277)} = 0.448$.

Example 4.1.5 As illustrated in Section 2.3, a Monte Carlo simulation used to estimate a probability ultimately involves the generation of a sequence of 0's and 1's with the probability estimate p being the ratio of the number of 1's to the number of trials. Thus p is the sample mean of this sequence of 0's and 1's and $\sqrt{p(1-p)}$ is the sample standard deviation.

4.1.2 COMPUTATIONAL CONSIDERATIONS

One significant computational drawback to calculating the sample standard deviation using the equation

$$s = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2}$$

is that it requires a *two-pass* algorithm. That is, the sample must be scanned once to calculate the sample mean by accumulating the $\sum x_i$ partial sums, and then scanned a second time to calculate the sample standard deviation by accumulating the $\sum (x_i - \bar{x})^2$ partial sums. This two-pass approach is usually undesirable in discrete-event simulation because it creates a need to temporarily store or re-create the *entire* sample, which is undesirable when n is large.

Conventional One-Pass Algorithm

There is an alternate, mathematically equivalent, equation for s^2 (and thus s) that can be implemented as a *one-pass* algorithm. The derivation of this equation is

$$\begin{aligned} s^2 &= \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2 \\ &= \frac{1}{n} \sum_{i=1}^n (x_i^2 - 2\bar{x}x_i + \bar{x}^2) \\ &= \left(\frac{1}{n} \sum_{i=1}^n x_i^2 \right) - \left(\frac{2}{n} \bar{x} \sum_{i=1}^n x_i \right) + \left(\frac{1}{n} \sum_{i=1}^n \bar{x}^2 \right) \\ &= \left(\frac{1}{n} \sum_{i=1}^n x_i^2 \right) - 2\bar{x}^2 + \bar{x}^2 \\ &= \left(\frac{1}{n} \sum_{i=1}^n x_i^2 \right) - \bar{x}^2. \end{aligned}$$

If this alternate equation is used, the sample mean and standard deviation can be calculated in one pass through the sample by accumulating the $\sum x_i$ and $\sum x_i^2$ partial sums, thereby eliminating the need to store the sample. As each new observation arises in a Monte Carlo or discrete-event simulation, only three memory locations are needed in order to store $\sum x_i$, $\sum x_i^2$, and the number of observations to date. There is, however, a potential problem with this one-pass approach — floating-point round-off error. To avoid overflow, the sums are typically accumulated with floating-point arithmetic, even if the data is integer-valued. Floating-point arithmetic is problematic in this case because the sample variance is ultimately calculated by subtracting two quantities that may be *very* large relative to their difference. That is, if the true value of s is tiny relative to $|\bar{x}|$ then accumulated floating-point round-off error may yield an incorrect value for s .

Fortunately there is an alternate one-pass algorithm (due to B. P. Welford in 1962) that can be used to calculate the sample mean and standard deviation. This algorithm is superior to the conventional one-pass algorithm in that it is much less prone to significant floating-point round-off error. The original algorithm is given in Welford (1962). Chan, Golub, and LeVeque (1983) survey other alternative algorithms. Welford's algorithm is based upon the following definition and associated theorem.

Welford's One-Pass Algorithm

Definition 4.1.2 Given the sample x_1, x_2, x_3, \dots , for $i = 1, 2, 3, \dots$, define the running sample mean and the running sample sum of squared deviations as

$$\begin{aligned}\bar{x}_i &= \frac{1}{i}(x_1 + x_2 + \dots + x_i) \\ v_i &= (x_1 - \bar{x}_i)^2 + (x_2 - \bar{x}_i)^2 + \dots + (x_i - \bar{x}_i)^2\end{aligned}$$

where \bar{x}_i and v_i/i are the sample mean and variance, respectively, of x_1, x_2, \dots, x_i .

Theorem 4.1.2 For $i = 1, 2, 3, \dots$, the variables \bar{x}_i and v_i in Definition 4.1.2 can be computed recursively by

$$\begin{aligned}\bar{x}_i &= \bar{x}_{i-1} + \frac{1}{i}(x_i - \bar{x}_{i-1}) \\ v_i &= v_{i-1} + \left(\frac{i-1}{i}\right)(x_i - \bar{x}_{i-1})^2\end{aligned}$$

with the initial conditions $\bar{x}_0 = 0$ and $v_0 = 0$.

Proof Both equations in this theorem are valid by inspection if $i = 1$. If $i > 1$ then we can write

$$\begin{aligned}i\bar{x}_i &= x_1 + x_2 + \dots + x_i \\ &= (x_1 + x_2 + \dots + x_{i-1}) + x_i \\ &= (i-1)\bar{x}_{i-1} + x_i \\ &= i\bar{x}_{i-1} + (x_i - \bar{x}_{i-1})\end{aligned}$$

from which the first equation follows via division by i . This establishes the first equation in the theorem. As in the prior discussion of the conventional one-pass algorithm, an alternate equation for v_i is

$$v_i = x_1^2 + x_2^2 + \dots + x_i^2 - i\bar{x}_i^2 \quad i = 1, 2, 3, \dots$$

From this alternate equation and the first equation in the theorem, if $i > 1$ we can write

$$\begin{aligned}
v_i &= (x_1^2 + x_2^2 + \cdots + x_{i-1}^2) + x_i^2 - i\bar{x}_i^2 \\
&= v_{i-1} + (i-1)\bar{x}_{i-1}^2 + x_i^2 - i\bar{x}_i^2 \\
&= v_{i-1} - i(\bar{x}_i^2 - \bar{x}_{i-1}^2) + (x_i^2 - \bar{x}_{i-1}^2) \\
&= v_{i-1} - i(\bar{x}_i - \bar{x}_{i-1})(\bar{x}_i + \bar{x}_{i-1}) + (x_i - \bar{x}_{i-1})(x_i + \bar{x}_{i-1}) \\
&= v_{i-1} - (x_i - \bar{x}_{i-1})(\bar{x}_i + \bar{x}_{i-1}) + (x_i - \bar{x}_{i-1})(x_i + \bar{x}_{i-1}) \\
&= v_{i-1} + (x_i - \bar{x}_{i-1})(x_i - \bar{x}_i) \\
&= v_{i-1} + (x_i - \bar{x}_{i-1}) \left(x_i - \bar{x}_{i-1} - \frac{1}{i}(x_i - \bar{x}_{i-1}) \right) \\
&= v_{i-1} + (x_i - \bar{x}_{i-1})^2 - \frac{1}{i}(x_i - \bar{x}_{i-1})^2
\end{aligned}$$

which establishes the second equation in the theorem.

Algorithm 4.1.1 Given the sample x_1, x_2, x_3, \dots , Welford's algorithm for calculating the sample mean \bar{x} and standard deviation s is

```

n = 0;
x̄ = 0.0;
v = 0.0;
while ( more data ) {
    x = GetData();
    n++;
    d = x - x̄;                               /* temporary variable */
    v = v + d * d * (n - 1) / n;
    x̄ = x̄ + d / n;
}
s = sqrt(v / n);
return n, x̄, s;

```

Program `uvs`

Algorithm 4.1.1 is a one-pass $O(n)$ algorithm that does not require prior knowledge of the sample size n . The algorithm is not as prone to accumulated floating-point round-off error as is the conventional one-pass algorithm, yet it is essentially as efficient. Program `uvs` at the end of this section is based on a robust version of Algorithm 4.1.1 designed to not fail if applied, by mistake, to an empty sample. This program computes the sample mean and standard deviation as well as the sample minimum and maximum.

Input to program `uvs` can be integer-valued or real-valued. The input is assumed to be a *text* data file, in a one-value-per-line format with no blank lines in the file. (The file `uvs.dat` is an example.) To use the program, compile `uvs.c` to produce the executable file `uvs`. Then at a command line prompt, `uvs` can be used in three ways:

- To have `uvs` read a disk data file, say `uvs.dat`, at a command line prompt use ‘<’ redirection as:

```
uvs < uvs.dat
```

- To have `uvs` filter the numerical output of a program, say `test`, at a command line prompt use a ‘|’ pipe as:

```
test | uvs
```

- To use `uvs` with keyboard input, at a command line prompt enter:

```
uvs
```

then enter the data, one value per line, being sure to signify an end-of-file as the last line of input [`^d` (Ctrl-d) in Unix or `^z` (Ctrl-z) in Microsoft Windows].

4.1.3 EXAMPLES

The two examples given here illustrate the effect of independent and dependent sampling on the sample statistics \bar{x} and s .

Example 4.1.6 If a $Uniform(a, b)$ random variate sample x_1, x_2, \dots, x_n is generated then, in the limit as $n \rightarrow \infty$, the sample mean and sample standard deviation will converge to the limits indicated

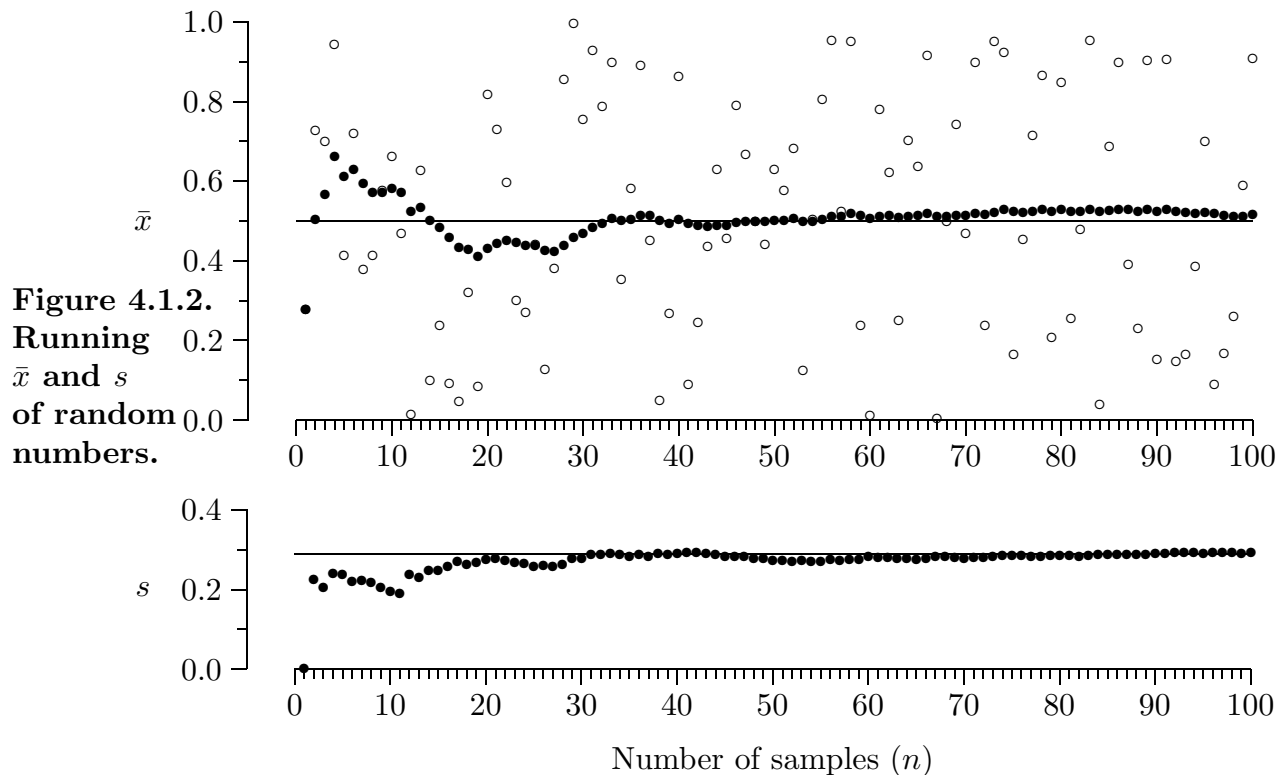
$$\bar{x} \rightarrow \frac{a+b}{2} \quad \text{and} \quad s \rightarrow \frac{b-a}{\sqrt{12}}.$$

(see Chapter 7 for more details concerning $Uniform(a, b)$ random variates.)

Welford’s algorithm was used in conjunction with a random variate sample generated by calls to `Random` (drawn from stream 0 with an `rngs` initial seed of 12345). Since the random variates so generated are independent $Uniform(0, 1)$, as the sample size increases the sample mean and standard deviation should converge to

$$\frac{0+1}{2} = 0.5 \quad \text{and} \quad \frac{1-0}{\sqrt{12}} \cong 0.2887.$$

In the upper display in Figure 4.1.2, the sample values are indicated with \circ ’s. The \bullet ’s show the running values of \bar{x} and indicate the convergence of the corresponding sample mean \bar{x}_n to 0.5 (the solid, horizontal line). As expected, the \circ and the \bullet overlap when $n = 1$. The plot of \bar{x} may remind students of grade point average (GPA) calculations which can be significantly influenced during the freshman and sophomore years, but remains fairly constant during the junior and senior years. Similarly, in the lower display in Figure 4.1.2, the \bullet ’s show the running values of s and indicate the convergence of the sample standard deviation $s_n = \sqrt{v_n/n}$ to 0.2887 (the solid, horizontal line). As discussed earlier in this section, the sample standard deviation is 0 when $n = 1$. Consistent with the discussion in Section 2.3, the convergence of both \bar{x} and s to their theoretical values is not necessarily monotone with increasing n . The final values plotted on each display corresponding to $n = 100$ are $\bar{x} = 5.15$ and $s = 0.292$.



Analogous to Example 4.1.6, if an *Equilikeley*(a, b) random variate sample is generated then, in the limit as $n \rightarrow \infty$, the mean and standard deviation of the sample will converge to the limits indicated

$$\bar{x} \rightarrow \frac{a+b}{2} \quad \text{and} \quad s \rightarrow \sqrt{\frac{(b-a+1)^2 - 1}{12}}.$$

Similarly, if an *Exponential*(μ) random variate sample is generated then

$$\bar{x} \rightarrow \mu \quad \text{and} \quad s \rightarrow \mu$$

as $n \rightarrow \infty$ and for a *Geometric*(p) random variate sample

$$\bar{x} \rightarrow \frac{p}{1-p} \quad \text{and} \quad s \rightarrow \frac{\sqrt{p}}{1-p}.$$

(Chapters 6 and 7 contain derivations of many of these results.)

Serial Correlation

The random variates generated in Example 4.1.6 were *independent* samples. Informally, independence means that each x_i in the sample x_1, x_2, \dots, x_n has a value that does not depend in any way on any other point in the sample. When an independent sample is displayed, as in Example 4.1.6, there will not be any deterministic “pattern” or “trend” to the data.

Because repeated calls to `Random` produce a simulated independent sample, there is no deterministic pattern or trend to the sample in Example 4.1.6. If, however, we were to display the time-sequenced output from a discrete-event simulation then that sample typically will not be independent. For example, if we display a sample of random waits experienced by consecutive jobs passing through a single-server service node then a pattern will be evident because the sample is *not* independent, particularly if the utilization is large. That is, if the i^{th} job experiences a large wait then the $(i + 1)^{\text{th}}$ job is likely to also experience a large wait with an analogous statement true if the i^{th} job experiences a small wait. In statistical terms, the wait times of consecutive jobs have positive *serial correlation* (see Section 4.4 for more details) and because of this the wait times in the sample will be dependent (i.e., not independent).

Hence independence is typically an appropriate assumption for Monte Carlo simulation, as illustrated in Example 4.1.6. It is typically *not* an appropriate assumption for discrete-event simulation, however, as illustrated in the following example.

Example 4.1.7 Program `ssq2`, with *Exponential*(2) interarrival times and *Uniform*(1, 2) service times, was modified to output the waits w_1, w_2, \dots, w_n experienced by the first $n = 100$ jobs. To simulate starting the service node in (approximate) steady-state the initial value of the program variable `departure` was set to 3.0. The `rng` initial seed was 12345. For this single-server service node it can be shown (see Section 8.5) that as $n \rightarrow \infty$ the sample mean and standard deviation of the wait times should converge to 3.83 and 2.83 respectively, as indicated by the solid horizontal lines in Figure 4.1.3.

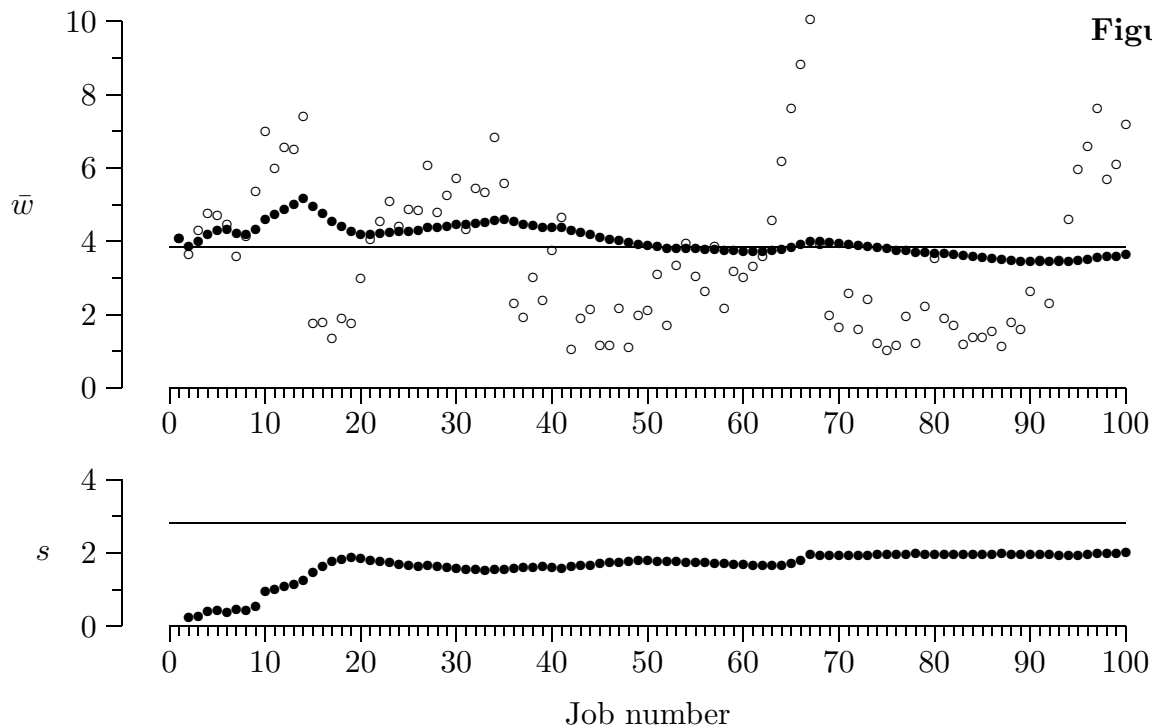


Figure 4.1.3.
Running
 \bar{x} and s
of wait
times.

As in Example 4.1.6, the sample values are indicated in the upper figure with \circ 's and the \bullet 's indicate the (unbiased) convergence of the corresponding sample mean to its expected steady-state value.* Unlike the upper figure in Example 4.1.6, this sample reveals a lack-of-independence pattern caused by high positive serial correlation in the sample wait times w_1, w_2, \dots, w_n . A sequence of longer-than-average waits is followed by a sequence of smaller-than-average waits. This pattern is quite different from that in Figure 4.1.2, where each observation is above or below the mean ($1/2$) as in a sequence of tosses of a fair coin. The running average wait does, however, converge to the theoretical value. Moreover, as illustrated in the lower figure, the high positive serial correlation produces a pronounced *bias* in the sample standard deviation. That is, the sample standard deviation consistently underestimates the expected value. In this case a profound modification to the sample variance equation (and thus the sample standard deviation equation) is required to remove the bias; just replacing $1/n$ with $1/(n-1)$ will not do it. (See Appendix F.)

4.1.4 TIME-AVERAGED SAMPLE STATISTICS

As discussed in previous chapters, time-averaged statistics play an important role in discrete-event simulation. The following definition is the time-averaged analog of Definition 4.1.1.

Definition 4.1.3 Given a function $x(t)$ defined for all $0 < t < \tau$ as a realization (sample path) of a stochastic process the *sample-path mean* is

$$\bar{x} = \frac{1}{\tau} \int_0^{\tau} x(t) dt.$$

The associated *sample-path variance* is

$$s^2 = \frac{1}{\tau} \int_0^{\tau} (x(t) - \bar{x})^2 dt$$

and the *sample-path standard deviation* is $s = \sqrt{s^2}$.

The equation for s^2 in Definition 4.1.3 is the two-pass variance equation; the corresponding one-pass equation is

$$s^2 = \left(\frac{1}{\tau} \int_0^{\tau} x^2(t) dt \right) - \bar{x}^2.$$

As the time-averaged statistic changes at the event times in a discrete-event simulation, only three memory locations are needed in order to store the running statistics $\int x(t) dt$, $\int x^2(t) dt$, and the length of the observation period to date.

* If the initial state of the service node had been idle, rather than adjusted to simulate starting the service node in (approximate) steady-state, then there would have been some *initial state bias* in the sample means.

For a discrete-event simulation a sample path is naturally *piecewise constant*. That is, as illustrated in Figure 4.1.4, there are *event times* $0 = t_0 < t_1 < t_2 < \dots < t_n = \tau$ and associated state values x_1, x_2, \dots, x_n on the time intervals $(t_0, t_1], (t_1, t_2], \dots, (t_{n-1}, t_n]$. The choice of making the left endpoint of these intervals open and the right endpoint of these intervals closed is arbitrary.

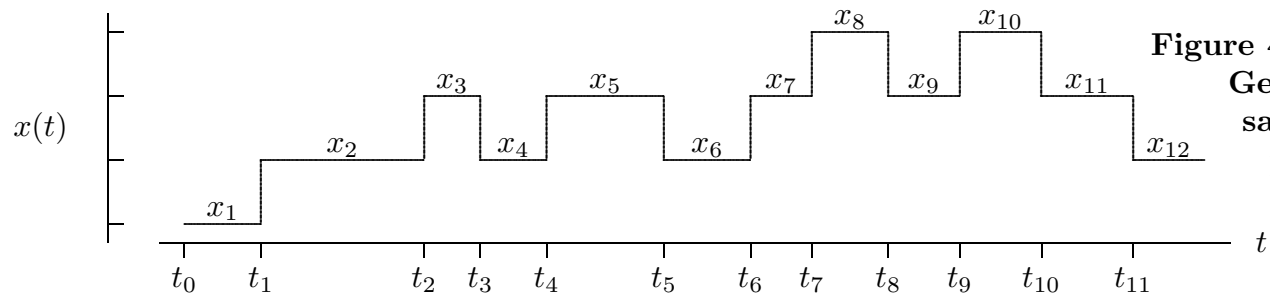


Figure 4.1.4.
Generic
sample
path.

If a sample path is piecewise constant then the sample-path *integral* equations for \bar{x} and s^2 in Definition 4.1.3 can be reduced to *summation* equations, as summarized by Theorem 4.1.3 that follows. The *defining formulas* given in Definition 4.1.3 are useful conceptually, but Theorem 4.1.3 gives the *computational formulas* that are used to compute \bar{x} and s^2 in practice.

Theorem 4.1.3 Given a sample path represented by the piecewise constant function

$$x(t) = \begin{cases} x_1 & t_0 < t \leq t_1 \\ x_2 & t_1 < t \leq t_2 \\ \vdots & \vdots \\ x_n & t_{n-1} < t \leq t_n. \end{cases}$$

and the inter-event times $\delta_i = t_i - t_{i-1}$ for $i = 1, 2, \dots, n$ then with $t_0 = 0$ and $\tau = t_n$ the sample-path mean is

$$\bar{x} = \frac{1}{\tau} \int_0^\tau x(t) dt = \frac{1}{t_n} \sum_{i=1}^n x_i \delta_i$$

and the sample-path variance is

$$s^2 = \frac{1}{\tau} \int_0^\tau (x(t) - \bar{x})^2 dt = \frac{1}{t_n} \sum_{i=1}^n (x_i - \bar{x})^2 \delta_i = \left(\frac{1}{t_n} \sum_{i=1}^n x_i^2 \delta_i \right) - \bar{x}^2.$$

Welford's Sample-Path Algorithm

The two-pass and one-pass algorithms are susceptible to floating-point round-off error as described earlier. Welford's algorithm can be extended to the time-averaged case by defining running sample means and running sample sum of squared deviations analogous to the earlier case.

Definition 4.1.4 With reference to Theorem 4.1.3, for $i = 1, 2, \dots, n$ the sample-path variant of Welford's algorithm is based on the definitions

$$\begin{aligned}\bar{x}_i &= \frac{1}{t_i}(x_1\delta_1 + x_2\delta_2 + \cdots + x_i\delta_i) \\ v_i &= (x_1 - \bar{x}_i)^2\delta_1 + (x_2 - \bar{x}_i)^2\delta_2 + \cdots + (x_i - \bar{x}_i)^2\delta_i\end{aligned}$$

where \bar{x}_i and v_i/t_i are the sample-path mean and variance, respectively, of the sample path $x(t)$ for $t_0 \leq t \leq t_i$.

Theorem 4.1.4 For $i = 1, 2, \dots, n$ the variables \bar{x}_i and v_i in Definition 4.1.4 can be computed recursively by

$$\begin{aligned}\bar{x}_i &= \bar{x}_{i-1} + \frac{\delta_i}{t_i}(x_i - \bar{x}_{i-1}) \\ v_i &= v_{i-1} + \frac{\delta_i t_{i-1}}{t_i}(x_i - \bar{x}_{i-1})^2\end{aligned}$$

with the initial conditions $\bar{x}_0 = 0$ and $v_0 = 0$.

4.1.5 EXERCISES

Exercise 4.1.1 Prove Theorem 4.1.1 by first proving that $d^2(x) = (x - \bar{x})^2 + s^2$.

Exercise 4.1.2 Prove Theorem 4.1.1 for the $1/(n-1)$ form of the sample variance.

Exercise 4.1.3 Relative to Theorem 4.1.2, prove that the sequence v_1, v_2, \dots, v_n satisfies the inequality $0 = v_1 \leq v_2 \leq \cdots \leq v_{n-1} \leq v_n = ns^2$.

Exercise 4.1.4^a What common sample statistic best fits the sample x_1, x_2, \dots, x_n in the sense of minimizing

$$d(x) = \frac{1}{n} \sum_{i=1}^n |x_i - x|?$$

Exercise 4.1.5 The statistic q^3 is the so-called sample *skewness* defined by

$$q^3 = \frac{1}{n} \sum_{i=1}^n \left(\frac{x_i - \bar{x}}{s} \right)^3.$$

(a) What is the one-pass version of the equation for q^3 ? (b) Extend the conventional (non-Welford) one-pass algorithm to also compute the sample statistic $q = (q^3)^{1/3}$. (c) What is the value of q for the data in the file `uvs.dat`?

Exercise 4.1.6 Look up the article by Chan, Golub, and LeVeque (1983) in the bibliography. Give a survey of existing algorithms for computing the sample variance listing the pros and cons of each method.

Exercise 4.1.7 (a) Generate an *Exponential*(9) random variate sample of size $n = 100$ and compute the proportion of points in the sample that fall within the intervals $\bar{x} \pm 2s$ and $\bar{x} \pm 3s$. Do this for 10 different `rngs` streams. (b) In each case, compare the results with Chebyshev's inequality. (c) Comment.

Exercise 4.1.8 Generate a plot similar to that in Figure 4.1.2 with calls to *Exponential*(17), rather than `Random` to generate the variates. Indicate the values to which the sample mean and sample standard deviation will converge.

Exercise 4.1.9^a Prove Theorems 4.1.3 and 4.1.4.

Exercise 4.1.10 (a) Given x_1, x_2, \dots, x_n with $a \leq x_i \leq b$ for $i = 1, 2, \dots, n$, what is the largest and smallest possible value of \bar{x} ? (b) Same question for s .

Exercise 4.1.11 Calculate \bar{x} and s by hand using the 2-pass algorithm, the 1-pass algorithm, and Welford's algorithm in the following two cases. (a) The data based on $n = 3$ observations: $x_1 = 1$, $x_2 = 6$, and $x_3 = 2$. (b) The sample path $x(t) = 3$ for $0 < t \leq 2$, and $x(t) = 8$ for $2 < t \leq 5$, over the time interval $0 < t \leq 5$.

Exercise 4.1.12^a The extent to which accumulated round-off error is a potential problem in any calculation involving floating-point arithmetic is determined, in part, by what is known as the computational system's "floating-point precision." One way to determine this precision is by executing the code fragment below.

```
typedef float FP;                                /* float or double */
const FP ONE = 1.0;
const FP TWO = 2.0;
    FP tiny = ONE;
    int count = 0;
while (ONE + tiny != ONE) {
    tiny /= TWO;
    count++;
}
```

(a) Experiment with this code fragment on at least three systems being sure to use both `float` and `double` data types for the generic floating-point type `FP`. (b) Explain what this code fragment does, why the `while` loop terminates, and the significance of the variables `tiny` and `count`. Be explicit.