Proceedings of the 2014 Winter Simulation Conference
A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds.

COMPUTATIONAL PROBABILITY APPLICATIONS

Lawrence M. Leemis

The College of William & Mary
Department of Mathematics
Williamsburg, VA 23187, U.S.A.

ABSTRACT

There is a boundary separating analytic methodology and simulation methodology. If a problem involves the flipping of coins or the rolling of dice, for example, analytic methods are generally employed. If a problem involves a complex series of queues with a nonstationary arrival stream, discrete-event simulation methods are generally employed. This tutorial considers problems that are near the boundary between analytic methods and simulation methods. We use the Maple-based APPL (A Probability Programming Language) to perform operations on random variables to address these problems. The problems considered are the infinite bootstrap, the probability distribution of the Kolmogorov–Smirnov test statistic, the distribution of the time to complete a stochastic activity network, finding a lower bound on system reliability, Benford's law, finding the probability distribution and variance–covariance matrix of sojourn times in a queueing model, probability distribution relationships, testing random numbers, bivariate transformations, and autoregressive moving average time series models.

1 INTRODUCTION

Certain problems in probability and stochastic processes, for example, probability models that can be boiled down to drawing balls from an urn, are best handled by analytic methods. Once a problem becomes slightly more complex, analytic methods are typically abandoned in favor of Monte Carlo or discrete-event simulation. This tutorial surveys the use of a computer algebra system to solve mathematically intractable stochastic modeling problems.

More specifically, this tutorial introduces APPL (A Probability Programming Language), which is a Maple-based probability language that has been used to solve difficult stochastic modeling problems. This language is inherently different than the dozens of statistical packages that are currently on the market. These packages manipulate data values whereas APPL manipulates random variables. The work presented in this tutorial is joint work with the author and his colleagues Kerry Connell, John Drew, Matt Duggan, Diane Evans, Andy Glen, Billy Kaczynski, Jeff Mallozzi, Raghu Pasupathy, Bruce Schmeiser, Jackie Taber, Erik Vargo, Keith Webb, and Jeff Yang. APPL is available from the author at no charge for non-commercial use.

This tutorial introduces APPL in the next section using two short examples. This is followed by sections that illustrate the use of APPL in solving problems in a diverse set of research areas. The application areas are bootstrapping, the Kolmogorov–Smirnov test statistic, stochastic activity networks, lower bounds on system reliability, Benford's law, queueing, probability distribution relationships, testing random numbers, bivariate transformations, and time series models.

2 APPL: A PROBABILITY PROGRAMMING LANGUAGE

APPL consists of a suite of Maple procedures that give exact solutions to probability problems. Sample procedures include:

- `Transform` to find the transformation of a random variable;
- `Convolution` to find the distribution of the sum of mutually independent random variables;
- `Mean` to find the expected value of a random variable;
- `OrderStat` to find the distribution of an order statistic;
- `Minimum` to find the distribution of a minimum of mutually independent random variables;
- `PDF, CDF, IDF` to find the PDF, CDF, IDF (inverse distribution function);
- `ExponentialRV`, and other popular parametric distributions.

The following two elementary examples illustrate the syntax associated with APPL.

Example 1 (sums of independent and identically distributed random variables). Let $X_1, X_2, \ldots, X_{10}$ be independent and identically distributed $U(0,1)$ random variables. Find

$$P\left(4 < \sum_{i=1}^{10} X_i < 6\right).$$

The historical approaches to solve problems like this are the central limit theorem or Monte Carlo simulation. The central limit theorem gives the approximate value of the desired probability because the population distribution is not normally distributed. The small sample size means that the error associated with the estimate might be substantial. Since the population mean and variance of the $U(0,1)$ distribution are $1/2$ and $1/12$, respectively, the estimated probability is

$$P\left(4 < \sum_{i=1}^{10} X_i < 6\right) = P\left(\frac{4-5}{\sqrt{10/12}} < \frac{\sum_{i=1}^{10} X_i - 5}{\sqrt{10/12}} < \frac{6-5}{\sqrt{10/12}}\right) = P(-1.0954 < Z < 1.0954) = 0.7267.$$

This particular application of the central limit theorem results in only one digit of accuracy. The tenths digit, 7, is correct, but the hundredths digit, 3, is incorrect.

The second historical approach to a problem of this type is Monte Carlo simulation. This requires custom programming and the estimate for the probability is given as a point and interval estimator. In theory, as many digits as desired can be obtained, but in practice, each additional digit of accuracy requires $100\times$ replications. The R code given below replicates the experiment of interest 1,000,000 times,

```
nrep = 1000000
count = 0
for (i in 1:nrep) {
  x = sum(runif(10))
  if (x > 4 & x < 6) count = count + 1
}
print(count / nrep)
```

After a call to `set.seed(3)` to initialize the random number stream, five runs of the Monte Carlo simulation yield the following results:

0.722060 0.722310 0.722529 0.721761 0.722818.

This is a huge improvement over the central limit theorem. We now have certainty in the first two digits of the solution, and near certainty in the third digit of the solution.

But neither the central limit theorem nor Monte Carlo simulation are ideal. They only give *estimates* of the probability of interest, and their error cannot be bounded with certainty.

APPL provides a mechanism for calculating the exact probability of interest. The APPL statements below solve the problem, returning the solution as an exact fraction. The first statement sets `n` to 10. The second statement sets `X` to a $U(0,1)$ random variable by using the `UniformRV` procedure. The third

statement calls the `ConvolutionIID` procedure to find the probability distribution of the sum of ten mutually independent and identically distributed $U(0,1)$ random variables. Finally, the fourth statement calculates the cumulative distribution function of the sum evaluated at 6 minus the cumulative distribution function of the sum evaluated at 4, which gives the result.

```
n := 10;
X := UniformRV(0, 1);
Y := ConvolutionIID(X, n);
CDF(Y, 6) - CDF(Y, 4);
```

These statements return the exact probability as

$$P\left(4 < \sum_{i=1}^{10} X_i < 6\right) = \frac{655177}{907200},$$

or, to four digits of precision, 0.7222.

Example 2 (distribution of the product of two independent random variables). Let $X \sim \text{triangular}(1,2,4)$ and $Y \sim \text{triangular}(1,2,3)$ be independent random variables, where the three parameters denote the minimum, mode, and maximum of the triangular distribution. Find the probability distribution of $V = XY$.

The solution choices in this case are limited. It can be worked out by hand or can be computed with APPL. The reader is spared the details of computing the probability distribution of V by hand. The APPL code for finding the probability of V is given below. The first two statements define the random variables `X` and `Y` with the `TriangularRV` procedure, and the third statement calls the `Product` procedure to compute the probability distribution of `V`.

```
X := TriangularRV(1, 2, 4);
Y := TriangularRV(1, 2, 3);
V := Product(X, Y);
```

The probability density function of V that is returned by APPL is

$$f_V(v) = \begin{cases} -\frac{4}{3}v + \frac{2}{3}\ln v + \frac{2v}{3}\ln v + \frac{4}{3} & 1 < v \le 2 \\ -8 + \frac{14}{3}\ln 2 + \frac{7v}{3}\ln 2 + \frac{10}{3}v - 4\ln v - \frac{5v}{3}\ln v & 2 < v \le 3 \\ -4 + \frac{14}{3}\ln 2 + \frac{7v}{3}\ln 2 + 2v - 2\ln v - v\ln v - 2\ln 3 - \frac{2v}{3}\ln 3 & 3 < v \le 4 \\ \frac{44}{3} - 14\ln 2 - \frac{7v}{3}\ln 2 - \frac{8}{3}v - 2\ln 3 + \frac{22}{3}\ln v - \frac{2v}{3}\ln 3 + \frac{4v}{3}\ln v & 4 < v \le 6 \\ \frac{8}{3} - 8\ln 2 - \frac{4v}{3}\ln 2 - \frac{2}{3}v + \frac{4}{3}\ln v + \frac{v}{3}\ln v + 4\ln 3 + \frac{v}{3}\ln 3 & 6 < v \le 8 \\ -8 + 8\ln 2 + \frac{2v}{3}\ln 2 + \frac{2}{3}v + 4\ln 3 - 4\ln v + \frac{v}{3}\ln 3 - \frac{v}{3}\ln v & 8 < v < 12. \end{cases}$$

Further details concerning APPL and further examples can be found in Glen, Evans, and Leemis (2001) and Drew, et al. (2008).

3 BOOTSTRAPPING

Bootstrapping is a well-established statistical technique that involves sampling data values with replacement, which is often known as *resampling*. Efron and Tibshirani (1993) give the data set shown in Table 1. The question that they pose is whether there is a statistically significant difference between the medians of the rat survival times (in days) of the two populations. The strategy is to estimate the standard error of the difference between the population medians.

The bootstrapping approach begins by generating B bootstrap samples, each of which consists of $n = 7$ samples drawn with replacement from the survival times in the treatment group: 16, 23, 38, 94, 99, 141,

Table 1: Rat survival times (days).

Group	Data	n	Median
Treatment	16, 23, 38, 94, 99, 141, 197	7	94
Control	10, 27, 30, 40, 46, 51, 52, 104, 146	9	46

and 197. The sample median of each bootstrap sample is calculated and stored. Next, calculate sample standard deviation of the B sample medians, which is an estimate of the standard error of the sample median for $B = 50$ bootstrap samples drawn from the treatment group. The S-Plus code below shows how this can be accomplished.

```
set.seed(1)
tr = c(16, 23, 38, 94, 99, 141, 197)
medn = function(x) {quantile(x, 0.50)}
bootstrap(tr, medn, B = 50)
```

This code returns the estimated standard error of the median of the treatment group as 41.18. Table 2 shows the bootstrap estimates of the standard error of the median calculated in this fashion for several values of B for both the treatment and control groups.

Table 2: Bootstrap estimates of the standard error of the median.

	$B = 50$	$B = 100$	$B = 250$	$B = 1000$	$B = +\infty$
Treatment	41.18	37.63	36.88	38.98	37.83
Control	20.30	12.68	9.538	13.82	13.08

The estimates of the standard error for the control group range from 9.538 to 20.30, for example, depending on the value of B. But since B is an arbitrary parameter specified by the modeler, one would like to use only the $B = +\infty$ column of Table 2. The entry in the $B = +\infty$ column from Table 2 for the treatment group are calculated via the following APPL statements.

```
treatment := [16, 23, 38, 94, 99, 141, 197];
X := BootstrapRV(treatment);
Y := OrderStat(X, 7, 4);
sqrt(Variance(Y));
```

The list named `treatment` is set to the seven survival times in the first statement. The APPL procedure `BootstrapRV` sets the random variable `X` to a discrete random variable with the seven support values in the list `treatment` with equal probabilities. The APPL procedure `OrderStat` determines the probability mass function of the sample median (the fourth ordered value) of a random sample of seven values drawn with replacement from the distribution of `X` The resulting probability mass function of Y, the sample bootstrap median for the treatment group, is

$$f(y) = \begin{cases} 8359/823543 & y = 16 \\ 80809/823543 & y = 23 \\ 196519/823543 & y = 38 \\ 252169/823543 & y = 94 \\ 196519/823543 & y = 99 \\ 80809/823543 & y = 141 \\ 8359/823543 & y = 197. \end{cases}$$

The associated standard error (the standard deviation of `Y`) for the treatment group in the "infinite bootstrap" is

$$\frac{2}{823543}\sqrt{242712738519382} \cong 37.8347,$$

which is the upper-right entry in Table 2. Likewise, the standard error for the control group is

$$\frac{1}{387420489}\sqrt{25662937134123797402} \cong 13.0759,$$

which is the lower-right entry in Table 2. The seemingly large difference between the two sample medians, $94-46=48$ days, is only

$$\frac{48}{\sqrt{37.83^2+13.08^2}} \cong 1.19$$

standard-deviation units away from zero, so we conclude that there is not a statistically significant difference between the median survival times of the two groups. The point here is that APPL frees us from drawing a finite number of bootstrap samples, and bootstrapping can be applied in this setting without having B as a part of the bootstrap process.

4 KOLMOGOROV–SMIRNOV TEST STATISTIC

The one-sample Kolmogorov–Smirnov goodness-of-fit test is used to compare an empirical distribution to a hypothesized or fitted probability distribution. The defining formula for the test statistic is

$$D_n = \sup_x |F(x) - F_n(x)|,$$

where $F_n(x)$ is the empirical cumulative distribution function associated with the n data values $x_1, x_2, \ldots, x_n$ and $F(x)$ is the hypothesized or fitted cumulative distribution function. Since $F(x)$ is a monotone increasing function of x, the supremum must occur at a data value, so the computational formula is

$$D_n = \max_{i=1,2,\ldots,n} \left\{ F\left(x_{(i)}\right) - \frac{i-1}{n}, \frac{i}{n} - F\left(x_{(i)}\right) \right\},$$

where $x_{(1)}, x_{(2)}, \ldots, x_{(n)}$ are the associated order statistics. In the all-parameters-known case associated with a known or hypothesized parent population distribution, Birnbaum (1952) gave an expression for the cumulative distribution function of D_n as

$$P\left(D_n < \frac{1}{2n} + v\right) = n! \int_{\frac{1}{2n}-v}^{\frac{1}{2n}+v} \int_{\frac{3}{2n}-v}^{\frac{3}{2n}+v} \cdots \int_{\frac{2n-1}{2n}-v}^{\frac{2n-1}{2n}+v} g(u_1, u_2, \ldots, u_n)\, du_n \ldots du_2\, du_1$$

for $0 \le v \le \frac{2n-1}{2n}$, where

$$g(u_1, u_2, \ldots, u_n) = 1$$

for $0 \le u_1 \le u_2 \le \cdots \le u_n$. For example, when $n=1$, $D_1 \sim U(1/2, 1)$. For $n=2$, the cumulative distribution function of D_2 is

$$F_{D_2}(y) = P(D_2 \le y) = \begin{cases} 0 & y \le \frac{1}{4} \\ 8\left(y - \frac{1}{4}\right)^2 & \frac{1}{4} < y < \frac{1}{2} \\ 1 - 2(1-y)^2 & \frac{1}{2} < y < 1 \\ 1 & y \ge 1. \end{cases}$$

These simple cases are easily worked out by hand, but the calculations become much more tedious as n increases. Using an algorithm developed in Drew, Glen, and Leemis (2000), a procedure named `KSRV` with a single argument `n` was implemented in APPL. A call to `KSRV(6)`, for example, returns the cumulative

distribution function

$$F_{D_6}(y) = \begin{cases} 0 & y < \frac{1}{12} \\ 46080y^6 - 23040y^5 + 4800y^4 - \frac{1600}{3}y^3 + \frac{100}{3}y^2 - \frac{10}{9}y + \frac{5}{324} & \frac{1}{12} \le y < \frac{1}{6} \\ 2880y^6 - 4800y^5 + 2360y^4 - \frac{1280}{3}y^3 + \frac{235}{9}y^2 + \frac{10}{27}y - \frac{5}{81} & \frac{1}{6} \le y < \frac{1}{4} \\ 320y^6 + 320y^5 - \frac{2600}{3}y^4 + \frac{4240}{9}y^3 - \frac{785}{9}y^2 + \frac{145}{27}y - \frac{35}{1296} & \frac{1}{4} \le y < \frac{1}{3} \\ -280y^6 + 560y^5 - \frac{1115}{3}y^4 + \frac{515}{9}y^3 + \frac{1525}{54}y^2 - \frac{565}{81}y + \frac{5}{16} & \frac{1}{3} \le y < \frac{5}{12} \\ 104y^6 - 240y^5 + 295y^4 - \frac{1985}{9}y^3 + \frac{775}{9}y^2 - \frac{7645}{648}y + \frac{5}{16} & \frac{5}{12} \le y < \frac{1}{2} \\ -20y^6 + 32y^5 - \frac{185}{9}y^3 + \frac{175}{36}y^2 + \frac{3371}{648}y - 1 & \frac{1}{2} \le y < \frac{2}{3} \\ 10y^6 - 38y^5 + \frac{160}{3}y^4 - \frac{265}{9}y^3 - \frac{115}{108}y^2 + \frac{4651}{648}y - 1 & \frac{2}{3} \le y < \frac{5}{6} \\ -2y^6 + 12y^5 - 30y^4 + 40y^3 - 30y^2 + 12y - 1 & \frac{5}{6} \le y < 1 \\ 1 & y \ge 1. \end{cases}$$

Subject to CPU and memory limitations, `KSRV` works for all values of `n`. Next, consider the more practical "parameters estimated from data" case for finding the probability distribution of the Kolmogorov–Smirnov test statistic. Following a derivation from Evans, Drew, and Leemis (2008) for the exponential distribution with $n = 2$ data values fitted by maximum likelihood, the APPL code required to compute the probability distribution of D_2 is

```
Y   := UniformRV(0, 1 / 2);
A   := 1 - exp(-2 * y);
B   := exp(-2 * y) - 1 / 2;
C   := 1 / 2 - exp(-2 * (1 - y));
ys  := solve(B = C, y)[1];
yss := solve(A = C, y)[1];
g   := [[unapply(B,y), unapply(C,y), unapply(A,y)], [0,ys,yss,1 / 2]];
D2  := Transform(Y, g);
```

which results in the probability density function of D_2:

$$f_{D_2}(y) = \begin{cases} \frac{1}{1-y} + \frac{1}{1/2+y} + \frac{2y}{(1/2+y)(1/2-y)} & \frac{3}{4} - \frac{1}{4}\sqrt{1+\frac{16}{e^2}} < y < \frac{1}{2}\sqrt{1-\frac{4}{e^2}} \\ \frac{1}{1-y} + \frac{1}{1/2+y} & \frac{1}{2}\sqrt{1-\frac{4}{e^2}} < y < \frac{1}{2} \\ \frac{1}{1-y} & \frac{1}{2} < y < 1 - \frac{1}{e}. \end{cases}$$

The pattern that is evident in $f_{D_2}(y)$ prompted us to derive the probability density function of D_3, but, unfortunately, no such pattern persisted. The exact probability distribution of the one-sample Kolmogorov–Smirnov test statistic in the all-parameters-known case under the null hypothesis remains an open question.

5 STOCHASTIC ACTIVITY NETWORKS

A stochastic activity networks arises in a *project management* setting in which activity durations are modeled by positive random variables with known probability distributions. Oftentimes the goal is to find the probability distribution of the random time to complete the network. Two popular techniques for evaluating a stochastic activity network are PERT and Monte Carlo simulation. APPL can be used to calculate the exact distribution of the time to complete a stochastic activity network.

Series–parallel networks constitute a class of stochastic activity networks that are easy to analyze because a sequence of maximum and convolution operations (which can be executed with the APPL `Maximum` and

`Convolution` procedures) can be used to decompose the network into a single arc. Leemis et al. (2006) consider the more difficult case of a non-series–parallel network. The probability distribution of the time to complete these networks requires an algorithm that applies conditional probability to arcs that are on multiple paths through the network. One of the simplest such networks is the bridge network shown in Figure 1. When the arc durations Y_{ij} are mutually independent $U(0,1)$ random variables, for example, the

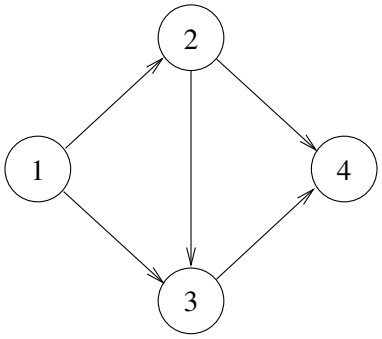

Figure 1: Bridge network.

cumulative distribution function of the time to complete the stochastic activity network, T_4, is

$$F_{T_4}(t) = \begin{cases} 0 & t \le 0 \\ \frac{11}{120}t^5 & 0 < t \le 1 \\ -\frac{1}{120}t^5 - \frac{1}{6}t^4 + \frac{2}{3}t^3 - \frac{1}{3}t^2 - \frac{1}{6}t + \frac{1}{10} & 1 < t \le 2 \\ \frac{1}{6}t^3 - \frac{3}{2}t^2 + \frac{9}{2}t - \frac{23}{6} & 2 < t \le 3 \\ 1 & t > 3 \end{cases}$$

Consider the stochastic activity network shown in Figure 2. Assuming that each of the activity durations is an independent exponential(1) random variable, the cumulative distribution function of the completion time for the network is

$$\begin{aligned} F_{T_6}(t) &= 1 + \frac{107}{4}e^{-2t} - \frac{71}{4}e^{-4t} - 8e^{-2t}t^2 - \frac{45}{2}e^{-2t}t - \frac{1}{6}e^{-2t}t^3 - \frac{1}{6}e^{-t}t^3 - 2e^{-t}t^2 - 2e^{-4t}t^2 \\ &- \frac{71}{2}e^{-3t}t + \frac{1}{8}e^{-2t}t^4 - \frac{1}{8}e^{-3t}t^4 - 9e^{-3t}t^2 + \frac{2}{3}e^{-3t}t^3 - 12e^{-4t}t - \frac{85}{4}e^{-3t} + \frac{45}{4}e^{-t} \end{aligned}$$

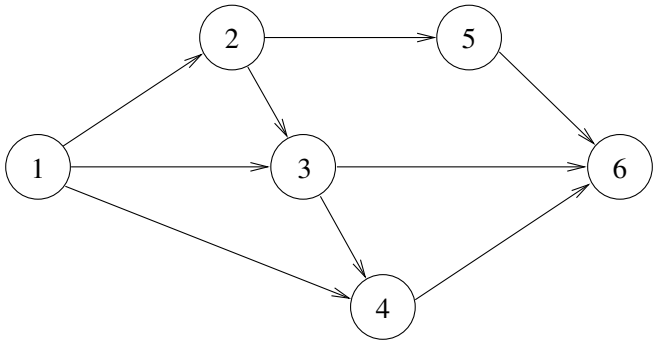

Figure 2: Stochastic activity network.

for $t > 0$. The mean network completion time is

$$E[T_6] = \int_0^\infty \left(1 - F_{T_6}(t)\right) dt = \frac{4213}{864} \cong 4.8762.$$

Although the two illustrations given here have identical distributions for activity durations for simplicity, this is not a requirement for the algorithm.

6 LOWER BOUND ON SYSTEM RELIABILITY

Failure data for a three-component series system is given in Table 3. Use bootstrapping to determine a 95% lower confidence bound on the system reliability for a series system of three independent components using the binary failure data (y_i, n_i), where

- y_i is the number of components of type i that pass the test,
- n_i is the number of components of type i on test,

for $i = 1, 2, 3$.

Table 3: Failure data for a three-component series system.

Component number	$i = 1$	$i = 2$	$i = 3$
Number passing (y_i)	21	27	82
Number on test (n_i)	23	28	84

Assuming that components fail independently, the point estimate for the system reliability is the product of the point estimates for the component reliabilities:

$$\frac{21}{23} \cdot \frac{27}{28} \cdot \frac{82}{84} = \frac{1107}{1288} \cong 0.8595.$$

The next step is to determine a lower confidence bound on the system reliability. We choose bootstrapping as an approach. For the first component, a bootstrap sample of size $n_1 = 23$ is taken with replacements from the 21 good components and 2 bad components, yielding a bootstrap fraction of bad components. Likewise for the second and third components. These three fractions are multiplied, giving a bootstrap system reliability, which will be centered around $1107/1288 \cong 0.8595$. The APPL code below calculates the exact distribution of this product in the random variable `T`.

```
X1 := BinomialRV(23, 21 / 23);
X1 := Transform(X1, [[x -> x / 23], [0, 23]]);
X2 := BinomialRV(28, 27 / 28);
X2 := Transform(X2, [[x -> x / 28], [0, 28]]);
X3 := BinomialRV(84, 82 / 84);
X3 := Transform(X3, [[x -> x / 84], [0, 84]]);
T  := Product(X1, X2, X3);
```

There are a possible $24 \cdot 29 \cdot 85 = 59,160$ potential mass values for `T`. Of these, only 6633 are distinct because the `Product` procedure combines repeated values. The lower 95% bootstrap confidence interval bound is the 0.05 fractile of the distribution of `T`, which is $6723/9016 \cong 0.7457$. This is consistent with bootstrapping experiments, but, once again, avoids the use of a finite number of bootstrap replications. Further details concerning the bootstrapping algorithm to determine a lower bound, plus a modification to the algorithm associated with components with perfect test results are given in Leemis (2006).

7 BENFORD'S LAW

Benford's law concerns the distribution of the leading digit in a data set. The history associated with Benford's law is given in Miller (2015). Simon Newcomb noticed that the early pages of logarithm tables

were more worn than the later pages in 1881. He conjectured that the probability mass function of X, the leading digit in a set of data, is

$$f_X(x) = P(X = x) = \log_{10}\left(1 + \frac{1}{x}\right) \qquad x = 1, 2, \ldots, 9.$$

So ones are the most likely and nines are the least likely leading digit. More specifically, $P(X = 1) = 0.301$ and $P(X = 9) = 0.0458$. Frank Benford apparently independently discovered the same probability distribution in 1938 and proceeded to fit the distribution to a wide variety of data sets. Benford's law has found a number of applications, including detecting election and accounting fraud. The goal in this section is to search for a probability distribution with positive support that satisfies Benford's law exactly.

Our approach for using APPL described here is given in more detail in Leemis, Schmeiser, and Evans (2000). Let T denote a random lifetime with survival function $S(t) = P(T \geq t)$, for $t > 0$. Let Y be the value of the leading digit in the lifetime T:

$$P(Y = y) = \sum_{i=-\infty}^{\infty} \left[S\left(y \cdot 10^i\right) - S\left((y+1)10^i\right)\right]$$

for $y = 1, 2, \ldots, 9$. We initially considered two measures of conformance to Benford's law:

$$c = \sum_{x=1}^{9} \frac{[P(Y = x) - P(X = x)]^2}{P(X = x)}$$

and

$$m = \max_{x=1,2,\ldots,9} \{|P(Y = x) - P(X = x)|\}.$$

The first measure is analogous to the chi-square goodness-of-fit test statistic and the second measure is analogous to the Kolmogorov–Smirnov goodness-of-fit test statistic. We calculated these measures of conformance for several well-known probability distributions, and the results are shown in Table 4.

Although Benford's law applies to a wide variety of data sets, there is, surprisingly, no perfect match to Benford's law among the lifetime distributions listed in Table 4. The distribution classes based on the hazard function given in the table are IFR (increasing failure rate), DFR (decreasing failure rate), BT (bathtub-shaped failure rate), and UBT (upside-down bathtub-shaped failure rate). Some distributions, such as the log logistic distribution, can come very close to a perfect match with Benford's law. Our experimentation revealed that the results for the exponential distribution with λ are identical for $10^i\lambda$, for $i = \pm 1, \pm 2, \ldots$. Both c and m increased for larger values of the shape parameter κ.

Table 4: Conformance to Benford's law for several survivor distributions.

Distribution	λ	κ	Class	c	m
Exponential	1		IFR/DFR	$0.61 \cdot 10^{-2}$	$0.29 \cdot 10^{-1}$
Exponential	5		IFR/DFR	$0.54 \cdot 10^{-2}$	$0.18 \cdot 10^{-1}$
Muth		0.1	IFR	$0.13 \cdot 10^{-1}$	$0.41 \cdot 10^{-1}$
Gompertz	5	1.1	IFR	$0.62 \cdot 10^{-2}$	$0.20 \cdot 10^{-1}$
Weibull	1	0.3	DFR	$0.37 \cdot 10^{-10}$	$0.16 \cdot 10^{-5}$
Weibull	1	2	IFR	0.19	0.11
Gamma	1	0.3	DFR	$0.15 \cdot 10^{-3}$	$0.29 \cdot 10^{-2}$
Gamma	1	2	IFR	$0.48 \cdot 10^{-1}$	$0.50 \cdot 10^{-1}$
Log logistic	1	0.3	DFR	$0.86 \cdot 10^{-21}$	$0.67 \cdot 10^{-11}$
Log logistic	1	2	UBT	$0.24 \cdot 10^{-1}$	$0.35 \cdot 10^{-1}$
Expon Power	1	0.3	BT	$0.48 \cdot 10^{-4}$	$0.17 \cdot 10^{-2}$

So our search for a probability distribution satisfying Benford's law exactly expanded beyond the popular probability models. Our next observation concerned the ability to generate a random variate from Benford's law. Since X has probability mass function

$$f_X(x) = P(X = x) = \log_{10}\left(1 + \frac{1}{x}\right) \qquad x = 1, 2, \ldots, 9,$$

it has cumulative distribution function

$$F_X(x) = P(X \le x) = \log_{10}(1 + x) \qquad x = 1, 2, \ldots, 9.$$

So for $U \sim U(0,1)$, a Benford random variate X is generated via

$$X \leftarrow \lfloor 10^U \rfloor.$$

(This formula can be generalized from base 10 to base b and from the leading digit to the leading r digits as $X \leftarrow \lfloor b^{U+r-1} \rfloor$.) This variate generation formula leads to a simple probability distribution that follows Benford's law exactly. If $U \sim U(0,1)$ and $T = 10^U$ then T has probability density function

$$f_T(t) = \frac{1}{t \ln 10} \qquad 1 < t < 10,$$

which is a distribution whose leading digit is a perfect match to Benford's law. The APPL `Transform` procedure calculated the distribution of T so as to find other distributions satisfying Benford's law exactly.

8 QUEUEING

APPL has also been applied to problems in queueing. Standard queueing formulas typically apply to steady-state results for simple models. Kaczynski, Leemis, and Drew (2012) wrote additional APPL procedures to find the probability distribution of the sojourn time (the time spent in the queue plus the time spent in service) for the initial customers in an M/M/1 queue. Three examples illustrate the use of the code.

Example 1 (sojourn time distribution). Consider an M/M/1 queue with arrival rate λ and service rate μ starting empty and idle at time $t = 0$. Calculate the probability distribution of the fourth customer's sojourn time, T_4, mean sojourn time, $E[T_4]$, and sojourn time variance, $V[T_4]$.

The following APPL statements calculate these quantities. The first statement sets `X` to the interarrival time distribution. The second statement sets `Y` to the service time distribution. The third statement sets `T` to the distribution of the fourth customer's sojourn time for a single-server queue that starts with no customers in the queue at time $t = 0$. The last two statements calculate the mean and variance of T_4.

```
X := ExponentialRV(lambda);
Y := ExponentialRV(mu);
T := Queue(X, Y, 4, 0, 1);
Mean(T);
Variance(T);
```

These statements return the probability density function of the fourth customer's sojourn time as

$$f_4(t) = \frac{1}{6(\lambda+\mu)^5}\mu^4 e^{-\mu t}\left(30\lambda^2 + 30\lambda^3 t + 24\lambda\mu + 24\lambda^2\mu t + 6\mu^2 + \right.$$
$$\left. 6\mu^2\lambda t + 9t^2\lambda^4 + 12t^2\lambda^3\mu + 3t^2\lambda^2\mu^2 + t^3\lambda^5 + 2t^3\lambda^4\mu + t^3\lambda^3\mu^2\right)$$

for $t > 0$ which has mean

$$E[T_4] = \frac{\mu^5 + 6\lambda\mu^4 + 26\mu^2\lambda^3 + 16\mu^3\lambda^2 + 17\mu\lambda^4 + 4\lambda^5}{\mu(\lambda+\mu)^5}$$

and variance

$$V[T_4] = \left(181\mu^2\lambda^8 + 484\mu^3\lambda^7 + 816\mu^4\lambda^6 + 868\mu^5\lambda^5 + 574\mu^6\lambda^4 + \right.$$
$$\left. 244\mu^7\lambda^3 + 40\mu\lambda^9 + 68\mu^8\lambda^2 + 12\mu^9\lambda + \mu^{10} + 4\lambda^{10}\right) \Big/ \left(\mu^2(\lambda+\mu)^{10}\right)$$

Example 2 (variance–covariance matrices) Find the variance–covariance matrix of the first three customer sojourn times in an initially empty and idle M/M/1 queue with arrival rate λ and service rate μ.

The number of potential sequences of arrival and departures to the queue for the first n customers is the Catalan number $(2n)!/(n!(n+1)!)$. The population covariance between first and second customer's sojourn times when $n = 3$ can be determined by the following APPL statement.

```
Cov(1, 2, 3);
```

When this statement is called for all elements, the resulting variance–covariance matrix is

$$\begin{bmatrix} \frac{1}{\mu^2} & \frac{\lambda(2\mu+\lambda)}{(\lambda+\mu)^2\mu^2} & \frac{\lambda^2(\lambda^2+4\lambda\mu+5\mu^2)}{(\lambda+\mu)^4\mu^2} \\ \bullet & \frac{2\lambda^2+4\lambda\mu+\mu^2}{(\lambda+\mu)^2\mu^2} & \frac{\lambda(2\lambda^2+8\lambda^2\mu+11\lambda\mu^2+2\mu^3)}{(\lambda+\mu)^4\mu^2} \\ \bullet & \bullet & \frac{3\lambda^6+18\lambda^5\mu+45\lambda^4\mu^2+54\lambda^3\mu^3+30\lambda^2\mu^4+8\lambda\mu^5+\mu^6}{(\lambda+\mu)^6\mu^2} \end{bmatrix}.$$

Example 3 Find the variance–covariance matrix of the first *nine* customer sojourn times in an initially empty and idle M/M/1 queue with arrival rate $\lambda = 1$ and service rate $\mu = 2$.

The queueing extension to APPL can perform the calculations symbolically or numerically. Again calling the `Cov` procedure to calculate the covariances, the variance–covariance matrix is

$$\begin{bmatrix} \frac{1}{4} & \frac{5}{36} & \frac{29}{324} & \frac{181}{2916} & \frac{1181}{26244} & \frac{2647}{78732} & \frac{18191}{708588} & \frac{127111}{6377292} & \frac{2699837}{172186884} \\ \bullet & \frac{7}{18} & \frac{13}{54} & \frac{239}{1458} & \frac{1543}{13122} & \frac{10303}{118098} & \frac{23485}{354294} & \frac{163493}{3188646} & \frac{3462503}{86093442} \\ \bullet & \bullet & \frac{1451}{2916} & \frac{8531}{26244} & \frac{53995}{236196} & \frac{356291}{2125764} & \frac{805705}{6377292} & \frac{5576849}{57395628} & \frac{39197977}{516560652} \\ \bullet & \bullet & \bullet & \frac{34514}{59049} & \frac{209794}{531441} & \frac{1357010}{4782969} & \frac{3031606}{14348907} & \frac{20810726}{129140163} & \frac{145390102}{1162261467} \\ \bullet & \bullet & \bullet & \bullet & \frac{12525605}{19131876} & \frac{77889229}{172186884} & \frac{170586983}{516560652} & \frac{1156711327}{4649045868} & \frac{8013045911}{41841412812} \\ \bullet & \bullet & \bullet & \bullet & \bullet & \frac{551583889}{774840978} & \frac{1162296371}{2324522934} & \frac{7727099083}{20920706406} & \frac{52871149859}{188286357654} \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \frac{10582107143}{13947137604} & \frac{67728246079}{125524238436} & \frac{454382575415}{1129718145924} \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \frac{225196533287}{282429536481} & \frac{1455144635743}{2541865828329} \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \frac{75890492486993}{91507169819844} \end{bmatrix}.$$

This matrix could be used to check the correctness of the associated discrete-event simulation code.

9 PROBABILITY DISTRIBUTION RELATIONSHIPS

Many of the probability distributions that are in common use (e.g., binomial, normal, exponential) are related to one another. For example, (a) a chi-square random variable with an even number of degrees of

freedom is an Erlang random variable, (b) the difference of two independent exponential random variables is a Laplace random variable, (c) a beta random variable with equal parameters approaches a normal random variable as the parameters go to ∞, (d) a binomial(n, p) random variable with fixed n and $p \sim$ beta is a beta–binomial random variable.

A chart has been placed on-line at `www.math.wm.edu/~leemis/chart/UDR/UDR.html` which displays and proves many of these relationships. The static version of this diagram is from Leemis and McQueston (2008). Based on the distributions in this diagram, Vargo, Pasupathy, and Leemis (2010) created moment ratio diagrams that give plots of the coefficient of variation vs. skewness and the skewness vs. kurtosis for many of the distributions in the diagram. APPL was required in order to check the relationships in the diagram as well as computing the moments for the moment ratio diagrams.

In order to assess a potential parametric distribution to use for modeling purposes based on a data set of n observations $x_1, x_2, \ldots, x_n$, we could conduct the following steps: (a) calculate the sample coefficient of variation and skewness

$$\hat{\gamma}_2 = \frac{s}{\bar{x}} \qquad \text{and} \qquad \hat{\gamma}_3 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s}\right)^3,$$

(b) plot the point $(\hat{\gamma}_2, \hat{\gamma}_3)$ on the moment ratio diagram of the coefficient of variation vs. skewness, (c) draw B bootstrap samples of size n from the data set, calculating $(\hat{\gamma}_2, \hat{\gamma}_3)$, (d) fit the bivariate normal distribution to the $(\hat{\gamma}_2, \hat{\gamma}_3)$ pairs, (e) plot the concentration ellipse associated with the fitted bivariate normal distribution. The probability distributions that are contained in the concentration ellipse are potential models for the data set.

10 TESTING RANDOM NUMBERS

Consider the Lehmer linear congruential generator

$$Z_i = aZ_{i-1} \bmod m,$$

where the modulus m and the multiplier a are positive integers satisfying $a < m$. This relationship is used to generate the pseudo-random numbers $U_i = Z_i/m$. George Marsaglia observed that random numbers that are produced by this generator fall in planes in two or more dimensions, and sometimes the number of planes is relatively small. While this negative aspect of random numbers generated by a Lehmer generator lessens their value for Monte Carlo simulation, it does not assess the *order* that the numbers are generated. If X_1, X_2, Y_1, Y_2 mutually independent $U(0,1)$ random variables, then the probability distribution of the distance between adjacent pairs (X_1, Y_1) and (X_2, Y_2) in the unit square, $D = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$, is calculated with the following APPL statements.

```
U1 := UniformRV(0,1);
U2 := UniformRV(0,1);
V1 := Difference(U1, U2);
g1 := [[x -> x * x, x -> x * x], [-infinity, 0, infinity]];
V2 := Transform(V1, g1);
V3 := Convolution(V2, V2);
g2 := [[x -> sqrt(x)], [0, 2]];
V4 := Transform(V3, g2);
```

The resulting probability density function of D is

$$f(x) = \begin{cases} 2x(x^2 - 4x + \pi) & 0 < x \le 1 \\ \dfrac{-2x\left(2\sqrt{x^2-1} + 4 - 4x^2 + 2\sqrt{x^2-1}\arcsin\left(\frac{x^2-2}{x^2}\right) + x^2\sqrt{x^2-1}\right)}{\sqrt{x^2-1}} & 1 < x < \sqrt{2}. \end{cases}$$

Duggan, Drew, and Leemis (2005) use this probability density function to test whether the distance between adjacent pairs $D = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$ for two random points (X_1, Y_1) and (X_2, Y_2) in the unit square follows this probability distribution.

11 BIVARIATE TRANSFORMATIONS

The `Transform` procedure in APPL finds the probability distribution of the random variable $Y = g(X)$. An extension to the APPL language developed by Yang, Drew, and Leemis (2012) can be used to find the probability distribution function of a transformation of two random variables. The `BiTransform` procedure gives the joint distribution of X and Y as its first argument, the function of interest $g(X,Y)$ as its second argument, and an option third argument $h(X,Y)$ as a dummy transformation. If the third argument is defaulted, the procedure will choose a dummy transformation. So the call to the procedure has the form

```
BiTransform(XY, g, h)
```

and can be used for continuous random variables X and Y. Two examples illustrate its use.

Example 1 The joint probability density function of X and Y is

$$f(x,y) = 2 \qquad x > 0, y > 0, x + y < 1.$$

Find the probability distribution of $U = g(X,Y) = X + Y$.

The APPL code to solve this problem is

```
XY := [[(x, y) -> 2], [[x > 0, y > 0, x + y < 1]],
       ["Continuous", "PDF"]];
g := [(x, y) -> x + y];
h := [(x, y) -> x - y];
U := BiTransform(XY, g, h);
```

The constraints associated with the definition of the support of the joint probability density function of X and Y must be entered as adjacent constraints; either a clockwise or counterclockwise fashion is acceptable. These APPL statements correctly return

$$f_U(u) = 2u \qquad 0 < u < 1$$

as the probability density function of `U`.

Example 2 The random variables X and Y are uniformly distributed on $(1,2) \times (-\pi, \pi)$. Find the joint probability distribution of $U = X \cos Y$ and $V = X \sin Y$ and the marginal distribution of U.

The APPL code to solve this problem is

```
X := UniformRV(1, 2);
Y := UniformRV(-Pi, Pi);
XY := JointPDF(X, Y);
g := [(x, y) -> x * cos(y)];
h := [(x, y) -> x * sin(y)];
BiTransform(XY, g, h);
```

These APPL statements correctly return

$$f_{U,V}(u,v) = \frac{1}{2\pi\sqrt{u^2 + v^2}} \qquad 1 < u^2 + v^2 < 4$$

as the joint probability density function of U and V. Note that the support of U and V is a donut-shaped region. The marginal probability density function of U is

$$f_U(u) = \begin{cases} \pi^{-1} \sinh^{-1} \sqrt{4u^{-2} - 1} & 1 < |u| < 2 \\ \pi^{-1} \left(\sinh^{-1} \sqrt{4u^{-2} - 1} - \sinh^{-2} \sqrt{u^{-1} - 1} \right) & |u| < 1. \end{cases}$$

12 ARMA MODELS

Box–Jenkins autoregressive (AR) moving average (MA) models are widely used to model time series in economics, engineering, meteorology, quality control, medicine, etc. An extension to APPL described in Webb and Leemis (2014) allows for the symbolic evaluation of ARMA models. The notation that is used to describe an ARMA model is

- Y_t is the value of the time series at time t,
- c is a real-valued constant,
- $\phi_1, \phi_2, \ldots, \phi_p$ are real-valued AR(p) parameters,
- $\theta_1, \theta_2, \ldots, \theta_q$ are real-valued MA(q) parameters,
- $\varepsilon_0, \varepsilon_1, \varepsilon_2, \ldots, \varepsilon_t$ are mutually independent error terms with mean 0.

The AR(p) model is

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t.$$

The MA(q) model is

$$Y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}.$$

The ARMA(p,q) model is

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}.$$

For more details, see Box and Jenkins (1994). Within the suite of programs in the time series extension for APPL is one known as "Exploratory Time Series Analysis," which is coded into an APPL procedure named `ETSA`. The APPL code to perform an exploratory time series analysis of an MA(3) model is

```
X := MA([1 / 5, -7 / 5, 11 / 5]):
ETSA(X);
```

This particular time series model is

$$Y_t = c + \varepsilon_t + \frac{1}{5}\varepsilon_{t-1} - \frac{7}{5}\varepsilon_{t-2} + \frac{11}{5}\varepsilon_{t-3}.$$

The `ETSA` procedure prints graphs of the autocorrelation function and the partial autocorrelation function. The procedure then performs a unit roots analysis, which indicates that the three unit roots are

$$-\frac{1}{33}\left(8171+33\sqrt{61305}\right)^{1/3} - \frac{16}{33\left(8171+33\sqrt{61305}\right)^{1/3}} + \frac{7}{33},$$

$$\frac{\left(8171+33\sqrt{61305}\right)^{1/3}}{66} + \frac{8}{33\left(8171+33\sqrt{61305}\right)^{1/3}} + \frac{7}{33} - \frac{i\sqrt{3}}{2}\left(-\frac{\left(8171+33\sqrt{61305}\right)^{1/3}}{33} + \frac{16}{33\left(8171+33\sqrt{61305}\right)^{1/3}}\right),$$

$$\frac{\left(8171+33\sqrt{61305}\right)^{1/3}}{66} + \frac{8}{33\left(8171+33\sqrt{61305}\right)^{1/3}} + \frac{7}{33} + \frac{i\sqrt{3}}{2}\left(-\frac{\left(8171+33\sqrt{61305}\right)^{1/3}}{33} + \frac{16}{33\left(8171+33\sqrt{61305}\right)^{1/3}},\right).$$

where $i = \sqrt{-1}$. All of these unit roots lie inside the unit circle. Assuming error terms are independent and identically distributed, an invertible representation of this model is

$$Y_t = c + \varepsilon_t - \frac{7}{11}\varepsilon_{t-1} + \frac{1}{11}\varepsilon_{t-2} + \left(\frac{1}{1089}\sqrt{61305} + \frac{4096}{35937(8171+33\sqrt{61305})} + \frac{8164}{35937}\right)\varepsilon_{t-3}.$$

The ETSA procedure then calculates the spectral density function for the model, which in this case is

$$f(\omega) = \frac{1}{\pi}\left(\frac{196}{25} - \frac{158}{25}\cos(\omega) - \frac{48}{25}\cos(2\omega) + \frac{22}{5}\cos(3\omega)\right) \qquad 0 < \omega < \pi.$$

Finally, the ETSA procedure displays two realizations of the process.

REFERENCES

Birnbaum, Z. W. 1952. "Numerical Tabulation of the Distribution of Kolmogorov's Statistic for Finite Sample Size". *Journal of the American Statistical Association* 47: 425–441.

Box, G., G. Jenkins. 1994. *Time Series Analysis: Forecasting & Control*. 3rd ed. Upper Saddle River, New Jersey: Prentice–Hall, Inc.

Drew, J. H., D. L. Evans, A. G. Glen, and L. M. Leemis. 2008. *Computational Probability: Algorithms and Applications in the Mathematical Sciences*. New York: Springer.

Drew, J. H., A. G. Glen, and L. M. Leemis. 2000. "Computing the Cumulative Distribution Function of the Kolmogorov–Smirnov Statistic". *Computational Statistics and Data Analysis* 34 (1): 1–15.

Duggan, M. J., J. H. Drew, and L. M. Leemis. 2005. "A Test of Randomness Based on the Distance Between Consecutive Random Number Pairs. In *Proceedings of the 2005 Winter Simulation Conference*, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 741-748. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Efron, B., and R. J. Tibshirani. 1993. *An Introduction to the Bootstrap*. New York: Chapman & Hall.

Evans, D. L., J. H. Drew, and L. M. Leemis. 2008. "The Distribution of the Kolmogorov–Smirnov, Cramer–von Mises, and Anderson–Darling Test Statistics for Exponential Populations with Estimated Parameters". *Communications in Statistics–Simulation and Computation* 37 (7): 1396–1421.

Glen, A. G., D. L. Evans, and L. M. Leemis. 2001. "APPL: A Probability Programming Language". *The American Statistician* 55 (2): 156-166.

Kaczynski, W. H., L. M. Leemis, and J. H. Drew. 2012. "Transient Queueing Analysis". *INFORMS Journal on Computing* 24 (1): 10–28.

Leemis, L.M. 2006. "Lower System Reliability Bounds from Binary Failure Data Using Bootstrapping". *Journal of Quality Technology* 38 (1): 2–13.

Leemis, L. M., M. J. Duggan, J. H. Drew, J. A. Mallozzi, and K. W. Connell. 2006. "Algorithms to Calculate the Distribution of the Longest Path Length of a Stochastic Activity Network with Continuous Activity Durations". *Networks* 48 (3): 143–165.

Leemis, L. M., and J. T. McQueston. 2008. "Univariate Distribution Relationships". *The American Statistician* 62 (1): 45–53.

Leemis, L. M., B. W. Schmeiser, and D. L. Evans. 2000. "Survival Distributions Satisfying Benford's Law". *The American Statistician* 54 (4): 236–241.

Miller, S. J., editor. 2015. *Theory and Applications of Benford's Law*. In press. Princeton, New Jersey: Princeton University Press.

Vargo, E., R. Pasupathy, and L. Leemis. 2010. "Moment-Ratio Diagrams for Univariate Distributions". *Journal of Quality Technology* 42 (3): 276–286.

Webb, K. H., and L. M. Leemis. 2014. "Symbolic ARMA Model Analysis" *Computational Economics* 43 (33): 313–330.

Yang, J. X., J. H. Drew, and L. M. Leemis. 2012. "Automating Bivariate Transformations". *INFORMS Journal on Computing* 24 (1): 1–9.

AUTHOR BIOGRAPHY

LAWRENCE M. LEEMIS is a Professor in the Department of Mathematics at The College of William & Mary. He has formerly held faculty positions at The University of Oklahoma and Baylor University. He received his B.S., M.S, and Ph.D. from Purdue University. His email address is leemis@math.wm.edu.