APPL: A Probability Programming Language

Maj. Andrew G. Glen, Diane L. Evans and Lawrence M. Leemis

Statistical packages have been used for decades to analyze large data sets or to perform mathematically intractable statistical methods. These packages are not capable of working with random variables having arbitrary distributions. This paper presents a prototype probability package named APPL (A Probability Programming Language) which can be used to manipulate random variables. Examples illustrate its use.

KEY WORDS: Algorithms; Computer algebra systems; Continuous probability distributions.

Andrew Glen is a Major in the United States Army, 649 Batson Avenue, Ft. Sill, OK, 75303 (E-mail: andylisaglen@hotmail.com). Diane Evans is PhD student in Applied Science at The College of William & Mary (E-mail: devans@math.wm.edu). Lawrence Leemis is Professor and Chairman, Department of Mathematics, The College of William & Mary, Williamsburg, VA 23187–8795 (E-mail: leemis@math.wm.edu). The authors gratefully acknowledge the assistance of Don Barr and John Drew for their help with various portions of this research work. The second author acknowledges support from the Clare Boothe Luce Program. The third author acknowledges support from a faculty research assignment from The College of William & Mary and support from Purdue University's School of Industrial Engineering.

1. INTRODUCTION

Probability theory consists of a vast collection of axioms and theorems that, in essence, provides the scientific community many contributions that include:

- the naming and description of random variables that occur frequently in applications,
- the theoretical results associated with these random variables, and,
- the applied results associated with these random variables for statistical applications.

No one volume categorizes its work in exactly these three ways, but the literature's comprehensive works accomplish these goals. Voluminous works, such as Johnson, Kotz, and Balakrishnan (1995), or succinct works, such as Evans, Hastings, and Peacock (2000), are organized on the first contribution above, naming and describing random variables. Works such as Hogg and Craig (1995), Casella and Berger (1990), and David (1981) organize their efforts according to the second contribution, covering theoretical results that apply to random variables. Works such as Law and Kelton (2000), Lehmann (1986), and D'Agostino and Stephens (1986) concentrate on the statistical applications of random variables, and tailor their explanations of probability theory to the portions of the field that have applications in statistical analysis.

In all these works, as well as countless others, one stark omission is apparent. There is no mention of an ability to automate the naming, processing, or application of random variables. This omission is even more profound when one considers the tedious nature of the mathematics involved in the execution of many of these results for all but the simplest of examples. In practice, the level of tedium makes the actual execution untenable for many random variables. Automation of certain types of these procedures could eradicate this tedium.

Many existing statistical software packages, such as SPSS, SAS, and S-Plus, have numeric tools to apply statistical procedures. Most computer algebra systems, such as Maple and Mathematica, contain built-in statistical libraries with symbolic capabilities for use in

statistical computations. Further, Karian and Tanis (1999, preface) have developed procedures in Maple to serve as a supplement for "statistical analysis and also explorations within a rich mathematical environment." The only effort to automate probability manipulations and calculations that we have found to date is the Mathematica-based *mathStatica* due to Rose and Smith (2001). This paper introduces a Maple-based conceptual probability software package, referred to as "A Probability Programming Language" (APPL), that fills the existing technology gap in probability theory.

APPL has been successfully integrated into three classes (Mathematical Statistics, Reliability, and Probability) at The College of William & Mary. Its main use is as a time-saving device and experimental tool. Students have the ability to work small problems from first principles, and APPL allows them to both check their work and solve more time-consuming problems.

The problem below (Hogg and Craig 1995, page 287) is a sample of the type of question that can be worked by hand, then have the solution checked using APPL. Furthermore, a student can easily change any of the arguments in the problem (e.g., the sample size or critical value) to see if the results of the change match their intuition.

Example 1. Let X_1 and X_2 be iid observations drawn from a population with PDF

$$f(x) = \theta x^{\theta-1} \qquad 0 < x < 1,$$

where $\theta > 0$. Test H_0: $\theta = 1$ versus H_1: $\theta > 1$ using the test statistic X_1X_2 and the critical region $C = \{(X_1, X_2)|X_1X_2 \geq 3/4\}$. Find the power function and significance level α for the test.

The APPL code to compute the power function is

```
n := 2;
crit := 3 / 4;
assume(theta > 0);
X := [[x -> theta * x ^ (theta - 1)], [0, 1], ["Continuous", "PDF"]];
T := ProductIID(X, n);
power := SF(T, crit);
```

which yields

$$\Pr(\text{rejecting } H_0|\theta) = 1 - (3/4)^\theta + \theta(3/4)^\theta \ln(3/4).$$

The fact that the population distribution is non-standard indicates that the random variable `X` must be defined using the "list-of-sublists" data structure used in APPL (and described in Section 3.1). The `assume` procedure defines the parameter space. The `ProductIID` procedure computes the product of two independent and identically distributed random variables `X` and assigns it to the variable `T`. Last, the power function is defined using the survivor function (`SF`) to compute Pr(`T` $\geq$ `crit` = 3 / 4).

To compute the significance level of the test, the Maple statement

```
alpha := subs(theta = 1, power);
```

is required, yielding $\alpha = 1/4 + (3/4)\ln(3/4) \cong 0.0342$. To plot the power function requires the additional Maple statement

```
plot(power, theta = 0 .. 4);
```

Obviously, this example can be generalized for different sample sizes, population distributions, and critical values with only minor modification.

We end this section with three other problems that might be assigned in an introductory mathematical statistics class, along with the APPL code to solve them.

Example 2. Let $X_1, X_2, \ldots, X_{10}$ be iid U(0,1) random variables. Find the probability that the sum of $X_1, X_2, \ldots, X_{10}$ is between four and six; i.e., compute $\Pr\left(4 < \sum_{i=1}^{10} X_i < 6\right)$.

The APPL code to solve this problem is

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

which yields exactly $\frac{655177}{907200}$, or approximately 0.722.

The central limit theorem and Monte Carlo simulation are often used for problems like this one. The central limit theorem yields only one digit of accuracy here. Monte Carlo requires custom programming and the result is often stated as an interval around the true value. Also, each additional digit of accuracy requires a 100-fold increase in the number of replications.

Example 3. Let $X \sim \text{triangular}(1, 2, 4)$ and $Y \sim \text{triangular}(1, 2, 3)$. If X and Y are independent, find the distribution of $V = XY$.

The APPL code to solve this problem is

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

which returns the probability density function of V as

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

Example 4. Let X be a random variable with the distribution associated with the Kolmogorov–Smirnov (KS) test statistic in the all parameters known case with a sample size of $n = 6$ (Drew, Glen, and Leemis, 2000). Similarly, let Y be a KS random variable (all parameters known) with $n = 4$. If X and Y are independent, find $\text{Var}[\max\{X, Y\}]$.

The APPL code to solve this problem is

```
X := KSRV(6);
Y := KSRV(4);
Z := Maximum(X, Y);
Variance(Z);
```

which yields $\frac{10251047454659775800001920152 79}{837932101455829893097199763456 00}$ or approximately 0.0122337.

In the last three examples, an instructor would be hesitant to assign these as homework or exam questions due to the tedium required to determine their solutions. The APPL language allows students to see that tools exist that can get exact solutions to problems previously placed in the "intractable" class.

2. NOTATION AND NOMENCLATURE

APPL presently considers only continuous random variables. Similar data structures and algorithms are being developed for discrete distributions. Use is made of the following acronyms and functional notation for representations of a continuous random variable X:

- probability density function (PDF) $f_X(x)$,
- cumulative distribution function (CDF) $F_X(x) = \int_{-\infty}^{x} f_X(s)\,ds$,
- survivor function (SF) $S_X(x) = 1 - F_X(x)$,
- hazard function (HF) $h_X(x) = \frac{f_X(x)}{S_X(x)}$,
- cumulative hazard function (CHF) $H_X(x) = \int_{-\infty}^{x} h_X(s)\,ds$, and
- inverse distribution function (IDF) $F_X^{-1}(u)$.

The term "piecewise" refers to PDFs (and other functions) that can be constructed by piecing together various standard functions, such as polynomials, logarithms, exponentials, and trigonometric functions, e.g., the triangular(1, 2, 3) PDF has two pieces, each of which is a linear function. The abbreviation "N(μ, σ)" is used to refer to the normal distribution, where the second parameter is the standard deviation, not the variance. Also, "U(a, b)" is used to represent the uniform distribution with parameters a and b. Subscripts in parentheses represent order statistics, e.g., the r^{th} order statistic associated with a random sample $X_1, X_2, \ldots, X_n$ is denoted by $X_{(r)}$. The abbreviation "iid" is used to denote independent and identically distributed random variables. The terms "fully-specified," "semi-specified," and "unspecified" are used to describe the degree to which parameters are specified as constants or fixed parameters in a distribution. For example, the exponential(1) distribution is a fully-specified distribution. The Weibull(1, κ) and the N(0, σ) are both semi-specified distributions. The triangular(a, b, c) and exponential(λ) distributions are both unspecified. Typewriter font is used for APPL statements. The Maple input prompt "`>`" has not been included in the examples.

3. APPL OVERVIEW

The notion of probability software is different from the notion of applied statistical software. Probability theory contains numerous theorems (e.g., the sum of normal random variables is normal) and calculations (e.g. the product of two random variables) that require symbolic, algebraic manipulations. Applied statistical calculations (e.g., mean) are usually numeric manipulations of data based on known formulas or algorithms associated with models. Availability of computer algebra systems such as Maple and Mathematica facilitate the development of software that will derive functions, as opposed to computing numbers.

As described in Maple Version 6's help menu, its statistics package "provides data analysis functions, such as various means and quantiles, plotting functions, such as histograms and scatter plots, and data manipulation functions, such as weighted moving average and standard scores. Also available are least square fit, random number generation, analysis of variance (one way), data import from a file and numerical evaluation of statistical distributions." The procedures in this package mainly carry out numeric computations and plots associated with *data sets*. Although this is a valuable feature of the Maple software, these procedures do not define or operate on random variables. For example, the Maple statistical procedure `mean` is unable to determine the mean of a N(2, 4) random variable, but it can compute the mean of the data set {1, 2, 3, 3, 6, 7}. Since the Maple statistical procedures cannot be applied to probability distribution functions, obtaining probability results with these procedures is impossible.

Karian and Tanis's (1999) statistics supplement to Maple "consists of about 130 procedures written specifically to promote explorations of probabilistic and statistical concepts." Their supplement includes procedures for finding descriptive statistics (e.g., `Mean`, `Median`, and `Variance`), generating random samples from distributions, plotting (e.g., `BoxWhisker`, `PlotEmpPDF`, and `StemLeaf`), working with regression and correlation problems, producing the PDF and CDF of some distributions, finding percentiles of some distributions, producing confidence intervals, performing an analysis of variance, performing goodness-of-fit and

nonparametric tests (e.g., `QQFit`, `ChiSquareFit`, and `KSFit`), and computing the convolution of two random variables. While Karian and Tanis have utilized their efforts in building mainly an excellent statistical package powered by Maple, we have focused our energy into constructing a probability package for the manipulation of random variables.

At present, APPL is limited to univariate, continuous, independent random variables, and the complicated transformations and combinations that can result between independent random variables. A set of algorithms that derives functional representations of distributions and uses these functions in a manner that is typically needed in applications is described in the appendix. Specifically, algorithms have been developed that will conduct the following operations:

- supply a common data structure for representing the distributions of univariate random variables,
- convert any functional representation of a random variable into another functional representation using the common data structure, i.e., allowing conversion amongst the PDF, CDF, SF, HF, CHF, and IDF,
- provide straightforward instantiation of well-known distributions, such as the exponential, normal, and uniform distributions, with either numeric or symbolic parameters,
- determine the distribution of a simple transformation of a continuous random variable, $Y = g(X)$, including piecewise transformations,
- calculate the PDF of sums of independent random variables, i.e., $Z = X + Y$,
- calculate the PDF of products of independent random variables, i.e., $Z = XY$,
- calculate the PDF of the minimum and maximum of independent random variables, i.e., $Z = \min\{X, Y\}$ and $Z = \max\{X, Y\}$,
- calculate the PDF of the r^{th} order statistic from a sample of n iid random variables,

- generate random variates associated with a random variable,
- plot any of the six functional forms of any fully-specified distribution, e.g., the CDF,
- provide basic statistical abilities, such as maximum likelihood estimation, for distributions defined on a single interval of support,
- determine common summary characteristics of random variables, such as the mean, variance, and moment generating function,
- complement the structured programming language that hosts the software (in this case Maple) so that all of the above mentioned procedures may be used in mathematical and computer programming in that language.

3.1 The Common Data Structure

Implicit in a probability software language is a common, succinct, intuitive, and manipulatable data structure for describing the distribution of a random variable. This implies there should be one data structure that applies to the PDF, CDF, SF, HF, CHF, and IDF. The common data structure used in this software is referred to as the "list-of-sublists." Specifically, any functional representation of a random variable is presented in a list that contains three sublists, each with a specific purpose. The first sublist contains the ordered functions that define the functional forms of the distribution. The PDF representation of the triangular distribution, for example, would have the two linear functions that comprise the two segments of its PDF for its first sublist. The second sublist is an ordered list of real numbers that delineate the end points of the intervals for the functions in the first sublist. The end point of each interval is the start point of the succeeding interval. The third sublist indicates what distribution form the functions in the first sublist represent. The first element of the third sublist is either the string `"Continuous"` for continuous distributions or `"Discrete"` for discrete distributions. The second element of the third sublist shows which of the six functional forms is used in the first sublist. The string `"PDF"`, for example, indicates the list-of-sublists is currently a PDF list-of-sublists.

Examples:

- The following APPL statement, executed in Maple, assigns a list-of-sublists that represents the PDF of a U(0, 1) random variable to the variable `X`:

```
X := [[x -> 1], [0, 1], ["Continuous", "PDF"]];
```

- The triangular distribution has a PDF with two pieces to its distribution. The following statement defines a triangular(0, 1, 2) random variable `X` as a list-of-sublists:

```
X := [[x -> x, x -> 2 - x], [0, 1, 2], ["Continuous", "PDF"]];
```

- An exponential random variable `X` with a mean of 2 can be defined in terms of its hazard function with the statement:

```
X := [[x -> 1 / 2], [0, infinity], ["Continuous", "HF"]];
```

- Unspecified parameters can be represented symbolically. A N(θ, 1) random variable `X` can be defined with the statement:

```
X := [[x -> exp(-(x - theta) ^ 2 / 2) / sqrt(2 * Pi)],
      [-infinity, infinity], ["Continuous", "PDF"]];
```

- The parameter space can be specified by using the Maple `assume` function. Consider the random variable T with HF

$$h_T(t) = \begin{cases} \lambda & 0 < t < 1 \\ \lambda t & t \geq 1 \end{cases}$$

for $\lambda > 0$. The random variable `T` can be defined by the statements:

```
assume(lambda > 0);
T := [[t -> lambda, t -> lambda * t], [0, 1, infinity],
      ["Continuous", "HF"]];
```

- The syntax allows for the endpoints of intervals associated with the support of a random variable to be specified symbolically. A U(a, b) random variable `X` is defined by:

```
X := [[x -> 1 / (b - a)], [a, b], ["Continuous", "PDF"]];
```

Common, continuous, univariate distributions, such as the exponential and normal distributions, can also be defined using a procedure that creates the appropriate list-of-sublists.

The following distributions have been pre-defined in APPL: *ArcSin, ArcTan, Beta, Cauchy, Chi, ChiSquare, Erlang, Error, Exponential, ExponentialPower, ExtremeValue, F, Gamma, GeneralizedPareto, Gompertz, HyperbolicSecant, HyperExponential, HypoExponential, IDB, InverseGaussian, InvertedGamma, KS, LaPlace, LogGamma, Logistic, LogLogistic, LogNormal, Lomax, Makeham, Muth, Normal, Pareto, Rayleigh, StandardCauchy, StandardNormal, StandardTriangular, StandardUniform, T, Triangular, Uniform,* and *Weibull.*

Each distribution is defined in a procedure whose name has the letters `RV` appended to its name given above. A chi-square random variable with seven degrees of freedom X, for example, can be defined by the APPL statement

```
X := ChiSquareRV(7);
```

which defines the appropriate list-of-sublists. The reader interested in the details associated with APPL is encouraged to read the appendix prior to reading the next section.

4. APPLICATIONS AND EXAMPLES

A key contribution of APPL is that it provides a new way of viewing existing problems and opens new avenues for their solutions. This section provides examples where the problem-solving paradigm can shift due to the ability to find exact distributions of random variables. The four subsections that follow highlight some diverse areas of application that can be addressed with APPL. The section starts with relatively simple examples involving the central limit theorem (CLT). In these examples, one will be able to determine the error of the CLT approximations to distributions involving sums of independent random variables. The second subsection contains a demonstration of how to use APPL to generate mathematically intractable tables, graphs, and charts, effectively reducing the need for the volumes containing these entities. The third subsection contains a demonstration of how the software allows for modeling lifetimes using hazard functions. The final subsection provides a discussion on a simple methodology by which APPL furthers one's ability to identify outliers in samples. These four examples serve only to suggest the nature of applications of APPL; many more applications exist.

4.1 Exactness in Lieu of CLT Approximations

The CLT with iid samples can be used to facilitate inference about the population mean. The CLT implies that $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$ is asymptotically normally distributed when the X_i's are iid random variables with finite mean and variance. Application of the CLT becomes restricted, however, with small sample sizes, with skewed data, or in the presence of outliers (Moore and McCabe, 1999, pp. 516–517). In these cases, an alternative method is to determine the exact distribution of $\bar{X}$. Once the distribution $\bar{X}$ is found, one can use it to make statistical inferences about the population mean. A few examples follow.

Let $X \sim$ exponential(1). By the CLT, the distribution of $Y = \bar{X}$ is approximately $\mathrm{N}(1, \frac{1}{\sqrt{n}})$ for large n. For the case $n = 3$, one can compute the exact distribution of Y with the following statements:

```
X := ExponentialRV(1);
n := 3;
Y := ConvolutionIID(X, n);
Y := Transform(Y, [[x -> x / n], [0, infinity]]);
```

The resulting Erlang PDF is $f_Y(y) = \frac{27}{2} y^2 e^{-3y}$ for $y > 0$, which is predictably non-normal in shape since n is too small for the normality of the CLT to apply to such a skewed underlying distribution. Now consider the case of $n = 30$, often considered a sufficiently large sample size for an adequate CLT approximation. Here a computation of the error associated with the approximation is provided. To obtain the PDF of $\bar{X}$, change the second line of code to `n := 30;` and re-execute the Maple statements. The result is the following Erlang PDF:

$$f_Y(y) = \frac{24631531333923339843750000}{10577732774609} y^{29} e^{-30y} \qquad y > 0.$$

Table 1 contains selected fractiles for both the exact distribution of $\bar{X}$ and its CLT approximation, the $\mathrm{N}(1, \frac{1}{\sqrt{30}})$ distribution. For the selected fractiles, the approximation is only good

Table 1: Fractiles for exact and approximated distributions

		Fractile			
Approach	Distribution	0.9	0.95	0.975	0.99
Exact	X	1.240	1.318	1.388	1.473
CLT	$\mathrm{N}(1, \frac{1}{\sqrt{30}})$	1.234	1.300	1.358	1.425

to about one digit beyond the decimal point. The additional APPL statements to generate the CLT values in Table 1 are:

```
Z := NormalRV(1, 1 / sqrt(n));
alpha := [0.9, 0.95, 0.975, 0.99];
for i from 1 to 4 do
  evalf(IDF(Z, alpha[i]));
od;
```

For a related application, an old, approximate method of generating random variates from the standard normal distribution (Park and Leemis, 2000, for example) is:

$$Z^* = U_1 + U_2 + \cdots + U_{12} - 6,$$

where the U_i's are iid U(0, 1) random variables. Using APPL, the PDF of Z^* is

$$f_{Z^*}(x) = \begin{cases} \frac{1}{39916800}\,(x+6)^{11}, & -6 < x < -5 \\ -\frac{1}{3628800}\,x^{11} - \frac{1}{67200}\,x^{10} - \frac{11}{30240}\,x^9 - \frac{107}{20160}\,x^8 - \frac{517}{10080}\,x^7 - \frac{2477}{7200}\,x^6 - \frac{11737}{7200}\,x^5 & \\ -\frac{54797}{10080}\,x^4 - \frac{250657}{20160}\,x^3 - \frac{1113317}{60480}\,x^2 - \frac{4726777}{302400}\,x - \frac{18595037}{3326400}, & -5 < x < -4 \\ \frac{1}{725760}\,x^{11} + \frac{1}{17280}\,x^{10} + \frac{11}{10080}\,x^9 + \frac{7}{576}\,x^8 + \frac{99}{1120}\,x^7 + \frac{631}{1440}\,x^6 & \\ +\frac{1199}{800}\,x^5 + \frac{1009}{288}\,x^4 + \frac{12199}{2240}\,x^3 + \frac{9385}{1728}\,x^2 + \frac{38533}{11200}\,x + \frac{894727}{665280}, & -4 < x < -3 \\ -\frac{1}{241920}\,x^{11} - \frac{1}{8064}\,x^{10} - \frac{11}{6720}\,x^9 - \frac{25}{2016}\,x^8 - \frac{33}{560}\,x^7 - \frac{13}{72}\,x^6 - \frac{143}{400}\,x^5 & \\ -\frac{239}{504}\,x^4 - \frac{583}{1120}\,x^3 - \frac{1619}{3024}\,x^2 - \frac{781}{5600}\,x + \frac{61297}{166320}, & -3 < x < -2 \\ \frac{1}{120960}\,x^{11} + \frac{1}{6720}\,x^{10} + \frac{11}{10080}\,x^9 + \frac{1}{252}\,x^8 + \frac{11}{1680}\,x^7 + \frac{1}{360}\,x^6 + \frac{11}{1200}\,x^5 & \\ +\frac{25}{504}\,x^4 + \frac{11}{3360}\,x^3 - \frac{563}{3024}\,x^2 + \frac{11}{50400}\,x + \frac{65521}{166320}, & -2 < x < -1 \\ -\frac{1}{86400}\,x^{11} - \frac{1}{14400}\,x^{10} + \frac{1}{1440}\,x^8 - \frac{23}{3600}\,x^6 + \frac{31}{720}\,x^4 - \frac{809}{4320}\,x^2 + \frac{655177}{1663200}, & -1 < x < 0 \\ \frac{1}{86400}\,x^{11} - \frac{1}{14400}\,x^{10} + \frac{1}{1440}\,x^8 - \frac{23}{3600}\,x^6 + \frac{31}{720}\,x^4 - \frac{809}{4320}\,x^2 + \frac{655177}{1663200}, & 0 < x < 1 \\ -\frac{1}{120960}\,x^{11} + \frac{1}{6720}\,x^{10} - \frac{11}{10080}\,x^9 + \frac{1}{252}\,x^8 - \frac{11}{1680}\,x^7 + \frac{1}{360}\,x^6 - \frac{11}{1200}\,x^5 & \\ +\frac{25}{504}\,x^4 - \frac{11}{3360}\,x^3 - \frac{563}{3024}\,x^2 - \frac{11}{50400}\,x + \frac{65521}{166320}, & 1 < x < 2 \\ \frac{1}{241920}\,x^{11} - \frac{1}{8064}\,x^{10} + \frac{11}{6720}\,x^9 - \frac{25}{2016}\,x^8 + \frac{33}{560}\,x^7 - \frac{13}{72}\,x^6 + \frac{143}{400}\,x^5 & \\ -\frac{239}{504}\,x^4 + \frac{583}{1120}\,x^3 - \frac{1619}{3024}\,x^2 + \frac{781}{5600}\,x + \frac{61297}{166320}, & 2 < x < 3 \\ -\frac{1}{725760}\,x^{11} + \frac{1}{17280}\,x^{10} - \frac{11}{10080}\,x^9 + \frac{7}{576}\,x^8 - \frac{99}{1120}\,x^7 + \frac{631}{1440}\,x^6 - \frac{1199}{800}\,x^5 & \\ +\frac{1009}{288}\,x^4 - \frac{12199}{2240}\,x^3 + \frac{9385}{1728}\,x^2 - \frac{38533}{11200}\,x + \frac{894727}{665280}, & 3 < x < 4 \\ \frac{1}{3628800}\,x^{11} - \frac{1}{67200}\,x^{10} + \frac{11}{30240}\,x^9 - \frac{107}{20160}\,x^8 + \frac{517}{10080}\,x^7 - \frac{2477}{7200}\,x^6 + \frac{11737}{7200}\,x^5 & \\ -\frac{54797}{10080}\,x^4 + \frac{250657}{20160}\,x^3 - \frac{1113317}{60480}\,x^2 + \frac{4726777}{302400}\,x - \frac{18595037}{3326400}, & 4 < x < 5 \\ -\frac{1}{39916800}\,x^{11} + \frac{1}{604800}\,x^{10} - \frac{1}{20160}\,x^9 + \frac{1}{1120}\,x^8 - \frac{3}{280}\,x^7 + \frac{9}{100}\,x^6 - \frac{27}{50}\,x^5 & \\ +\frac{81}{35}\,x^4 - \frac{243}{35}\,x^3 + \frac{486}{35}\,x^2 - \frac{2916}{175}\,x + \frac{17496}{1925}, & 5 < x < 6. \end{cases}$$

The following statements determine the exact distribution of this 12-fold convolution and the fourth moment of Z^* and Z:

```
U := UniformRV(0, 1);
```

```
U12 := ConvolutionIID(U, 12);
Zstar := Transform(U12, [[x -> x - 6], [-infinity, infinity]]);
Z := NormalRV(0, 1);
Kurtosis(ZStar);
Kurtosis(Z);
```

The distribution of Z^* is useful because, like the standard normal distribution, its mean is zero, its variance is one, and its skewness is zero. The first difference between Z^* and Z is in the fourth moment, $E\left[\left(\frac{Z^*-E[Z^*]}{\sqrt{V[Z^*]}}\right)^4\right] = E\left[(Z^*)^4\right]$, which equals 2.9 for Z^* and 3.0 for a standard normal random variable Z.

In the overlaid Maple plots of the standard normal PDF and $f_{Z^*}(x)$ shown in Figure 1, it is

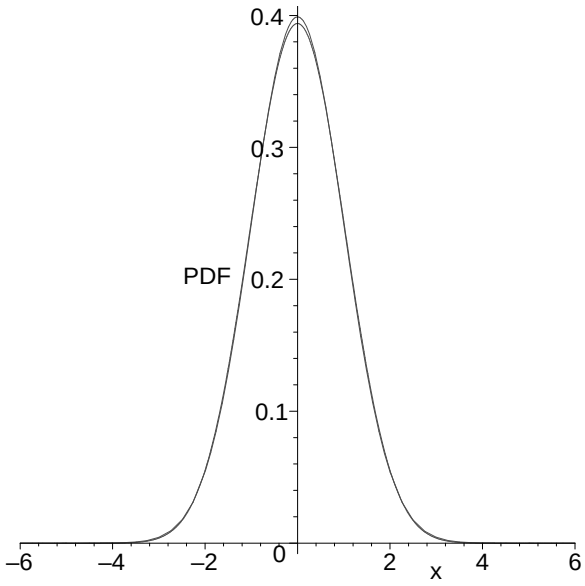

Figure 1: Overlaid plots of $f_{Z^*}(x)$ and the standard normal PDF.

apparent that this approximate distribution fits the standard normal distribution in the PDF very nicely except around $x = 0$ [where $f_Z(0) = \frac{1}{\sqrt{2\pi}} \cong 0.399$ and $f_{Z^*}(0) = \frac{655177}{1663200} \cong 0.394$] and, of course, in the tails, since the approximate distribution only has support on $(-6, 6)$. To obtain overlaid plots of these two distributions, the APPL `PlotDist` procedure and Maple `plots[display]` procedure are used.

```
Plot1 := PlotDist(Zstar, -6, 6):
Plot2 := PlotDist(Z, -6, 6):
plots[display]({Plot1, Plot2});
```

4.2 A Mathematical Resource Generator

APPL is also able to generate tables, graphs, and charts. The term *resource* refers to tables, graphs, and charts, to include such items as means, critical points, variances, and so on. APPL can be invoked to reduce the need for such tables. Additionally, since these computations can be produced automatically, programs can be designed to produce needed values in real time, as opposed to being restricted to knowing in advance which element of which table will be needed for program execution. There are many references for critical tables for common distributions, in addition to common tables given in introductory statistics tables, that software such as this (and others) effectively replace. Highlighted, however, will be two recent books, *CRC Handbook of Tables for Order Statistics from Inverse Gaussian Distributions with Applications* (Balakrishnan and Chen, 1997) and *CRC Handbook of Tables for the Use of Order Statistics in Estimation* (Harter and Balakrishnan, 1996). These books were selected because of the relatively complex nature of the information they present. Also, they were chosen to highlight that even as late as 1997, the scientific community had to rely on tables and charts for a significant amount of probabilistic information.

First is a comparison of Balakrishnan and Chen's text with APPL's capabilities. This book begins with 39 pages of theory and references to the inverse Gaussian (IG) distribution and its order statistics, followed by 645 pages of PDF plots, expected value tables, covariance tables, estimation tables, and so forth. APPL is able to create a number of these figures and tables, and in some cases goes beyond those presented in the text. Page 50, for example, shows the $\mathrm{N}(0,1)$ and the standardized IG(0.8) PDFs overlaid in a single plot. The PDF of a standardized IG random variable X (which is listed as equation (2.14) on page 7 of the text) is:

$$f_X(x) = \frac{1}{\sqrt{2\pi}} \left(\frac{3}{3+kx}\right)^{3/2} e^{-3x^2/(6+2kx)} \qquad -\frac{3}{k} < x < \infty\,.$$

The two plots may be overlaid and animated for k increasing from zero to one as follows:

```
Z := NormalRV(0, 1);
X := [[x -> (3 / (3 + k * x)) ^ (3 / 2) *
            exp(-3 * x ^ 2 / (6 + 2 * k * x)) / sqrt(2 * Pi)],
```

```
      [-3 / k, infinity], ["Continuous", "PDF"]];
NormalExpr := op(unapply(Z[1](x))(x));
InvGaussExpr := op(unapply(X[1](x))(x));
unassign('k');
plots[animate]({NormalExpr, InvGaussExpr}, x = -4 .. 4, k = 0 .. 1);
```

The plot is shown in Figure 2 for $k = 0.8$. To execute the animation, first select the plot by clicking on it. Then choose "Play" from the "Animation" menu. Balakrishnan and Chen use

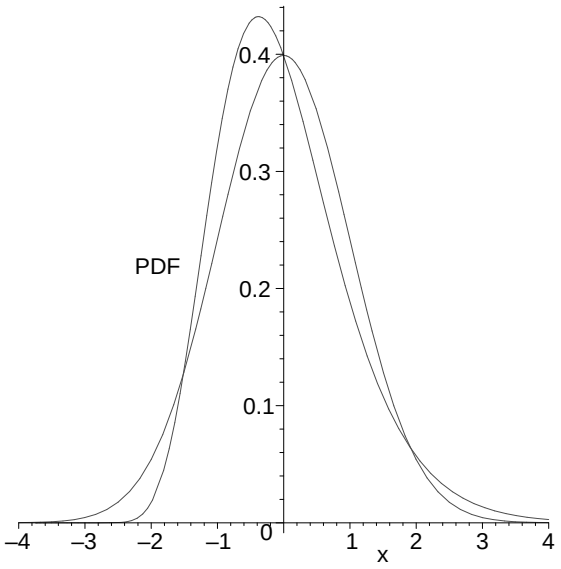

Figure 2: Overlaid plots of the standard normal and standard IG(0.8) distributions.

such plots to show how the two distributions separate as k increases from zero. For plotting the IG PDF, one is not limited to the standardized version of the distribution. One could plot any IG PDF, CDF, HF, and so on. Clearly, one is not limited to just PDF plots in APPL, nor is one limited to plots of only standardized IG distributions.

Another recent resource, *CRC Handbook of Tables for the Use of Order Statistics in Estimation* (Harter and Balakrishnan, 1996), also gives extensive charts for various applications of order statistics. Appendix C of this book, covering pages 326–530, lists tables of various means, variances, and covariances of order statistics from many common distributions. For example, Table C1.1 gives means of order statistics for the $N(0, 1)$ distribution, Table C2.1 for the means of exponential(1) order statistics, and so on. APPL can replicate such tables. For instance, to produce the mean of the $n = 8$, $r = 6$ order statistic from the Weibull(1, 0.5) distribution, use the statements:

```
X := WeibullRV(1, 0.5);
X86 := OrderStat(X, 8, 6);
Mean(X86);
```

which returns 1.760598073, compared to 1.76060 in Harter and Balakrishnan (1996, p. 392).

Unlike Harter's text, the software can go beyond computing means of order statistics. One can, for example, compute the 98th percentile of the same Weibull order statistic and give the probability that the order statistic exceeds 1.92 with the additional statements:

```
evalf(allvalues(IDF(X86, 0.98)));
SF(X86, 1.92);
```

The correct real values returned are 6.48028842 and 0.321864302, respectively. Furthermore, we are not limited to the relatively few base distributions that Harter and Balakrishnan present in their Appendix C, but can enter any Weibull parameter values.

There is no implication that these, or any of the vast resource collections, are archaic. We envision this software augmenting the information presented in texts such as these. Foreseen is a shifting away from looking up values in tables; instead, one will encode the creation of necessary values directly into programs. The first 208 pages of Harter and Balakrishnan's text, for example, still review theory behind the creation and use of order statistics. APPL adds to the resource base that books such as these provide.

4.3 Modeling with Hazard Functions

The capabilities of APPL allow a shift in the paradigm of parameterized model design in new ways. The reliability of a system is often given in terms of the instantaneous rate of failure, i.e., the hazard function. Reliability texts often classify lifetime distributions into *distribution classes* that are named after the shape of the hazard function. Leemis (1995, Chapter 3), among others, defines many of these classes, to include the increasing failure rate (IFR), the decreasing failure rate (DFR), and the bathtub-shaped (BT) hazard function.

One possible use for APPL is to model systems using a hypothesized shape of a hazard function. Should a system be hypothesized to have a bathtub-shaped HF for example, there are only a few common distributions with HFs with such a shape. Leemis (1995, p. 100) lists only two of 16 common reliability distributions as having a BT-shaped HF. Instead of being

limited to these distributions, one may hypothesize a family of distributions with BT-shaped HFs. For example, a second-order polynomial HF of the form $a(x-b)^2$ will have the BT shape as long as $a > 0$ and $b > 0$.

Here is an example of using APPL to model with hypothesized HFs. Consider a sample of failure times [1, 11, 14, 16, 17]. Assuming it is hypothesized that the system should be fit by the quadratic BT-shaped HF distribution, one can fit these data to the unspecified distribution as follows.

```
assume(a > 0);
assume(b > 0);
T := [[t -> a * (t - b) ^ 2], [0, infinity], ["Continuous", "HF"]];
PDF(T);
```

The last statement returns the general form of the PDF:

$$f_T(t) = a(t^2 - 2tb + b^2)\, e^{-t\,a\,(t^2-3\,t\,b+3\,b^2)/3} \qquad t > 0.$$

One can find the values $\hat{a}$ and $\hat{b}$ that maximize the likelihood function as follows:

```
data := [1, 11, 14, 16, 17];
hat := MLE(T, data, [a, b]);
```

The results are: $\hat{a} = 0.0037937$, $\hat{b} = 5.93512$. The plot of the SF for T overlaid on the empirical SF is shown in Figure 3.

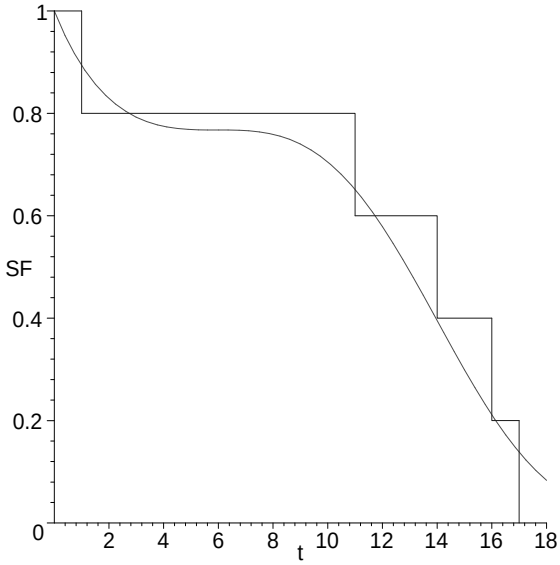

Figure 3: The SF of the hypothesized BT-shaped hazard function fit to the sample [1, 11, 14, 16, 17] overlaid on the empirical SF.

For another example of modeling in terms of hazard functions, let us hypothesize that risk to a system is seasonal. Such is the case in many applications such as structures at risk to occurrences of category five hurricanes. A periodic hazard function might be modeled in such a case. Suppose one hypothesizes a family of periodic HF distributions having HFs of the form

$$h_T(t) = a + b\,\sin(c\,t), \qquad\qquad t > 0;\; a > |b|, \text{ and } a, b, c \in \Re\,.$$

The parameter a represents a measure of the long-term constant risk associated with T. In other words, increased values of a correspond to a higher likelihood of chance failures. The parameters b and c control the amplitude and period of the HF, modeling the severity and length of the cyclic stresses. One can instantiate the unspecified distribution with the following statement

```
T := [[t -> a + b * sin(c * t)], [0, infinity], ["Continuous", "HF"]];
```

and the PDF may be found with `PDF(T)`. Letting the parameters take on values of $a = 1$, $b = 0.5$, and $c = 10$, one gets the PDF (using the `PlotDist` procedure) plotted in Figure 4, a peculiarly multi-modal PDF that decreases exponentially over time.

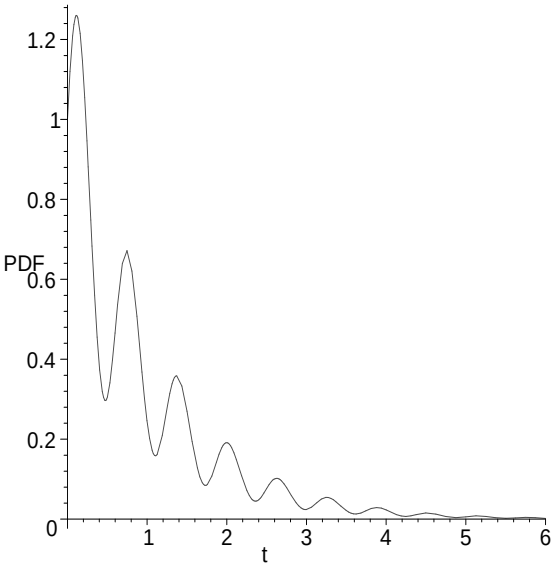

Figure 4: The PDF of the distribution having periodic hazard function $h_T(t)$ with parameters $a = 1$, $b = 0.5$, and $c = 10$.

4.4 Outlier Detection

The theory and practice of identifying outliers in a data set is another contribution provided by APPL. The literature contains ample methods for identifying outliers in samples from a normal population. Regarding detection of outliers, D'Agostino and Stephens (1986, p. 497) write "We shall discuss here only the underlying assumption of normality since there is very little theory for any other case." Sarhan and Greenberg (1962, p. 302) and David (1981) propose a number of test statistics based on standardized order statistics of normally distributed data. They provide charts of acceptance and rejection regions in lieu of p-values. For instance, in the fully-specified case where μ and σ are known, Sarhan and Greenberg propose the test statistic $(X_{(n)} - \mu)/\sigma$ for testing for extreme observations in normal samples. APPL is a useful tool for calculating the distribution of order statistics.

APPL may contribute to the outlier detection problem in at least three ways:

1. There is no need to standardize the test statistic, since the distribution of the r^{th} order statistic may be found with the procedure `OrderStat`.

2. One need no longer rely on charts of rejection regions, since once the distribution of the r^{th} order statistic is known, one can directly calculate p-values.

3. One is not limited to the assumption of normality. APPL facilitates finding the distribution of the r^{th} order statistic of a wide range of distributions.

Here are a few examples. In the sample {1, 2, 3, 4, 5, 10}, it is possible that the last element in the sample is an outlier. Assuming that the population has a mean of $\mu = 3$, one could find the order statistic probability $\Pr(X_{(6)} \geq 10)$ for samples of size $n = 6$ from various underlying distributions. Let us consider the following possible population distributions, each with a mean of 3 (set $\lambda = \sqrt{\pi}/6 \cong 0.295$ for the Weibull case): Weibull(0.295, 2), exponential(1/3), N(3, 2), and N(3, 4). The probability $\Pr(X_{(6)} \geq 10)$ is the significance level against the null hypothesis that $X_{(6)}$ came from this underlying distribution. The four probabilities can be found with the following statements:

```
X1 := WeibullRV(sqrt(Pi) / 6, 2);
X2 := ExponentialRV(1 / 3);
X3 := NormalRV(3, 2);
X4 := NormalRV(3, 4);
evalf(SF(OrderStat(X1, 6, 6), 10));
evalf(SF(OrderStat(X2, 6, 6), 10));
evalf(SF(OrderStat(X3, 6, 6), 10));
evalf(SF(OrderStat(X4, 6, 6), 10));
```

The results are shown in Table 2. An interpretation is that for $n = 6$ the value 10 is an

Table 2: $\Pr(X_{(6)} \geq 10)$ for $n = 6$ for several population distributions

Distribution	$\Pr(X_{(n)} \geq 10)$
Weibull$(0.295, 2)$	0.000973
exponential$(1/3)$	0.195839
N$(3, 2)$	0.001395
N$(3, 4)$	0.217532

outlier with respect to the N(3, 2) distribution at the 0.0014 level of significance and to the Weibull(0.295, 2) population distribution at the 0.001 level of significance. It is not a significant outlier to the other two population distributions.

An extension to this detection example is found in the last two population distributions, the normal distributions with σ equal to 2 and 4. One sets a specified level of significance, say $\alpha = 0.05$, and solves for the value of σ that will show a maximum value of 10 to be an outlier for the normal family of distributions. In effect what is desired is a solution to the following. Letting $X \sim$ N(3, σ), find σ such that $\Pr(X_{(6)} < 10) = 0.95$. One can find σ exactly by manipulating the CDF list-of-sublists representation of $X_{(6)}$ in the following way.

The first element of the first sublist contains the CDF of the sixth order statistic, so it is set equal to 0.95 and solved for σ. The following code produces the solution $\sigma \cong 2.933571640$.

```
X := NormalRV(3, sigma);
X6 := OrderStat(X, 6, 6);
X6 := CDF(X6);
solve(X6[1][1](10) = 0.95, sigma);
```

The first line of code sets the semi-specified normal distribution as the population distribution for X. The second line determines the PDF of $X_{(6)}$. The third line converts the PDF of $X_{(6)}$ to a CDF. The last line isolates the first element of the first sublist of `X6` which is the

CDF function of the independent variable x and the unknown parameter σ. The entry `(10)` provides the independent variable a value of 10. Then the Maple `solve` procedure solves for the unknown parameter `sigma` which represents σ. Using 0.95 as the right-hand side of the equation, as opposed to 95/100, causes Maple to find the floating point solutions.

Similarly, outlier detection can be performed for other population distributions for other parameters of interest given a specific minimum or maximum value.

5. CONCLUSION

APPL consists of Maple procedures that allow a modeler to define and perform operations on continuous, univariate random variables. The examples given in this paper illustrate how APPL's ability to automate these calculations can extend the reaches of probability theory. Many of the examples can be used in probability or mathematical statistics classes. APPL has spawned additional research on topics such as:

- the Kolmogorov–Smirnov statistic (Drew, Glen, and Leemis, 2000),
- order statistics from discrete populations (Evans, Leemis, and Drew, 2000),
- goodness-of-fit testing (Glen, Leemis, and Barr, 2000),
- Benford's Law (Leemis, Schmeiser, and Evans, 2000),
- input modeling for discrete-event simulation (Evans and Leemis, 2000),
- convolutions of discrete random variables (Evans, Leemis, and Drew, 2001).

Work is presently underway to extend APPL procedures to discrete univariate random variables. Future work in the area of computational probability may include multivariate distributions, time series analysis, and stochastic processes.

REFERENCES

Balakrishnan, N., and Chen, W. W. S. (1997), *CRC Handbook of Tables for Order Statistics from Inverse Gaussian Distributions with Applications*, New York: CRC.

Casella, G., and Berger, R. (1990), *Statistical Inference,* Pacific Grove, CA: Wadsworth and Brooks/Cole, Inc.

D'Agostino, R. B., and Stephens, M. A. (1986), *Goodness-of-Fit Techniques,* New York: Marcel Dekker.

David, H. A. (1981), *Order Statistics* (2nd ed.), New York: Wiley.

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

Evans, M., Hastings, N., and Peacock, B. (2000), *Statistical Distributions* (3rd ed.), New York: Wiley.

Evans, D., and Leemis, L. (2000), "Input Modeling Using a Computer Algebra System," *Proceedings of the 2000 Winter Simulation Conference*, J. A. Joines, R. R. Barton, K. Kang, P. A. Fishwick, eds., Institute of Electrical and Electronics Engineers, Orlando, Florida, 577–586.

Evans, D. L., Leemis, L. M., and Drew, J. H. (2001), "Algorithms for Computing the Distributions of Convolutions of Discrete Random Variables," Technical report, Mathematics Department, The College of William & Mary.

Evans, D., Leemis, L., and Drew, J. (2000), *Algorithms for Determining the Distribution of Order Statistics from Discrete Populations*, Technical report, Mathematics Department, The College of William & Mary.

Glen, A., Leemis, L., and Barr, D. (2000), "Order Statistics in Goodness-of-fit Testing," *IEEE Transactions on Reliability*, forthcoming.

Glen, A. G., Leemis, L. M., and Drew, J. H. (1997), "A Generalized Univariate Change-of-Variable Transformation Technique," *INFORMS Journal on Computing*, 9, 288–295.

Glen, A. G., Leemis, L. M., and Drew, J. H. (2000), "Computing the Distribution of the Product of Two Continuous Random Variables," Technical report, Mathematics Department, The College of William & Mary.

Hand, D. J., Daly, F., Lunn, A. D., McConway, K. J., and Ostrowski, E., Eds. (1994), *A Handbook of Small Data Sets*, New York: Chapman and Hall.

Harter, H. L., and Balakrishnan, N. (1996), *CRC Handbook of Tables for the Use of Order Statistics in Estimation*, New York: CRC.

Hogg, R. V., and Craig, A. T. (1995), *Mathematical Statistics* (5th ed.), Englewood Cliffs, NJ: Prentice Hall.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1995), *Continuous Univariate Distributions* (vol. 2, 2nd ed.), Upper Saddle River, NJ: Prentice Hall.

Karian, Z. A., and Tanis, E. A. (1999), *Probability and Statistics: Explorations with MAPLE* (2nd ed.), New York: Wiley.

Law, A. M., and Kelton, W. D. (2000), *Simulation Modeling and Analysis* (3rd ed.), New York: McGraw–Hill.

Leemis, L. (1995), *Reliability: Probabilistic Models and Statistical Methods,* Englewood Cliffs, NJ: Prentice Hall.

Leemis, L., Schmeiser, B., and Evans, D. (2000), "Survival Distributions Satisfying Benford's Law," *The American Statistician*, Volume 54, Number 4 (forthcoming).

Lehmann, E. L. (1986), *Testing Statistical Hypothesis* (2nd ed.), New York: Wiley.

Moore, D. S., and McCabe, G. P. (1999), *Introduction to the Practice of Statistics* (3rd ed.), New York: W. H. Freeman and Company.

Park, S., and Leemis, L. M. (2000), *Discrete-event Simulation: A First Course*, Department of Computer Science, The College of William & Mary.

Rose, C., and Smith, M. D. (2000), "Symbolic Maximum Likelihood Estimation with Mathematic," *Journal of the Royal Statistical Society,* Series D, 49, 229–240.

Rose, C., and Smith, M. D. (forthcoming: 2001), *Mathematical Statistics and Mathematica*,

New York: Springer–Verlag.

Sarhan A. E., and Greenberg, B. G. (1962), *Contributions to Order Statistics,* New York: Wiley.

Tanis, E. (1999), "A Review of Maple V© Release 5," *The American Statistician*, 53, 389–392.

APPENDIX

This appendix contains descriptions of the procedures comprising the core of APPL. The APPL tree diagram in Figure 4 summarizes the procedures described in Sections A.2 through A.11.

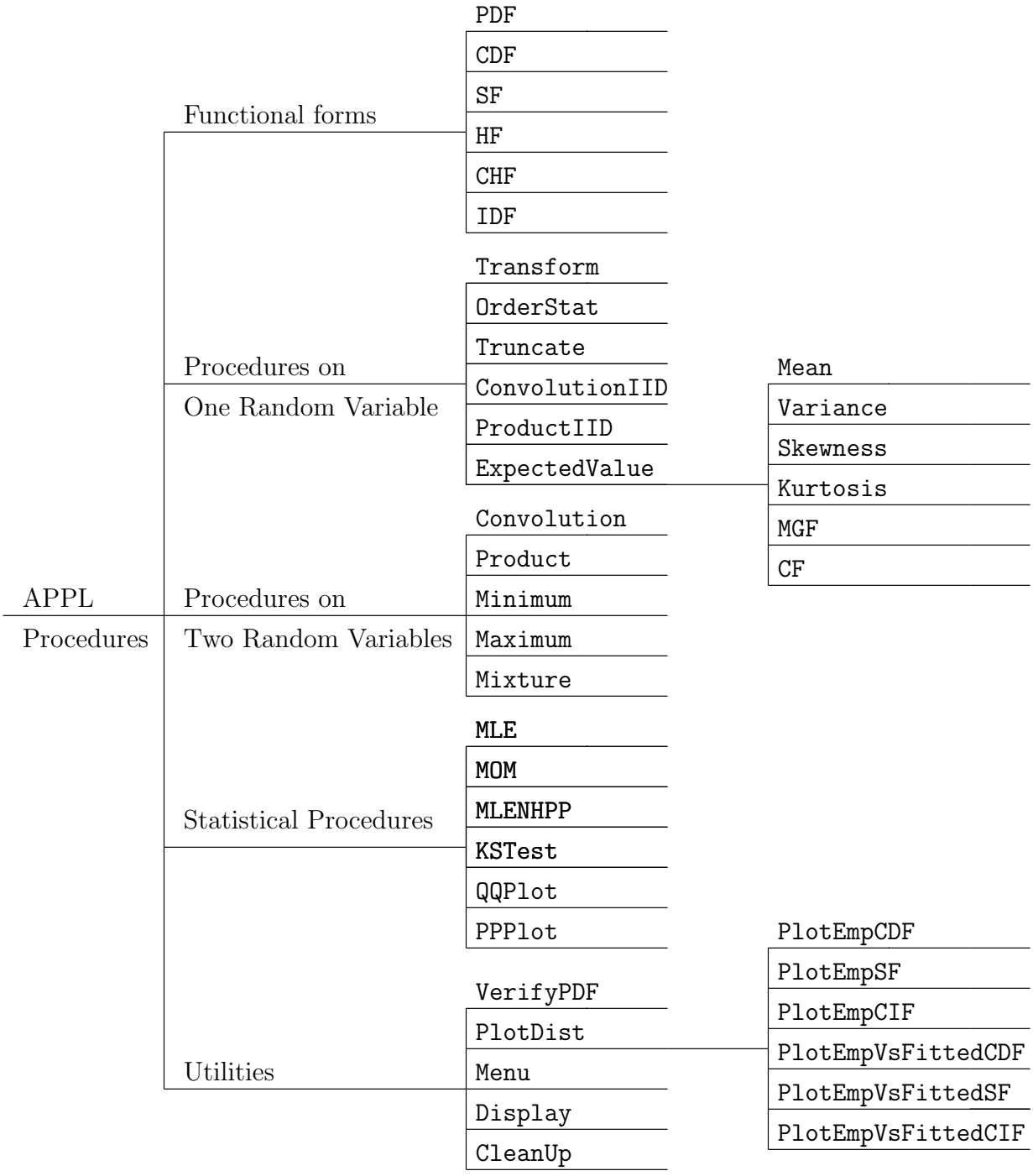

Figure 5. APPL tree diagram.

The notation used in defining APPL syntax in this appendix (in order of appearance) follows:

- $param1, param2, \ldots, paramk$ are numeric (e.g., 6, 2/3, 2.7) or symbolic parameters,
- X and Y are random variables defined in the list-of-sublists form described in Section 3.1,
- x is a numeric or symbolic value,
- brackets [ ] denote optional parameters,
- double quotes denote character strings,
- *low* and *high* are numeric values,
- g is a function,
- n and r are positive integers such that $n \geq r$,
- a capitalized parameter indicates that it must be entered as a Maple list.

A.1 Common Continuous, Univariate Distributions

Syntax: The statement

`X := `*DistributionName*`RV(`$param1, param2, \ldots, paramk$`);`

assigns to the variable `X` a list-of-sublists representation of the specified random variable. The choices for *DistributionName* are listed in Section 3.1.

Purpose: Instantiates common distributions. While the list-of-sublists is a data structure that lends itself to the mathematics of the software, it is not an instantly recognizable form for representing a distribution. APPL provides a number of simple procedures that take definitions of common distributions and convert them to a list-of-sublists format.

Special Issues: The suffix `RV` is added to each name to make it readily identifiable as a distribution assignment, as well as to avoid conflict with Maple-specific names such as `normal`

and `gamma`. The first letter of each word is capitalized, which is the case for all procedures in APPL. Usually the returned value is given as a PDF, but in the case of the IDB distribution (Leemis, 1995), a CDF is returned since the CDF of the IDB distribution is easier for Maple to manipulate (e.g., integrate, differentiate) than the PDF. Certain assumptions are made about unspecified parameters. For example, an assignment of an unspecified exponential random variable (see the second example below), will result in the assumption that $\lambda > 0$. This assumption, as with all other distributions' assumptions, are only applied to unspecified parameters.

Examples:

- An exponential(1) random variable X may be created with the following statement:

  ```
  X := ExponentialRV(1);
  ```

- An exponential(λ) random variable X, where $\lambda > 0$, may be created as follows:

  ```
  X := ExponentialRV(lambda);
  ```

- These procedures also allow a modeler to re-parameterize a distribution. An exponential ($\frac{1}{\theta}$) random variable Y, where $\theta > 0$, for example, may be created as follows:

  ```
  assume(theta > 0);
  Y := ExponentialRV(1 / theta);
  ```

- The semi-specified Weibull(λ, 1), where $\lambda > 0$, distribution may be created as follows:

  ```
  X := WeibullRV(lambda, 1);
  ```

  Note that this special case of the Weibull distribution is equivalent to an exponential(λ) distribution.

A.2 The Six Representations of Distributions

Syntax: The statement

$$\textit{FunctionalForm}(X, [x]);$$

returns either (1) the list-of-sublists format of the desired functional representation of the distribution, where *FunctionalForm* is `PDF`, `CDF`, `SF`, `HF`, `CHF`, or `IDF`, or (2) the value of the functional form at the value x.

Purpose: The 6×6 distribution conversion ability, a variation of the matrix outlined by Leemis (1995, p. 55), is provided so that the functional form of a distribution can be converted to and from its six well-known forms, the PDF, CDF, SF, HF, CHF, and IDF. This set of procedures takes one form of the distribution as an argument and returns the desired form of the distribution in the appropriate list-of-sublists format. For the one-argument call, the functional representation will be returned. For the two-argument call, the value of the function at the value x will be returned.

Special Issues: The procedures are fairly robust against non-specified parameters for the distributions that will be converted (see the fourth example below).

Examples:

- To obtain the CDF form of a standard normal random variable:

```
X := NormalRV(0, 1);
X := CDF(X);
```

or, equivalently, in a single line,

```
X := CDF(NormalRV(0, 1));
```

Since the CDF for a standard normal random variable is not closed form, APPL returns:

$$X := \left[\left[x \to \frac{1}{2}\,\mathrm{erf}\left(\frac{1}{2}\,x\,\sqrt{2}\right) + \frac{1}{2}\right],\, [-\infty,\, \infty],\, [\text{“Continuous”},\ \text{“CDF”}]\right]$$

where `erf` is the Maple error function defined by $\mathrm{erf}(x) = \frac{2}{\pi}\int_0^x e^{-t^2}dt$.

- If $X \sim \mathrm{N}(0, 1)$, then the following statements can be used to find $\Pr(X < 1.96) \cong 0.975$.

```
X := StandardNormalRV();
prob := evalf(CDF(X, 1.96));
```

- Should the hazard function of a unit exponential distribution be entered, its associated PDF may be determined as follows:

```
X := [[x -> 1], [0, infinity], ["Continuous", "HF"]];
X := PDF(X);
```

- For the case of unspecified parameters, the following statements convert an unspecified Weibull PDF to an unspecified Weibull SF:

```
X := WeibullRV(lambda, kappa);
X := SF(X);
```

which returns:

$$X := \left[\left[x \rightarrow e^{(-x^{\tilde{\kappa}}\,\tilde{\lambda}^{\tilde{\kappa}})}\right], [0, \infty], [\text{“Continuous”}, \text{“SF”}]\right]$$

The tildes after the parameters indicate that assumptions have been made concerning the parameters (i.e., $\lambda > 0$ and $\kappa > 0$) in the `WeibullRV` procedure.

- Finding a quantile of a distribution requires the IDF procedure. The 0.975 quantile of a Weibull(1, 2) distribution can be found with the statement

  ```
  quant := IDF(WeibullRV(1, 2), 0.975);
  ```

  If `0.975` is replaced by a random number, a Weibull variate is generated.

- The procedures handle distributions defined in a piecewise fashion. The random variable T with hazard function

$$h_T(t) = \begin{cases} \lambda & 0 < t < 1 \\ \lambda t & t \geq 1 \end{cases}$$

  for $\lambda > 0$, for example, has survivor function

$$S_T(t) = \begin{cases} e^{-\lambda t} & 0 < t < 1 \\ e^{-\lambda(t^2+1)/2} & t \geq 1 \end{cases}$$

  which can be found using the APPL statements

```
assume(lambda > 0);
T := [[t -> lambda, t -> lambda * t], [0, 1, infinity],
      ["Continuous", "HF"]];
SF(T);
```

- The procedures can be nested so that if the random variable `X` has been defined in terms of its PDF, then the statement

  ```
  X := PDF(CDF(HF(SF(CHF(IDF(X))))));
  ```

  does nothing to the list-of-sublists representation for `X`, assuming that all transformations can be performed analytically (e.g., exponential, Weibull).

A.3 `PlotDist`

Syntax: The statement

$$\texttt{PlotDist(}X, [low], [high]\texttt{);}$$

plots the current list-of-sublists defined X between the optional arguments *low* and *high* on a coordinate axis.

Purpose: To give a graphical representation of any list-of-sublists represented distribution. The arguments *low* and *high* define the minimum and maximum values desired on the horizontal axis. If defaulted, Maple defines the limits of the plot.

Special Issues: A distribution function must be fully-specified for a plot to be generated.

Examples:

- The following statements will generate the plot of the PDF for the triangular(1, 2, 3) distribution:

```
X := TriangularRV(1, 2, 3);
PlotDist(X);
```

- To plot the CHF of the exponential(1) distribution for $0 < t < F_X^{-1}(0.95)$, enter the statements:

```
X := ExponentialRV(1);
PlotDist(CHF(X), 0, IDF(X, 0.95));
```

- To see a progression of the five PDFs of the order statistics (the `OrderStat` procedure is introduced in Section A.6) for an exponential(1) distribution, enter:

```
X := ExponentialRV(1);
n := 5;
for i from 1 to n do
  PlotDist(OrderStat(X, n, i), 0, 10);
od;
```

  The result is five PDFs plotted sequentially. This sequence could be of use to an instructor explaining the progressive nature of order statistics to first-year probability students.

Algorithm: The algorithm loops through each of the pieces of the definition of the random variable, executing a Maple `plot` procedure to plot multiple functions on a single set of axes.

A.4 `ExpectedValue`

Syntax: The statement

`ExpectedValue(`$X, [g]$`);`

returns the expected value of a function of a random variable. If the optional argument g (a function) is not provided, then the procedure assigns g to be `x -> x`.

Purpose: Finds the expected value of a function of a random variable.

Special Issues: Procedures `Mean`, `Variance`, `Skewness`, `Kurtosis`, `MGF`, and `CF` are special cases of `ExpectedValue`.

Examples:

- To find the expected value and moment generating function of a standard normal random variable, type:

```
X := NormalRV(0, 1);
meanX := ExpectedValue(X);
Mx := MGF(X)
```

  The mean can also be determined by

```
meanX := Mean(StandardNormalRV());
```

- Unspecified distributions may also be used. The expected value of the square root of an exponential(λ) random variable is calculated with the statements:

```
X := ExponentialRV(lambda);
meanX := ExpectedValue(X, x -> sqrt(x));
```

Algorithm: The algorithm is a straightforward implementation of the following result. Let the continuous random variable X have PDF $f_X(x)$. Let $g(X)$ be a continuous function of X. The expected value of $g(X)$, when it exists, is given by

$$E[g(X)] = \int_{-\infty}^{\infty} g(x) \cdot f_X(x) dx \,.$$

A.5 `Transform`

Syntax: The statement

`Transform(`X, g`);`

returns the PDF of the transformed random variable in the list-of-sublists format.

Purpose: Determine the PDF of the transformation of a random variable of the form $Y = g(X)$, where $g(X)$ may be defined in a piecewise fashion.

Special Issues: The transformation function must also be defined in an altered list-of-sublists format. The modeler must break the transformation into piecewise monotone segments.

Examples:

- Let $X \sim \mathrm{U}(0, 1)$ and $Y = g(X) = 4X$. To determine the PDF of Y:

```
X := UniformRV(0, 1);
g := [[x -> 4 * x], [-infinity, infinity]];
Y := Transform(X, g);
```

- The following statements determine the distribution of the square of an inverse Gaussian random variable with $\lambda = 1$ and $\mu = 2$:

```
X := InverseGaussianRV(1, 2);
g := [[x -> x ^ 2], [0, infinity]];
Y := Transform(X, g);
```

- An example of using `Transform` to: (a) find the negative of a random variable is in Section A.8; (b) find the reciprocal of a random variable is in Section A.7; and (c) divide a random variable by a constant is in Section A.11.

Algorithm: The theorem which provides the basis for the algorithm and the details associated with the algorithm are found in Glen, Leemis, and Drew (1997).

A.6 `OrderStat`

Syntax: The statement

`OrderStat(`X, n, r`);`

returns the PDF of the r^{th} of n order statistics drawn from a population having the same distribution as the random variable X.

Purpose: Return the distribution of a specified order statistic. The procedure's arguments are defined as follows: the population distribution is represented in the list-of-sublists format,

the positive integer sample size n, and the positive integer order statistic index r, where $r \leq n$. The procedure returns the marginal PDF for the r^{th} order statistic in the list-of-sublists format.

Special Issues: This procedure is robust for unspecified parameters in the population distribution, unspecified n and r, and piecewise distributions.

Examples:

- The minimum of six iid exponential(1) random variables, which in turn is exponentially distributed with $\lambda = 6$, is found with the statements:

```
X := ExponentialRV(1);
Y := OrderStat(X, 6, 1);
```

- In this example, n, r, and the distribution are unspecified.

```
X := ExponentialRV(lambda);
Y := OrderStat(X, n, r);
```

  The result is the general form of the r^{th} order statistic from an exponential(λ) population:

$$Y := \left[\left[x \rightarrow \frac{n!\,(-\mathrm{e}^{(-\lambda\tilde{}\,x)} + 1)^{(r-1)}\,\lambda\tilde{}\,\mathrm{e}^{(-\lambda\tilde{}\,x\,(n-r+1))}}{(r-1)!\,(n-r)!}\right], [0, \infty], [\text{“Continuous”}, \text{“PDF”}]\right]$$

- As seen in the example in Section A.3 for `PlotDist`, the `OrderStat` procedure may be embedded in loops and other programming capabilities in Maple.

Algorithm: The algorithm is a straightforward implementation of the following result (Hogg and Craig, 1995, p. 198):

$$f_{X_{(r)}}(x) = \frac{n!}{(r-1)!(n-r)!} F_X(x)^{r-1} \cdot (1 - F_X(x))^{n-r} \cdot f_X(x).$$

A.7 `Product` and `ProductIID`

Syntax: The statement

`Product`(X, Y)`;`

returns the PDF of the product of X and Y.

Purpose: Compute the PDF of products of random variables. The arguments can be any list-of-sublists representation of a distribution. This algorithm is explained in Glen, Leemis, and Drew (2000). The similar procedure `ProductIID(`X, n`)` computes the PDF of the n-fold product of iid random variables.

Special Issues: The two random variables in the argument list are assumed to be independent, but not necessarily identically distributed in `Product`. Distributions do not have to be fully-specified, as seen in the third example. The algorithm may be used in conjunction with `Transform` to compute the PDF of ratios of random variables, as seen in the fourth example.

Examples:

- A plot of the PDF of the product of a standard normal and a U(0, 1) random variable is found with the following statements:

```
X := NormalRV(0, 1);
Y := UniformRV(0, 1);
PlotDist(Product(X, Y));
```

- The height of the PDF of the product of three independent standard normal random variables at one is found with:

```
X := NormalRV(0, 1);
Z := ProductIID(X, 3);
evalf(PDF(Z, 1));
```

  The PDF of Z is in terms of Maple's `BesselK` function.

- The PDF of the product of two independent unspecified exponential(λ) distributions is a PDF also in terms of Maple's `BesselK` function:

```
X := ExponentialRV(lambda);
Y := Product(X, X);
```

  The procedure returns the following list-of-sublists:

$$Y := [[v \rightarrow 2\,\lambda^{\sim 2}\,\text{BesselK}(0,\, 2\,\lambda^{\sim}\,\sqrt{v})],\, [0,\, \infty],\, [\text{“Continuous”},\, \text{“PDF”}]]$$

- For ratios of random variables, employ the transformation ability of the software as in

the following example, which could be used to calculate the distribution of a random rate, given independent distributions for distance and time:

```
U := UniformRV(0, 10);
T := ExponentialRV(1);
R := Product(U, Transform(T, [[x -> 1 / x], [0, infinity]]));
```

A.8 `Convolution` and `ConvolutionIID`

Syntax: The statement

`Convolution`(X, Y);

returns the PDF of the sum of X and Y.

Purpose: Return the PDF of the convolution of two independent random variables X and Y in the usual list-of-sublists format. Similarly, the procedure `ConvolutionIID`(X, n) computes the PDF of the n-fold convolution of iid random variables.

Special Issues: The random variables entered as arguments are assumed to be independent, but not necessarily having the same distribution in `Convolution`. The ability to compute differences of random variables is inherent in the software by employing the transformation ability, as in the third example below.

Examples:

- The sum of a standard normal random variable and a U(0, 1) random variable has a PDF found as follows:

  ```
  X := NormalRV(0, 1);
  Y := UniformRV(0, 1);
  Z := Convolution(X, Y);
  ```

- In a review of **Maple V© Student Version: Release 5** in *The American Statistician*, Tanis (1999, p. 391) does an example concerning the sum of two continuous random variables with PDF

  $$f_X(x) = \frac{3}{2}x^2 \qquad -1 < x < 1.$$

  The PDF and plot of the sum of three of these random variables is found in APPL as follows:

```
X := [[x -> (3 / 2) * x ^ 2], [-1, 1], ["Continuous", "PDF"]];
Y := ConvolutionIID(X, 3);
PlotDist(Y);
```

- To find the PDF of the difference between a uniform and exponential random variable:

  ```
  X := UniformRV(0, 10);
  Y := ExponentialRV(1);
  Z := Convolution(X, Transform(Y, [[y -> -y], [-infinity, infinity]]));
  ```

  This call to `Transform` finds the distribution of the opposite of the random variable Y, so the PDF of the random difference $Z = X - Y$ is computed.

Algorithm: To compute the convolution distribution of $Z = X + Y$, `Convolution` carries out the transformation $Z = \ln(e^X e^Y)$ using the `Transform` and `Product` procedures.

A.9 `Minimum`

Syntax: The statement

`Minimum(`X, Y`);`

returns the PDF of the minimum of the two random variables listed as arguments.

Purpose: Computes the PDF of the minimum of X and Y, entered in the usual list-of-sublists format, where X and Y are assumed to be independent and continuous.

Special Issues: The two random variables in the argument list are assumed to be independent, but not necessarily identically distributed. The procedure is robust for unspecified parameters for the distributions (see the third example below). The procedure is able to handle piecewise random variables, as illustrated in the first example below where two distributions with only one interval each in their PDFs result in a minimum with two intervals.

Examples:

- The PDF of the minimum of a standard normal random variable and a U(0, 1) random variable is found as follows:

  ```
  X := NormalRV(0, 1);
  Y := UniformRV(0, 1);
  Z := Minimum(X, Y);
  ```

- The PDF of the minimum of two unspecified iid Weibull random variables is found with:

```
X := WeibullRV(lambda, kappa);
Y := Minimum(X, X);
```

This call to `Minimum` returns `Y` as:

$$Y := \left[\left[x \to 2\,\lambda^{\sim\kappa^{\sim}}\,x^{(\kappa^{\sim}-1)}\,\kappa^{\sim}\,\mathrm{e}^{\left(-2\,\lambda^{\sim\kappa^{\sim}}\,x^{\kappa^{\sim}}\right)}\right], [0,\infty], [\text{“Continuous”}, \text{“PDF”}]\right]$$

Algorithm: The procedure uses the CDF technique to find the PDF of the minimum of two independent random variables. Special consideration is given to piecewise random variables, as the CDF technique requires the intervals to be aligned properly.

A.10 `Maximum`

Syntax: The statement

`Maximum(`X, Y`);`

returns the PDF of the maximum of X and Y.

Purpose: Compute the PDF of the maximum of X and Y, entered in the usual list-of-sublists format, where the X and Y are assumed to be independent and continuous.

Special Issues: The two random variables in the argument list are assumed to be independent, but not necessarily identically distributed. There are no separate procedures necessary for finding the PDF of the maximum or minimum of n iid random variables, since this is already possible with `OrderStat`.

Examples:

- The maximum of a standard normal random variable and a U(0, 1) random variable is found as follows:

```
X := NormalRV(0, 1);
Y := UniformRV(0, 1);
Z := Maximum(X, Y);
```

- The maximum of two independent unit exponential random variables, which could represent the system lifetime of a parallel arrangement of a two-component system, is found as follows:

```
X := ExponentialRV(1);
Z := Maximum(X, X);
```

Algorithm: This procedure relies on the `Minimum` and `Transform` procedures to determine the distribution of the maximum. Specifically, it determines the PDF of $Z = \max\{X, Y\}$ by performing the transformation $Z = -\min\{-X, -Y\}$.

A.11 Maximum Likelihood Estimation

Syntax: The statement

$$\texttt{MLE}(X, Data, Parameters, [Rightcensor])\texttt{;}$$

returns the MLE of the parameters listed in the list *Parameters* a result of the sample observations listed in *Data* for sampling from a population with a distribution represented by the random variable X.

Purpose: Compute real, and symbolic, when possible, estimates of parameters for distributions. The *Data* argument can either be a fully-specified list of real numbers or the unspecified list of symbols. The *Parameters* argument is a list of the unknown parameters to be estimated. The optional *Rightcensor* argument is a right-censor vector of ones and zeros, where one indicates an observed value and zero indicates a right-censored value.

Special Issues: Any distribution represented as a list-of-sublists may be used in the estimation procedure. The procedure is limited by Maple's solving capability, especially in the case of unspecified samples. An advantage of this approach to ML estimation is that the distribution of the estimator can be found, as in the fourth example below. Similar work has been done by Rose and Smith (2000) and incorporated into their mathStatica package.

Examples:

- For the semi-specified inverse Gaussian distribution where σ is known (or assumed) but μ is unknown, one finds $\hat{\mu} = 3$ for the data set 1, 2, 3, 4, 5 as follows:

  ```
  X := InverseGaussianRV(1, mu);
  muhat := MLE(X, [1, 2, 3, 4, 5], [mu]);
  ```

- Should both parameters need to be estimated, then the parameter list would include both of the unknown parameters as follows:

```
X := InverseGaussianRV(lambda, mu);
paramhat := MLE(X, [1, 2, 3, 4, 5], [lambda, mu]);
```

The call returns the estimated values in the form of a list assigned to `paramhat`, where `paramhat` is now the list `[300 / 37, 3]` corresponding to the MLEs $\hat{\lambda} = \frac{300}{37}$ and $\hat{\mu} = 3$.

- To illustrate a right-censored data set, consider the "time to vomit", in minutes, for $n = 21$ subjects placed in a cubical cabin mounted on a hydraulic piston and subjected to vertical motion for two hours (Hand et al., 1994, p. 13). Letting 0 represent a censored value and 1 represent an uncensored value, one can determine an estimate for the parameter λ of an exponential distribution with the statements:

```
X := ExponentialRV(lambda);
VData := [30, 50, 50, 51, 66, 82, 92, seq(120, i = 8 ..  21)];
RCensor := [1, 1, 0, 1, 0, 1, 1, seq(0, i = 8 ..  21)];
lambdahat := MLE(X, VData, RCensor);
```

These statements return the parameter estimate $\hat{\lambda} = \frac{5}{2101}$.

- For the case of a general random sample $X_1, X_2, \ldots, X_n$, i.e., the sample is unspecified, but n is fixed, one would type the following statements:

```
X := NormalRV(mu, 1);
muhat := MLE(X, [x1, x2, x3, x4, x5], [mu]);
```

This code returns the MLE $\bar{X}$ for $n = 5$. To find the distribution of this estimator:

```
Y := Transform(ConvolutionIID(X, 5), [[x -> x / 5], [-infinity, infinity]]);
```

One can then find its mean, variance, quantiles, and so forth, using APPL.