PMETRIC and PMETGEN: Programs For the Simulation & Analysis of Quantal Data

Version 2.0

Jeff Miller
Department of Psychology
University of Otago
Dunedin, New Zealand

Sept 17, 2002

Contents

1 Overview

This document describes two programs, one for data analysis (PMETRIC) and one for data generation (PMETGEN). Most users will mainly be interested in the data analysis program, but advanced users may also want to use the data generation program, in computer simulations.

2 Introduction to PMETRIC

PMETRIC estimates the parameters of a probability distribution from a data function relating the proportion of a certain (binary) response to a physical quantity. This type of data analysis—often called "probit" analysis—is used in several subject areas, including bioassay (analysis of dose/response curves) and psychophysics (analysis of psychometric functions). In brief, the program reads a file containing the observed data (e.g., quantal dose/response curve), and it computes either maximum-likelihood or minimum-chi-square estimates of the parameters (mean, median, standard deviation, etc) of the underlying probability distribution. It also computes the bootstrap standard error of each of each estimate, which simulations indicate are good variance estimators (Foster & Bischof, 1987, 1991).

In bioassay, for example, a researcher might want to determine the relationship between the dosage of a certain poison and the probability that a certain pest animal consuming that dose will die. In a typical study, each of k different dosages, $C_1 \ldots C_k$, is given to N_i different animals The number of deaths at dosage i, G_i, is counted to estimate mortality for that dosage. Such data are typically analyzed with a statistical model assuming that any given animal has a minimum lethal dosage and that the animal dies if and only if it is given a dosage greater than or equal to its minimum lethal dose. Thus, an observed G_i/N_i value is an estimate of the population proportion of animals for whom the lethal dose is less than or equal to C_i.

The analogous problem arises in psychophysical research examining psychometric functions. In this case, the C_i values might be intensities of a given auditory tone. The tone is played to an observer N_i times at each intensity value, and each time the observer indicates whether or not he heard it. The statistical model assumes that the observer has a minimum detectable intensity value (fluctuating across time), and that the observer reports hearing the tone on each presentation if and only if it is more intense than the minimum intensity value at that moment. Thus, an observed G_i/N_i value is an estimate of the probability that the instantaneous minimum detectable intensity value is less than or equal to C_i.

In standard probit analysis, the underlying probability distribution is assumed to be normal (i.e., Gaussian). PMETRIC allows this assumption but does not require it. Instead, the user may do the comparable analysis assuming a variety of alternative underlying distributional shapes (e.g., gamma, uniform), and the user may obtain nonparametric estimates using the Spearman-Kärber method (e.g., Epstein & Church, 1944; Kärber, 1931; Spearman, 1908). Based on an extensive simulation study, in fact, we would recommend that the Spearman-Kärber method be used under a wide variety of circumstances (Miller & Ulrich, 2001).

More generally, the type of analysis carried out by this program can be used with any data collected in the following situation (cf. Finney, 1978):

1. A researcher selects an ordered set of k constants $C_1, C_2, C_3, \ldots, C_k$ (e.g., $5, 10, 15, 20, \ldots, 50$) roughly spanning a probability distribution.

2. For each constant C_i, the researcher takes N_i independent random samples X_{ij} from the distribution, $j = 1, \ldots N_i$. The value of X_{ij} cannot be observed directly, however. Instead, the researcher observes *only* Y_{ij}, where
$$Y_{ij} = \begin{cases} 0 & \text{if } X_{ij} \le C_i \\ 1 & \text{if } X_{ij} > C_i \end{cases}$$

3. The data are summarized by counting the number of observations greater than each C_i:
$$G_i = \sum_{j=1}^{N_i} Y_{ij}$$

4. The problem is to estimate the probability distribution of the X_{ij} values from the k observed proportions, G_i/N_i, i=1…k.

With such data, the parameters of the distribution of the X_{ij} values can be estimated by either maximizing likelihood or minimizing chi-square. For any given set of parameter values, the likelihood of the specific ordered set of X values is

$$L = \prod_{i=1}^{k} p_i^{(N_i - G_i)} \cdot (1 - p_i)^{G_i}$$

where $p_i = CDF(C_i)$ with the given parameter values (Finney, 1971, chap. 5).[1] When requested to use the maximum-likelihood type of probit fit, PMETRIC adjusts parameters iteratively to maximize this value (actually, to minimize the negative of the natural logarithm of this value) using the numerical search algorithm of Rosenbrock (1960). The simplex minimum value that it reports at the end of its search is -Ln(L).

Alternatively, for any given set of parameter values, a chi-square goodness-of-fit test may be computed as (Guilford, 1936)[2]

$$\chi^2 = \sum_{i=1}^{k} N_i \frac{(\hat{p}_i - p_i)^2}{p_i \cdot (1 - p_i)}$$

where $p_i = CDF(C_i)$ with the given parameter values and $\hat{p}_i = (N_i - G_i)/N_i$. When requested to use the ChiSq type of probit fit, PMETRIC adjusts parameters iteratively to minimize this value. The simplex minimum value that it reports at the end of its search is the minimum obtained value of χ^2.

In standard probit analysis, the function $CDF(\cdot)$ refers to the cumulative normal probability function, but PMETRIC allows a variety of alternative cumulative distributions functions to be used (e.g., uniform, gamma).

3 PMETRIC Installation & Test

1. Unzip the distribution file. It will be named something like PMETRIC.ZIP, except that the last few letters of PMETRIC will be changed to numbers to reflect the current version (e.g., PMETRI11.ZIP is version 1.1). Note that subdirectories called "`In`" and "`WinFPC.OK`" are created; these are for test purposes (see point after next).
2. Move the program files PMETRIC.EXE and PMETGEN.EXE to a directory in your path.
3. If you like, you can run some quick tests of the program to make sure that everything is working properly on your machine and your version of the operating system. To do the quick tests, open a command window and change its current directory into the subdirectory "`In`". Then run the batch file GoWinFPC.Bat. The batch file should run a series of test runs, and it should produce numerous output files called *.rpt, *.mtb, *.out, *.prm, and gen*.dat. When the batch file is done, check that every newly created file in the "`In`" subdirectory is identical to the file with the same name in the "`WinFPC.OK`" subdirectory. If this check works, it is likely that everything is OK. If it fails, contact the author.

---

[1] To get the likelihood of the G_i values instead, you would include the binomial coefficients $\binom{N_i}{G_i}$ in this product. This has no effect on the parameter estimation, however, because these constant multipliers can be factored out, i.e.,

$$\prod_{i=1}^{k} \binom{N_i}{G_i} \cdot p_i^{(N_i - G_i)} \cdot (1 - p_i)^{G_i} = \prod_{i=1}^{k} \binom{N_i}{G_i} \prod_{i=1}^{k} \cdot p_i^{(N_i - G_i)} \cdot (1 - p_i)^{G_i}$$

I thank Rolf Ulrich for supplying the information in this footnote.

[2] The chi-square test can be derived by conceiving of the probit data set as a multinomial with k categories. Let N_i be the number of independent observations at each C_i, let G_i and $N_i - G_i$ be the numbers of successes and failures, and let p_i be the predicted probability of a success (i.e., CDF(C_i) in a yes/no task or $1/m + (1 - 1/m) \cdot$ CDF(C_i) in an m-alternative forced-choice task). Then the standard chi-square test for a multinomial is computed as

$$\chi^2 = \sum_{i=1}^{k} \left[\frac{(G_i - p_i \cdot N_i)^2}{p_i \cdot N_i} + \frac{((N_i - G_i) - (1 - p_i) \cdot N_i)^2}{(1 - p_i) \cdot N_i} \right]$$

This formula can be simplified to obtain the formula given by Guilford (1936). I thank Rolf Ulrich for supplying the information in this footnote.

4 PMETRIC Input Files

Before using PMETRIC, you must prepare one or two input files. You must always prepare a file containing the data (i.e., values of C_i, G_i, and $N_i - G_i$), and it should be given a name with the extension ".DAT". The other, optional input file contains control statements describing the desired analysis, and it should be given the extension ".RSP". RSP stands for "response", because these files convey the responses that you would give if you were controlling the program with a more traditional on-screen interface. If you are happy with the default analysis options, you can omit this file.

The files "`EXAMPL*.DAT`" and "`EXAMPL*.RSP`" show examples. Both types of input files must be plain text (ASCII) files.

4.1 Format of the Data File

As shown in "`Exampl1.DAT`", the basic data format includes three lines per data set. The first line gives the C_i values, in increasing order, with two extra values (for which no data were collected) called C_0 and C_{k+1} that are used in connection with the Spearman-Kärber method, as discussed later, and also used in some cases with percentile estimation. The second line gives the G_i values in the same order as the C_i values. The third line gives the $N_i - G_i$ values, also in the same order as the C_i values. Note that no values are specified for G_0, G_{k+1}, $N_0 - G_0$, or $N_{k+1} - G_{k+1}$.

Note also that spaces (not tabs) should be used to separate the numbers in the data file. The spacing is not significant to the program as long as different numbers are separated by at least one space, but of course it is easier for a person to read if the numbers line up in columns as shown in the example files.

As shown in "`Exampl2.DAT`", the program can also process several data sets in a given run (these will be referred to as different "CASEs"). The same sequence of lines simply repeats for each new case. Note that a blank line is required between cases.

If the stimulus values are the same for all cases, they need only be included for the first case, as shown in "`Exampl3.DAT`". The `CommonCs` parameter must be included in the RSP file to let PMETRIC if this shortcut has been used. In this case, a blank line appears after the C values are listed at the beginning of the file, and in addition there is a blank line after each case except the last (see file "`Exampl3.DAT`").

As described in section 7, the user may choose to specify values of N_i instead of $N_i - G_i$ on the third line of the input file by using the `TotalN` option. Alternatively, the user can omit this line altogether if N_i is a constant and the `FixedN` option is specified.

4.2 Format of the Control File

The file "`Exampl1.RSP`" shows a simple control file with these lines:

```
NProbit 1
   Normal(3,2)
WriteMTB
Seed Start Default.rng
```

The first line tells the program that it should perform the probit analysis using just one assumed underlying distribution, and the second line tells the program (a) this underlying distribution should be the normal distribution, and (b) 3 and 2 are reasonable starting guesses for its mean and standard deviation. The third line asks for an optional output file in the MTB format, and this is mainly for my convenience in testing. The fourth line tells the random number generator to start at the default seed rather than a random one. The purpose of this is to make sure that the bootstrapping gives exactly the same results when you run the program as I got when I ran it, so that our output values should be exactly equal.

Quite a few different analysis options can be controlled from the RSP control file, and these are described in detail in section 7. First, however, I tell how to run the program and describe the output that it produces.

Table 1: First Part of Output for Data in File “`Exampl1.Dat`”

—- Case 1 —-

Level	NTrials	obs. freq.	monot. freq.
-100.00			0.00
1.00	20	0.00	0.00
2.00	20	0.10	0.10
3.00	20	0.25	0.25
4.00	20	0.40	0.40
5.00	20	0.80	0.80
100.00			1.00
	100		

5 Running PMETRIC

PMETRIC is invoked from the command line, and the parameters specified on the command line indicate the names of the input and output files. You may either of two possible forms of syntax for the command line, as described in the next two subsections. The former is simpler; the latter, more flexible.

5.1 Invoking PMETRIC Without Switches

One syntax is to invoke PMETRIC with a statement including zero, one, or two parameters, as shown here:

```
C> pmetric
C> pmetric foo
C> pmetric foo bar
```

If no parameters are specified, PMETRIC reads the data from the file “`PMETRIC.DAT`”, reads control parameters from the file “`PMETRIC.RSP`”, and writes its output to the file “`PMETRIC.OUT`”.

If one parameter (“foo”) is specified, PMETRIC reads the data from the file “`foo.DAT`”, reads control parameters from the file “`foo.RSP`”, and writes its output to the file “`foo.OUT`”.

For advanced users only: if two parameters (“foo” and “bar”) are specified, PMETRIC reads the data from the file “`bar.DAT`”, reads control parameters from the file “`foo.RSP`” and it skips to the location “bar” in the control file (as described in section 7), and writes its output to the file “`bar.OUT`”. Note that this option can only be used if the control file “`foo.RSP`” contains a line with just “bar” in it to serve as the location to which pmetric should skip. For an example of this, see the description of “GOTO label” in section 13.8.

5.2 Invoking PMETRIC With Switches

The other syntax is to invoke PMETRIC with a statement including explicit switches to specify the names of input, control, and output files, and the skip location. As an example,

```
C> pmetric -i myinput -c myctrl -o myout -s myskip
```

This specifies that the input is to be read from file “`myinput.dat`”; the control parameters are to be read from “`myctrl.rsp`”; the output is to be written to files “`myout.out`”, “`myout.rpt`”, etc.; and the program is to skip to the location tty myskip in the control file before it starts to read parameters. (Note the default extensions added to all file names!) The skip parameter is optional (no skip is the default).

6 PMETRIC Output

I will explain the output using the file “`exampl1.out`” as an example.

Table 2: Second Part of Output for Data in File “`Exampl1.Dat`”

(std. errors based on 200 bootstrap samples)

**** Report of Analysis Based on Normal(4.036,1.37)

Measures of location					
Mean	: 4.04	(0.22)	Median	: 4.04	(0.22)
Measures of dispersion					
SD	: 1.37	(0.25)	DL	: 0.92	(0.17)
Measures of skewness					
E[(x-mean)^3]/SD^3	: 0.00	(0.00)	(q3-2*q2+q1)/(q3-q1)	: -0.00	(0.00)
E[(x-mean)^3]$^{(1/3)}$	: 0.00	(0.00)	(z9-2*z5+z1)/(z9-z1)	: -0.00	(0.00)
			(Mean-Median)/SD	0.00	(0.00)
Measures of kurtosis					
E[(x-mean)^4]/SD^4	: 3.00	(0.00)	DL/(z9-z1)	: 0.26	(0.00)
E[(x-mean)^4]$^{(1/4)}$	: 1.80	(0.33)			
Ln(Likelihood)	: -42.08	(4.96)	Chi-square	: 1.43	(3.04)
Chi-square df	3.00		Pr(Chi-square)	: 0.70	(0.30)

Note. Parametric estimates are shown on the left side of the table and nonparametric estimates are shown on the right (except for the measures of fit, which are both parametric). Associated with each estimate is a standard error computed using 100 bootstrap samples. SD and DL are the dispersion parameters standard deviation and difference limen, respectively. q_i is the ith quartile of the distribution, and z_i is the ith decile.

The first section of the output looks like what is shown in Table 1. This table simply summarizes the data table for the first data set (“Case 1”) in the input data file. The “Level” column displays the C_i values, the next column displays the number of trials tested at that C_i, the “obs. freq.” column reports the observed frequency G_i/N_i, and the “monot. freq.” column displays monotonized frequencies used in conjunction with the Spearman-Kärber method.

The next section of output looks like what is shown in Table 2. One line simply reports the number of bootstrap samples generated to compute bootstrap standard errors. 200 is the default. The next line shows the distribution estimated by the maximum-likelihood procedure. With these data, this maximum occurs with $\mu = 4.036$ and $\sigma = 1.37$.

The next block of lines in Table 2 shows the estimated mean and median. These are simply the mean and median of the distribution resulting from the maximum likelihood estimation. With the normal underlying distribution, these values are necessarily identical, because the normal mean always equals its median. Following each value, in parentheses, is its standard error as estimated by bootstrapping.

The next block of lines in the table shows analogous values for the standard deviation (SD) and difference limen (DL). The DL is the term used in psychophysical research for the difference between the 25th and 75th percentile points of the distribution.

The next two blocks of lines show analogous values for measures of skewness and kurtosis computed as shown in the diagramatic formulas. In these formulas, q1, q2, and q3 are the 25th, 50th, and 75th percentile values, respectively, and z1, z5, and z9 are the 10th, 50th, and 90th percentile values. Given the assumption of an underlying normal distribution, the estimated skewness and kurtosis values are not informative in this analysis. Skewness is always zero for any normal distribution, and kurtosis is always three, so probit analysis actually provides no way to estimate these values. One of the advantages of the Spearman-Kärber method is that it can estimate skewness and kurtosis.

The next block shows the negative of the natural log of the likelihood function, -Ln(L), computed with the best maximum-likelihood estimates, and the value of the chi-square computed with the minimum chi-square estimates. The chi-square goodness of fit statistic is computed using the formula given in Guilford (1954, p. 134, formula 6.16), and this chi-square has $(k - f)$ degrees of freedom, where k is the number of C_i values for which data have been obtained and f is the number of free parameters associated with the distribution being fit.[3] With probit analysis, for example, $f = 2$ because the normal μ and σ are the free parameters to be estimated. Below the chi-square value is the probability or significance level of this value; if this significance is less than .05, then the model can be rejected

---

[3] The program does *not* automatically group C_i values to ensure expected frequencies of at least 5, so the chi-square approximation may be poor if there are relatively few observations per C_i value.

as an adequate approximation of the data. The chi-square value is omitted from the report for the Spearman-Kärber analysis, because it is not meaningful in this case.

That is the end of the output from the standard probit analysis. The remaining output in the file shows analogous values obtained by the Spearman-Kärber method. Because this is a nonparametric method, estimates are more independent of one another (e.g., mean and median can be different), and the estimates of skewness and kurtosis are meaningful in addition to the estimates of location and dispersion.

One output value is defined differently for the Spearman-Kärber method than for the probit methods—namely, the chi-square value. There is no way to compute a meaningful chi-square value for the Spearman-Kärber method. Instead, the “chi-square” value is really an uncorrected raw mean. This is identical to the regular Spearman-Kärber mean for the yes/no task, but it can be different for mAFC tasks. With the latter tasks, it is possible for an observed probability to be less than chance due to binomial variability. For computation of the “regular” Spearman-Kärber mean, observed probabilities less than chance are adjusted to the chance value. For the “uncorrected raw mean” reported in the chi-square position, the mean is computed using the observed probabilities before adjustment, using the equation given by Ulrich and Miller (2003). In fact, the simulations conducted by Ulrich and Miller indicated that the best estimate of the mean for an mAFC task is actually this “uncorrected raw mean” rather than the corrected one, and we recommend its use. The corrected mean is rather biased.

7 PMETRIC Analysis Options

Numerous aspects of PMETRIC’s behavior can be controlled by the user via a set of analysis options. These are listed below, approximately in order from most-frequently to least-frequently used. In addition to the options listed in this section, further options common to PMETRIC and PMETGEN are listed in section 13.

All analysis options are specified by including one or more lines in the RSP file. An option is specified by typing its name as the first word on a line of the input RSP file (preceding blank space on the line is ignored). If the option requires further information, that information is typed as successive words on the same line (i.e., separated by one or more blank spaces) or, in a few cases to be described, on successive lines. Within an RSP file, the asterisk character (*) is used to indicate comment material: anything following an asterisk on the same line is ignored.

7.1 NProbit

As illustrated in the file “`exampl1.RSP`”, this option is followed by a number (N_d) to indicate the number of different underlying distributions to use in the probit-type analysis. It must then be followed by N_d individual lines, each indicating the distribution name and starting parameter values for one of these distributions. For example, the following lines might appear in an RSP file:

```
nprobit  4       * Use 4 different distributions.  This part is just a comment.
   normal(0,1)   * Try probit with normal starting at mu=0 and sd=1
   gamma(5,.1)   * Also try with gamma starting at shape=5 and rate=.1
   uniform(-1,5) * Also try with uniform distribution with bounds -1, 5.
   logistic(0,1) * Also try logistic distribution with mean 0 and beta 1.
* Note that four distributions have been listed, corresponding to "nprobit 4".
```

The complete list of probability distributions that can be specified for probit analysis within PMETRIC, along with their parameters, can be found in section 14. These distributions can also be used for generating simulated psychometric function data by PMETGEN. The options include not only dozens of standard distributions but also various transformations of these distributions (e.g., linear, log, power) and other distributions formed by taking convolutions, mixtures, order statistics, and so on, as derived from CUPID (Miller, 1998). In fact, PMETRIC is itself merely an extension of CUPID for this particular type of data analysis. Due to memory limitations, the discrete distributions known to CUPID have been omitted from the DOS version of PMETRIC, but this is unlikely to be of much consequence because these distributions are rarely suitable as models for quantal data.

If it is not specified, NProbit is assumed to be 1 and the assumed distribution is normal(0,1). NProbit can be set to 0 if you don’t want any probit analyses but only want the values from the Spearman-Kärber method.

7.2 NBootSamples

This option is followed by an integer to specify the number of bootstrap samples to be used in computing bootstrap standard errors. The default is 200. For example:

```
NBootSamples 30000 * Use lots of bootstrap samples!
```

7.3 TotalN

If you would like to specify the data with values of N_i instead of $N_i - C_i$ in the input data file, then just include the line `TotalN` in the control RSP file.

7.4 UseLog

In some research, it is common practice to log-transform the stimulus values prior to the analysis to eliminate positive skew. If you would like PMETRIC to transform the stimulus values when they are read in, you can request this with the command `UseLog`. If this option is used, all of PMETRIC's output is in the scale of the logged stimulus values (natural log) rather than the original ones. In particular, you should be careful when specifying the starting parameters for probit analysis to give parameters than are meaningful in terms of the logged stimulus values, not the original ones.

7.5 FixedN

If the value of N_i is the same for all stimulus levels and cases, this can be specified with an option like:

```
FixedN 100 * 100 trials at each stimulus level
```

If this option is selected, the third line (i.e., N_i or $N_i - C_i$) of each case must be omitted from the input data file.

7.6 NoMomentMatch

By default, when PMETRIC carries out a probit-type analysis, it starts the parameter search process by adjusting the distributional parameters to produce a mean and variance equal to the mean and variance *of the stimulus values* (i.e., ignoring the observed frequencies). These parameters generally provide a pretty good initial guess from which PMETRIC's search routines can converge to the optimal solution. You may omit this moment-matching step, however, with the `NoMomentMatch` option. In that case, the parameter search always starts from the default parameters given in the statement of the probit distribution. For example, if you specified a probit option of Normal(0,1), the parameter search would always start with a mean of 0 and a standard deviation of 1. If the stimulus values were far from this region—e.g., in the range of 200–400—then the parameter search routines would probably not converge on the optimal solution.

7.7 NoComputeSpear

This option tells PMETRIC to omit the Spearman-Kärber analysis. There is also an option `ComputeSpear` saying to perform the analysis, but it is not needed because this analysis is performed by default.

7.8 InFile

Use this option to set the name of the input data file. For example:

```
InFile MyFile.Dat * Data will be read from MyFile.Dat
```

7.9 Verbose

If you specify the `Verbose` option, PMETRIC will write more information to the screen so that you can keep track of its progress. This is particularly useful if it bombs and you want to see where.

7.10 MissingValue

When a sample is pathological and it is impossible to compute probit estimates, a missing value is written out in place of such requested estimates. By default, this value is -9999. You may change it to any real number via a command like `MissingValue 999.99`.

7.11 DoubleSearch

Maximum-likelihood and minimum-χ^2 estimates are obtained with the simplex search algorithm (Rosenbrock, 1960). Like all numerical search algorithms, it can get trapped in locally- but not globally-optimal solutions under some circumstances. The likelihood of such trapping can be reduced by running the algorithm twice, starting it for the second time at the finishing point of the first time. If the `DoubleSearch` option is specified, then this strategy is employed for all parameter searching. Naturally, if this option is selected then the program takes almost twice as long to run.

7.12 ExpansionFactor

During bootstrapping, random generation of new samples may occasionally generate what I call a "pathological" sample. In such a sample, G_i tends to decrease as C_i increases, rather than showing the expected increase. Although rare, this can happen during the bootstrap resampling process because of binomial variability (naturally it happens more often when the number of trials is smaller). When it does happen, no model has high likelihood because every model predicts that the G_is must increase. As a consequence of low likelihoods and numerical problems, the parameter search routine usually wanders off into some really strange area of the parameter space when estimating parameters for a pathological sample. The resulting extremely unusual parameters estimated for that bootstrap sample can contaminating the bootstrap standard error.

The purpose of the "expansion factor" option is to define a criterion to identify these pathological cases so that they can be discarded and they will not contaminate the estimated bootstrap SE. (In fact, pathological bootstrap samples are replaced with nonpathological ones to keep constant the number of usable bootstrap samples).

Within PMETRIC, the expansion factor, F, is a numerical value used as follows to identify pathological samples. A pathological bootstrap sample is defined as one for which any of the following conditions is met:

1. estimated mean is less than $(C_1 + C_k)/2 - F \times (C_k - C_1)/2$
2. estimated mean is greater than $(C_1 + C_k)/2 + F \times (C_k - C_1)/2$
3. estimated standard deviation is greater than $F \times (C_k - C_1)$

for *any* of the estimation procedures (i.e., Spearman-Kärber or probit). With $F = 1$, these conditions would reduce to

1. estimated mean is less than C_1
2. estimated mean is greater than C_k
3. estimated standard deviation is greater than $(C_k - C_1)$

If $F \leq 0$, PMETRIC does no checking for pathological samples.

By default, $F = 100$ so that only extremely pathological samples will be replaced. Use the command ExpansionFactor to change the value of F. For example:

```
ExpansionFactor 0 * Do not check for pathological samples.
```

7.13 Output Format Control

There are three available output formats. One looks like the output in the file "`exampl1.out`". This is called the "OUT" format. This format is written by default.

A second available output format is a table with one row per case and one column for each of the values computed from each case. This is the "MTB" format, and it may be useful if you want to process a large number of cases and then import the results into another statistics package for further analysis. This format is not written by default.

A third available output format is similar to the MTB format except for the header lines, which are appropriate for my MrF ANOVA program. This is the "RPT" format, and it will be useful if you want to analyze PMETRIC output with MrF. This format is not written by default.

The OUT, MTB, and RPT options are independent, so you can get any combination of them by specifying options appropriately. To change the default settings for the output format, simply include in the RSP file any combinations of the commands `WriteMTB`, `NoWriteMTB`, `WriteRPT`, `NoWriteRPT`, `WriteOUT`, and `NoWriteOUT`. The default settings are WriteOut, NoWriteMTB, and NoWriteRPT. For example, if you include just the `WriteMTB` command, you will get both the OUT format and the MTB format.

In addition to choosing the type of output file, there are a number of other options concerning the control of output, as described below. Some of these options are relevant only for certain types of output files, as indicated.

NCases The number of cases is written at the beginning of the MTB file, so this number must be specified in the RSP file. For example,

```
NCases 24 * There are 24 input data sets.
```

WriteDVs To avoid cluttering up your output file, you may want to write out only a subset of the summary measures computed by the program. In that case, you can use this option to select which ones you want, as follows. Following WriteDVs on the same line, include the number of summary measures (K_s) that you would like to have written. Then, the following K_s lines should include one number per line, where each number is the sequential number, relative to the full set written when this option is not specified, of one of the summary measures you would like to have written out. For example, the following requests that only the median and DL be written (note that these are normally the 11th and 12th summary measures written out for each distribution):

```
WriteDVs 2
   11   * The median is the 11th summary measure in the standard list.
   12   * The DL is the 12th summary measure in the standard list.
```

MomentsWanted Computation of higher moments sometimes involves rather slow numerical integration. If you are only interested in the lower moments, then, you might want to omit computation of higher moments to speed up the program. Unfortunately, PMETRIC is not smart enough to figure out which moments to compute automatically from the WriteDVs option described above, so you must explicitly tell it which moments to compute with a command like:

```
MomentsWanted 2  * Compute moments only up to the second.
```

The default is MomentsWanted 4, but you can speed up the program by reducing this to 3, 2, or 1.

WritePCTs Instead of having the program write out summary DVs, you may want to have it write out the percentiles of the psychometric function. This output is requested with the WritePCTs option. The format is:

```
WritePCTs
    5   *  Start writing with the 5th percentile
   95   *  ... and continue up to the 95th percentile
   10   *  ... in steps of 10%
```

Note that cumulative probabilities are automatically monotonized when the WritePCTs option is selected, just as they are for the Spearman-Kärber method, to ensure that each of the desired percentiles occurs only once within the distribution. In addition, the extreme stimulus values C_0 and C_{k+1} will be used to estimate (by linear interpolation) percentile values that are outside the range of observed cumulative probabilities associated with the C_1 to C_k values. Note also that when the WritePCTs option is selected, the program assumes that you do not want any DVs of the type selected with the WriteDVs option. If you do want both, you must include the WriteDVs option after WritePCTs is specified.

NGroups The Mrf program needs to know how many groups of subjects are included in the input file, so you are allowed to specify the appropriate value. For example,

```
NGroups 2 * There are 2 groups of Ss.
```

NCasesPerS The Mrf program also needs to know how many different psychometric functions have been analyzed per subject (i.e., how many different conditions are tested within-Ss). For example,

```
NCasesPerS 3 * Each S provided pmetric fns for 3 conditions.
```

This variable also influences the numbering of files if WriteCurves is specified (see below).

Tab Delim By default, the columns in the MTB file are separated by spaces. You can have them separated by tabs with this option or by commas with the next one.

Comma Delim

7.14 Output of Observed and Monotonized Functions

It may also be useful to write the observed and monotonized functions to files for importing into graphing programs, and this can be accomplished with the `WriteCurves` option. If this option is selected, a separate *.col file is written for each case that is analyzed, containing three columns of numbers: the stimulus levels, the observed relative frequency at that level, and the monotonized relative frequency at that level (much like the tabular summary written at the beginning of each case in the OUT file).

The different cases are distinguished by being written to different output files. It is assumed that the first NCasesPerS cases come from subject one, and these will be written to files called `S001C001.col`, `S001C002.col`, `S001C003.col`, ... Then next NCasesPerS cases are are assumed to come from subject 2 and are written to files called `S002C001.col`, ... And so on.

7.15 Maximum Likelihood versus Minimum Chi-Square Estimation

By default, PMETRIC computes the maximum-likelihood estimates of the distributional parameters and then reports the properties of the distribution (e.g., mean, median, etc) with these parameter values. By including the option `ChiSquare`, however, the user instructs PMETRIC to compute minimum chi-square estimates instead.

Note that both maximum-likelihood and minimum chi-square estimates are actually always computed, so that both measures of fit can be reported. Thus, specifying the `ChiSquare` option simply changes whether the estimated mean, median, and so on are computed from the distribution with its maximum-likelihood or minimum chi-square values.

For completeness, the option `Likelihood` may also be specified, although this is the default option.

7.16 MaxNStimLevels

By default, PMETRIC allocates memory for a maximum of 200 stimulus levels or doses (the k in the technical description given in section 2). If that is not enough, this value can be increased by specifying this option in the RSP file. For example:

```
MaxNStimLevels 1000 * Up to 1000 different doses
```

7.17 ParmCodes

In some situations, it may be desirable to fix a parameter of the model underlying probit analysis. For example, you might know on a priori grounds that the mean of the underlying normal distribution was 0 and you might want to allow only the standard deviation to vary during maximum likelihood estimation. PMETRIC provides this capability by allowing you to specify whether each parameter of a distribution is Fixed, Real, or an Integer.

To constrain one or more parameters of a distribution, you enter an optional extra string of letters (F, R, and I) for each distribution listed under the NProbits option. For example:

```
nprobit  2           * Use 2 different distributions.
   normal(0,1)   FR  * Fix the mean at zero and let sigma vary as a real.
   gamma(5,.1)   IF  * Let the number-of-exponentials parameter vary as an
                     *  integer and hold the exponential rate fixed.
```

In this example, both strings have two letters because each distribution has two parameters. The string of letters under discussion here is called the “ParmCodes” string, and more information on it can be found in the CUPID documentation.

7.18 MinStepSize

This option is used to control the minimum step size used by the parameter search algorithm in probit analysis. For example:

```
MinStepSize 0.00001
```

The default is 1.0e-8.

8 Introduction to PMETGEN

PMETGEN can be used to generate simulated data of the sort analyzed by PMETRIC. It has two main uses:

1. Researchers in the planning phase of an experiment may wish to run computer simulations of a proposed design to estimate its standard errors and statistical power.
2. Researchers interested in comparing different statistical procedures for the analysis of such data may wish to generate many simulated data sets in order to compare the performance of the different procedures.

9 PMETGEN Input Control File

Before using PMETGEN, you must prepare a control file describing the exact experimental situation to be simulated. This file should be given the extension “.GEN”, and the files “`GENEX*.GEN`” show examples. The input file must be a plain text (ASCII) file.

The file “`GENEX1.GEN`” shows a fairly simple control file. The lines starting with asterisks are explanatory comments, as is all the text following an asterisk on a line. Here are the operative lines of the file:

```
Distribution Normal(0.5,0.25)
NStimLevels      5
NTrialsPerLevel 60
NCases  300
```

The first line tells the program that the true underlying distribution has the shape of a cumulative normal distribution with mean 0.5 and standard deviation 0.25. Many other distributions can be used, such as the gamma, lognormal, exponential, uniform, and so on. For a complete list of the possible distributions, the user should consult the documentation of the CUPID program.

The second line says that five stimulus levels should be used. (These stimulus levels will be placed at default locations; further information on the setting of stimulus levels is given in section 12.) The third line says that the experiment should include 60 trials at each stimulus level, and the fourth says that the program should generate 300 simulated data sets.

10 Running PMETGEN

PMETGEN is invoked from the command line with a statement including one or two parameters, as shown here:

```
C> pmetgen foo
C> pmetgen foo bar
```

If one parameter (“foo”) is specified, PMETGEN reads control parameters from the file “`foo.GEN`”, and writes its output to the file “`foo.DAT`”.

For advanced users only: if two parameters (“foo” and “bar”) are specified on the command line when invoking PMETGEN, the program reads control parameters from the file “`foo.GEN`”. In addition, however, it skips to the location “bar” in the control file (as described in section 12). The purpose of this is to allow a single GEN file to control data generation for a number of slightly different examples. The different options are listed first in the GEN file, and at the end of each option you include a statement like “GOTO COMMON”. Then, you list the common parameters after the label common.

11 PMETGEN Output

The main output of PMETGEN is a data file that can be read and analyzed by PMETRIC (just as if the data came from actual experiments rather than simulated ones). It contains one data set per simulated experiment (but in some types of research it may make sense to regard the data sets as coming from different individual subjects within a single experiment).

A secondary output file has the same name but the extension .PRM. This file contains a list of the stimulus values tested and some self-explanatory summaries of the true parameters of the underlying distribution.

12 PMETGEN Options

Individual options for controlling the behavior of PMETGEN are listed below, approximately in order from most frequently used to least frequently used. All options are specified by including one or more lines in the control file, using the same formats as those of the PMETRIC control files. The control files for PMETGEN have the extension GEN (RSP was not used again to avoid conflict with the RSP files used to analyze the generated data). In addition to the options listed in this section, further options common to PMETRIC and PMETGEN are listed in section 13.

12.1 Distribution

As illustrated in the file “`GENEX1.GEN`”, this option is followed by the name of the underlying probability distribution to be used in generating the data. The complete set of probability distributions that can be specified for generating data is the same as that which can be used for probit analysis within PMETRIC. For example, any of the following would be a legal specifications of the true underlying function: `Logistic(0,1)`, `Uniform(0,1)`, `Gamma(5,0.01)`, `Exponential(1.2)`.

12.2 NStimLevels, NTrialsPerLevel, and NCases

These options are used to set the desired number of stimulus levels, number of experimental trials per stimulus level, and number of simulated data sets to generate, respectively.

12.3 Setting the Stimulus Levels

Once the number of stimulus levels is determined, it is necessary to indicate what stimulus levels are to be used. There are three options for doing this, which will be referred to as three “versions.” The default is version B.

Version A With this version, PMETGEN computes stimulus values that are equally spaced in the *probability* domain. Here is an example:

```
Distribution Normal(0.5,0.25)
NStimLevels 5
Version A
StepsToExtreme 1
```

With this set of commands, the stimulus values are set as follows:

1. Compute 5 equally-spaced probability values, P_i—i.e., 0.1, 0.3, 0.5, 0.7, and 0.9.
2. Compute S_i such that each S_i has the desired cumulative probability P_i within the indicated distribution.
3. Using the indicated value of StepsToExtreme, β, compute the lower bound $S_0 = S_1 - \beta \times (S_2 - S_1)$ and the upper bound $S_6 = S_5 + \beta \times (S_5 - S_4)$. The final set of stimulus values and bounds is

Stimulus	Value	Probability
0	-0.010	0.000
1	0.180	0.100
2	0.369	0.300
3	0.500	0.500
4	0.631	0.700
5	0.820	0.900
6	1.010	1.000

Version B With this version, the user specifies the percentiles of the upper and lower stimulus values, and PMETGEN computes intermediate values that are equally spaced in the *stimulus* domain. Here is an example:

```
Distribution Normal(0.5,0.25)
NStimLevels 5
Version B
HiPctile 0.95
LowPctile 0.05
StepsToExtreme 1
```

With this set of commands, the stimulus values are set as follows:

1. Compute the lowest stimulus value S_1 to be the value at the 5th percentile of the indicated distribution and the highest stimulus value S_5 to be the value at the 95th percentile. This yields $S_1 = 0.089$ and $S_5 = 0.911$.
2. Compute a step size, Δ, such that this range is covered with five stimuli: $\Delta = \frac{0.911-0.089}{5-1} = 0.206$.
3. Compute $S_i = S_1 + (i - 1) \times \Delta$ for $i = 2, \ldots, 4$.
4. Using the indicated value of StepsToExtreme, β, compute the lower bound $S_0 = S_1 - \beta \times \Delta$ and the upper bound $S_6 = S_5 + \beta \times \Delta$. The final set of stimulus values and bounds is

Stimulus	Value	Probability
0	-0.117	0.000
1	0.089	0.050
2	0.294	0.205
3	0.500	0.500
4	0.706	0.795
5	0.911	0.950
6	1.117	1.000

Version C With this version, the user specifies each stimulus value and both lower and upper bounds individually. For example:

```
Distribution Normal(0.5,0.25)
NStimLevels 5
Version C
 -20  -1 -0.5  0 0.5 1  20
```

This indicates that the five stimulus values are at exactly the specified values of -1, -0.5, 0, 0.5, and 1. Surrounding these values are the lower and upper bounds of -20 and 20. Thus, the final set of stimulus values and bounds is

Stimulus	Value	Probability
0	-20.0	0.0
1	-1.0	0.0
2	-0.5	0.00003
3	0.0	0.0223
4	0.5	0.5
5	1.0	0.977
6	20.0	1.0

12.4 Output Files of Parameter Values

PMETGEN can write out the true parameter values corresponding to the simulated distribution (e.g., true mean) in either of two formats. One format is written to a file with the extension .PRM, and this one is similar to the format of the parameters estimated by PMETRIC. The other format is written to a file with the extension .PR2; these are the same values, but in a format that is more easily machine-readable. The user can control whether either, neither, or both of these files is written by specifying any combination of the following parameters in the control file:

```
WantPRM    *  Do write the PRM file
WantPR2    *  Do write the PR2 file
NoPRM      *  Do not write the PRM file
NoPR2      *  Do not write the PR2 file
```

By default, the PRM file is written and the PR2 file is not.

12.5 NoRandomTrials

This option tells PMETGEN to generate data in accord with the exact probabilities associated with each stimulus value. In this case, the numbers of responses at each stimulus value are not generated randomly but are set to match the expected frequencies as closely as possible (given the constraint to generate whole numbers). This option is useful in evaluating the pure lack of fit of a model (e.g., how badly does a normal-based probit analysis fit data generated by a gamma underlying distribution) without contamination by random variation.

13 Other Control Options Common to PMETRIC and PMETGEN

A number of program-control options are common to both PMETRIC and PMETGEN.

13.1 CommonCs

Include this option to indicate that the C_i values are the same for all cases and *are* (with PMETRIC) or *should be* (with PMETGEN) listed only on the first line of the data file. For example:

```
CommonCs * All data sets use same stimulus values / doses.
```

13.2 Forced-Choice Tasks

By default, the analysis is carried out (PMETRIC) or the data are generated (PMETGEN) assuming that responses come from a Yes/No task. The programs can also analyze and generate data from an m-alternative forced choice task, however. To indicate that such a task is to be analyzed or simulated, include a parameter like

```
mAFC 3 * data from 3-alternative forced choice task.
```

With 2. . . 9 alternatives, you may abbreviate this as 2AFC, 3AFC, 4AFC, . . . , 9AFC.

Note for PMETRIC: A complication that can arise in the analysis of m-AFC tasks is that observed performance can be worse than chance at a certain stimulus level if performance is actually near chance at that level and if the binomial variability produces a less-than-expected number of responses. When estimated probabilities are corrected for guessing, then, it is possible for an estimated probability to be negative. PMETRIC checks for this complication and corrects any negative estimated probabilities to values of zero.

13.3 OutFile

Use this option to set the name of the output file. For example:

```
OutFile MyFile * Output will be written to MyFile.*
```

The output files of PMETRIC get the extensions OUT and MTB, and the output files of PMETGEN get the extensions DAT, PRM, and PR2. By default, the output file name is the same as the input file name, with the appropriate changes of extension.

13.4 FieldWidth and DecPlace

These two separate options are used to control the format in which numerical values are written out. For example:

```
FieldWidth 14 * Allow 14 spaces for each number.
```

```
Decplace 8 * Give 8 decimal places for each number.
```

The defaults are 8 and 2, respectively. In PMETGEN, this option applies only to the stimulus levels, because the actual generated data are integers. In PMETRIC, it applies to all computed estimates.

13.5 Controlling the Seed of the Random Number Generator

Both programs use a random number generator—PMETGEN for generating simulated data and PMETRIC for doing bootstrap iterations. Three options are provided for controlling the seed of the random number generator:

```
SEED SAVE START filename
```

saves the random number seed into the specified file. Saving is done at the beginning of the program, and it can be retrieved in another program run so that the subsequent run can be carried out with the identical sequence of random numbers, if desired.

```
SEED SAVE END filename
```

also saves the random number seed into the specified file. With this option, however, saving is done at the end of the program. If the seed is then retrieved in another program run, the subsequent run will be carried out starting from the ending point of the previous sequence of random numbers, eliminating the possibility of overlap in the random sequences.

```
SEED START filename
```

causes the program to start by reading the seed from the indicated file (which should previously have been written with the SEED SAVE START or SEED SAVE END option already described).

13.6 Controlling the Accuracy of Numerical Integration and Inversion

In many cases where explicit formulas have not been programmed in, PMETRIC computes values by numerical integration. The speed and accuracy of this iterative procedure are controlled by a parameter called `IntegralPrecision`, which may be set with a command like

```
IntegralPrecision 0.00001
```

The default is 1.0e-7.

Simiarly, when PMETRIC must find the inverse of a CDF for a distribution with no explicit InverseCDF function built in, it uses a numerical search algorithm to find the desired X value corresponding to the specified P. You may control the accuracy required for the search to stop with the commands

```
InversePrecisionP 0.00001
```

and

```
InversePrecisionX 0.00001
```

(the defaults are 1.0e-7). These control the precision with which the desired P value and the value of the random variable X must be known when finding the inverse of the CDF.

13.7 Comment Character

The comment character (asterisk, by default), can be changed to any ASCII character. For example:

```
COMMENT ! * Change takes place starting with next line.
```

would change the comment character to an exclamation point.

13.8 Flow of Control

PMETRIC and PMETGEN normal read their control parameters line by line through the RSP and GEN files. Two commands will alter this behavior, and these are useful in preparing a single control file to carry out several different analyses or simulations in different runs.

END If this command is encountered, processing of the input control file stops.

GOTO label If this command is encountered, processing of the input control file skips to a line on which the indicated label appears, and continues on from the next line following that.

As an example, one might prepare the following file “`TwoDists.RSP`” to generate simulated data from both a normal and a uniform distribution:

```
normal
Distribution Normal(0,1)
goto common
uniform
Distribution Uniform(-1,1)
goto common

common
NStimLevels 5
NTrialsPerLevel 60
NCases 300
```

PMETGEN could be invoked with the optional second parameter `normal` to use the normal distribution and `uniform` to use the uniform distribution, with all other data generation options in common.

14 Available Distributions

14.1 Continuous Distributions

Here are the primitive continuous distributions that have been at least partially implemented so far:

Beta(A, B) The Beta distribution is defined over the interval from zero to one, and its shape is determined by its two parameters A and B. Its PDF is

$$f(x) = \frac{1}{\beta(A,B)} x^{A-1}(1-x)^{B-1}, \quad 0 < x < 1$$

The mean is $A/(A+B)$, and the variance is $AB(A+B)^{-2}(A+B+1)^{-1}$.

Cauchy(L, S) This distribution is defined in terms of location and scale parameters L and $S > 0$, respectively. Its PDF is

$$f(x) = \frac{1}{\pi S \left[1 + \left\{\frac{x-L}{S}\right\}^2\right]}$$

ChiSquare(df) This is a generalization of the distribution of the sum of *df* independent squared standard normals. Its parameter is *df* — a positive real number.

Chi(df) This is the distribution of the positive square root of a ChiSquare random variable. Its parameter is df — a positive real number.

Cosine For $0 \leq x \leq \Pi/2$, $f(x) = \cos(x)$ and $F(x) = \sin(x)$.

ExGaussian(μ, σ, λ) This is the distribution of the sum of independent Normal and Exponential random variables. Its parameters are the μ and σ of the Normal, and the rate λ of the Exponential.

ExGaussRat(μ, σ, r) This is just a reparametrization of the ExGaussian. Its parameters are the μ and σ of the Normal, and the ratio, r, of the mean of the exponential to the sigma of the normal.

Exponential(λ) This distribution is well-known. By default, the parameter is the rate λ; the mean is $1/\lambda$.

ExpSum($r1, r2$) This is the sum (convolution) of two exponentials with *different* rates. The two parameters are the two rates, which must be different enough to avoid numerical errors. For the convolution of exponentials with the same rates, of course, you should use the Gamma.

ExpSumT($r1, r2$,Cutoff) This is the sum (convolution) of two exponentials with *different* rates truncated at a given cutoff value. The first two parameters are the two rates, which must be different enough to avoid numerical errors; the third parameter is the upper truncation point. For the convolution of exponentials with the same rates, of course, you should use the Gamma.

ExpoNo(μ, σ) I just invented this as an ad-hoc solution for a problem I was working on one time, so I don't know whether it will ever be useful again. And I certainly don't know whether I gave it a reasonable name. In any case, it is a transformation of a normal random variable X. Specifically, it is the distribution of

$$Y = \frac{e^X}{1 + e^X}$$

where X has a normal distribution with mean μ and standard deviation σ. The two parameters of this distribution are the μ and σ of the underlying normal X.

ExtremeVal(α, β) Extreme-value Type I distribution (a.k.a. Fisher-Tippett distribution, Gumbel distribution, sometimes also called the double exponential distribution, to be confused with the Laplace distribution), with parameters α and $\beta > 0$. The CDF is

$$F(x) = \exp\left[-e^{-(x-\alpha)/\beta}\right]$$

ExWald(μ, σ, a, λ) This is the distribution of the sum of two independent random variables: one from a three-parameter Wald distribution with parameters (μ, σ, a); and one from an exponential distribution with rate λ.

F(dfNumer,dfDenom) This is Fisher's distribution of the ratio of two independent normed Chi-square distributions, as commonly used in linear models (e.g., analysis of variance). The two integer parameters are the degrees of freedom of the numerator and denominator, respectively.

Gamma(N, λ) This is the distribution of the sum of N exponentials, each with rate λ. In this distribution, N must be a positive integer. In the RNGamma distribution (see below), N is any positive real.

Geary(SampleSize) The Geary statistic arises in testing to see whether a set of observations come from a normal distribution (D'Agostino, 1970).

GenErr(Mu,Scale,Shape) This is the general error distribution (e.g., Evans, Hasting, & Peacock, 1993, p. 57) with PDF

$$f(x) = \frac{\exp\left[-|x - \text{Mu}|^{\text{Shape}}/\left(2 \cdot \text{Scale}\right)\right]}{\text{Scale}^{1/2} \cdot 2^{(1+1/\text{Shape})} \cdot \Gamma(1 + 1/\text{Shape})}$$

This version uses the shape parameter denoted α by Evans et al. Note that the Laplace and normal distributions are special cases of this distribution with Shape equal 1 and 2, respectively.

HypTan(Scale) This is the Hyperbolic Tangent distribution, whose PDF and CDF are[4]

$$\begin{aligned} f(x) &= \frac{4 \cdot \beta}{\left[e^{\beta x} + e^{-\beta x}\right]^2} \\ F(x) &= \frac{e^{\beta x} - e^{-\beta x}}{e^{\beta x} + e^{-\beta x}} \end{aligned}$$

where β is the scale parameter. This distribution arises as a model of psychometric functions (e.g., Strasburger, 2001).

Inverse Gaussian See the Wald distribution.

Laplace(L, S) Also known as the double exponential. In terms of location and scale parameters, L and $S > 0$, respectively, the PDF is

$$f(x) = \frac{1}{2S} e^{-|x-L|/S}$$

Logistic(μ, β) This distribution is defined in terms of a location parameter μ and a scale parameter β. The cumulative form of the distribution is

$$F(x) = \frac{1}{1 + e^{\frac{-(x-\mu)}{\beta}}}$$

LogNormal(μ, σ) This is the distribution of X such that $\ln(X)$ is normally distributed. The parameters are the μ and σ of the normal.

Naka-Rushton(Scale) This is the distribution of $X \geq 0$ such that

$$\begin{aligned} f(x) &= \frac{2 \cdot x \cdot \alpha^2}{(1 + (\alpha \cdot x)^2)^2} \\ F(x) &= \frac{(\alpha \cdot x)^2}{1 + (\alpha \cdot x)^2} \end{aligned}$$

where α is the scale parameter.[5] In the actual distribution, moments above the first do not exist; they do exist in PMETRIC's truncated version of the distribution, however.

NoncentralF(dfNumer,dfDenom,Noncentrality) This is the distribution of the ratio of independent noncentral and central chi-squares, with the former in the numerator. It is most often used in the computation of power of the F test. The noncentrality parameter is defined in terms of the dfNumer normal random variables whose sum of squares is yields the chi-square in the numerator. Specifically,

$$\lambda = \sum_{i=1}^{\text{dfNumer}} \Lambda_i^2$$

where Λ_i is the expected value of the ith random variable contributing to this sum of squares.

Normal(μ, σ) I'll bet you know this one already. Parameters are μ and σ, not σ^2.

Pareto1(K,A) This is a Pareto distribution of the first kind, as defined by Johnson, Kotz, and Balakrishnan (1994, vol 1, p 574), with PDF and CDF

$$\begin{aligned} f(x) &= A \cdot K^A \cdot x^{-(A+1)} \\ F(x) &= 1 - \left(\frac{K}{A}\right)^A \end{aligned}$$

where $K > 0$, $A > 0$, and $x > K$.

---

[4] I thank Rolf Ulrich for supplying these.

[5] I thank Rolf Ulrich for supplying the PDF.

Quantal(Threshold) This distribution is related to the Poisson. This is the distribution of $X \geq 0$ such that

$$F(x) = 1 - \sum_{t=0}^{T-1} \frac{x^t}{t!} e^{-x}$$

This distribution arises as a model of psychometric functions in visual psychophysics (e.g., Gescheider, 1997, p. 85). The threshold parameter, T, represents an observer's fixed threshold for the number of quanta of light that must be detected before saying "Yes, I saw the stimulus." Quanta are assumed to be emitted from the stimulus according to a Poisson distribution with parameter x. Then, $F(x)$ is the psychometric function for the probability of saying "Yes" as a function of the mean number of quanta, x, emitted by the stimulus. Note that it makes no real sense to think of x as a random variable in this example, but the probability distribution provides a useful model anyway.

Quick(Scale,Shape) This is the distribution of $X \geq 0$ with PDF and CDF[6]

$$\begin{aligned} f(x) &= \frac{2^{-\left(\frac{x}{\alpha}\right)^{\beta}} \cdot \left(\frac{x}{\alpha}\right)^{\beta} \cdot \beta \cdot \ln(2)}{x} \\ F(x) &= 1 - 2^{-\left(\frac{x}{\alpha}\right)^{\beta}} \end{aligned}$$

where α is the scale parameter and β is the shape parameter. This distribution arises as a model of psychometric functions (e.g., Quick, 1974; Strasburger, 2001).

Rayleigh(σ) If Y_1 and Y_2 are independent normal random variables with mean 0 and standard deviation σ, then $X = \sqrt{Y_1^2 + Y_2^2}$ has a Rayleigh distribution with scale parameter σ. The PDF is

$$f(x) = e^{-x^2/(2\sigma^2)} \frac{x}{\sigma^2}$$

RNGamma(RN, λ) See "Gamma". In this version, the shape parameter RN is a real number rather than an integer.

rPearson(SampleSize) This is the sampling distribution of Pearson's r (correlation coefficient) under the null hypothesis that the true correlation is zero (and assuming the usual bivariate normality). The parameter is *SampleSize*, the number of pairs of observations across which the correlation is computed.

t(df) Student's t-distribution, with parameter *df*.

Triangular(B, T) In this distribution the density function has the shape of an equilateral triangle across some range. The parameters are the bottom (B) and the top of the range (T). The PDF is then:

$$f(x) = \begin{cases} (x - B) \times H_p & \text{if } B \leq x \leq \frac{B+T}{2} \\ (T - x) \times H_p & \text{if } \frac{B+T}{2} \leq x \leq T \end{cases}$$

where H_p is the height of the PDF at its peak, adjusted to so that the total area of the triangle is 1.0.

TriangularG(B, P, T) In this (more general triangular) distribution, the density function has the shape of a not-necessarily-equilateral triangle across some range. The parameters are the bottom of the range (B), the point at which the triangle reaches its maximum (P), and the top of the range (T). The PDF is then:

$$f(x) = \begin{cases} \frac{(x-B) \times H_p}{P-B} & \text{if } B \leq x \leq P \\ \frac{(T-x) \times H_p}{T-P} & \text{if } P \leq x \leq T \end{cases}$$

where H_p is the height of the PDF at its peak, adjusted to so that the total area of the triangle is 1.0.

Uniform(B, T) This is the distribution in which all values are equally likely within some range. The parameters are the bottom and the top of the range, B and T.

---

[6] I thank Rolf Ulrich for supplying these.

UniGap(T) This is an equal-probability mixture of two uniform distributions, one extending from $-T$ to 0 and the other extending from T to $2 \cdot T$. It is "model 4" of Sternberg and Knoll (1973). The median is somewhat arbitrarily defined as $T/2$.

Wald(μ, λ) This distribution arises in problems involving the first passage time with Brownian motion and positive linear drift. As defined by Johnson, Kotz, and Balakrishnan (1994, Volume 1), the standard two-parameter inverse Gaussian has the PDF:

$$f(x) = \left[\frac{\lambda}{2\pi x^3}\right]^{1/2} \exp\left\{-\frac{\lambda}{2\mu^2 x}(x-\mu)^2\right\}$$

where $\mu > 0$, $\lambda > 0$, and $x \geq 0$. μ is the mean and μ^3/λ is the variance.

Wald3(μ, σ, a) This is a three-parameter version of the Wald distribution. Specifically, assume a one-dimensional Wiener diffusion process starting at position 0 at time 0 and drifting with average rate μ and variance σ^2, and consider X to be the first passage time through position a. The PDF of X is

$$f(x) = \frac{a}{\sigma\sqrt{2\pi x^3}} \cdot \exp\left[-\frac{(a-\mu x)^2}{2\sigma^2 x}\right]$$

where all three parameters and x must be positive.

Weibull(Scale,Power,Origin) As defined by Johnson & Kotz (1970, p. 250): "X has a *Weibull distribution* if there are values of the parameters $c(>0)$, $\alpha(>0)$, and ν_0 such that

$$Y = \left[\frac{(X-\nu_0)}{\alpha}\right]^c$$

has the exponential distribution with rate = 1". Here, the parameters c, α, and ν_0 are referred to as the "scale," "power," and "origin" parameters, respectively.

The CDF of the Weibull is therefore

$$F(x) = 1 - \exp(-[(x-\nu_0)/c]^\alpha)$$

Computations are increasingly inaccurate for powers less than about 0.9, however.

14.2 Discrete Distributions

Here are the primitive *discrete* distributions that have been at least partially implemented so far:

Binomial(N, p) The distribution of the number of successes in N Bernoulli trials, with probability p of success on each trial.

Constant(C) This is a degenerate distribution that always takes on the same value. Its parameter is that value. Perhaps surprisingly, it can be convenient to have this distribution available. *Warning:* For technical reasons, the constant distribution does not work well in many of the derived distributions discussed in the next section. Thus, it should be avoided whenever possible. For example, you should always use:

```
LinearTrans(Gamma(2,.01),1,100)
```

rather than the equivalent

```
Convolution(Gamma(2,.01),Constant(100))
```

Geometric(P) The distribution of the trial number of the first success in a sequence of Bernoulli trials, where P is the probability of success on each try.

List(filename) This random variable allows you to define any discrete set of X values, each with its own arbitrary probability. The command

```
List(FileName)
```

tells PMETRIC to read the distribution from the indicated file. The first line in the file contains the number of X values in the distribution. After that, there should be one line for each X value, with the first number on the line being the X value itself and the second number being the probability of that X value. These values need not be sorted, and in fact the same X value can appear on several different lines, in which case the associated probabilities will be summed. (If X's do appear on more than one line, then the first line in the file should actually contain the number of X-containing lines rather than the number of distinct X's.)

Poisson(U) X has a Poisson distribution with parameter U if

$$\Pr(X = x) = \frac{e^{-U} U^x}{x!}, \quad x = 0, 1, 2, \ldots, U > 0$$

The mean and variance both equal U.

UniformInt(Low,High) This is the distribution of equally likely integer values between the two integer parameters, Low and High, inclusive.

14.3 Transformation Distributions

PMETRIC can form a new random variable (Y) by taking a mathematical transformation of an existing one (X). The following table lists the transformations recognized by PMETRIC, illustrating the syntax for each. Also listed are the constraints on the values of X.

Transformation	Example of Syntax	Constraints on Values of X
ArcSin ($Y = \sqrt{(\phi(X/2))}$)	`ArcSinT(Uniform(.5,1))`	
Exponential ($Y = e^X$)	`ExpTrans(Uniform(.5,1))`	X not too far from 0.
Inverse ($Y = 1/X$)	`InverseTrans(Uniform(.5,1))`	X not too close to 0.
Linear ($Y = A \times X + B$)	`LinearTrans(Uniform(.5,1),2,10)`	
Natural Log ($Y = \ln[X]$)	`LnTrans(Uniform(0.5,1))`	$X > 0$
Power ($Y = X^p$)	`PowerTrans(Uniform(.5,1),2)`	$X > 0$

where $\phi(Z)$ is the probability that a standard normal random variable is less than Z.

14.4 Derived Distributions

PMETRIC also knows about various sorts of distributions that can be derived from one or more primitive or "basis" distributions. In most cases, PMETRIC can compute moments, PDF's, CDF's, random numbers, etc, for the derived distribution just as it can for the primitive distributions defined above.

Convolution(RV1,RV2) This is the distribution of a sum of *independent* random variables, RV1 and RV2, where RV1 and RV2 are each legal distributions in their own right. For example,

```
Convolution(Normal(0,1),Uniform(0,1))
```

specifies the convolution of these normal and uniform distributions, and

```
Convolution(Normal(0,1),Uniform(0,1),Gamma(3,0.01))
```

specifies the convolution of the three indicated distributions.

In general, to define a convolution, the user types something of the form:

```
Convolution(BasisDist1(Parms),...,BasisDistK(Parms))
```

where `Parms` stands for the parameters associated with each of the distributions. There are K random variables summed together, and the distributions of these summed variables are simply listed, separated by commas.

PMETRIC is not very smart about convolutions. At this point, it only knows how to compute means, variances, and random numbers in an intelligent way. Everything else is computed using (recursive) numerical integration, which tends to be pretty slow. Also, PMETRIC does not "realize" that some convolutions result in a new distribution about which it already knows (e.g., convolution of two normals is normal). Thus, computations involving these convolutions proceed via numerical integration even though direct computation would be possible.

The current version can handle convolutions where all distributions are discrete, all are continuous, or some are is discrete and some continuous, but it cannot handle convolutions in which one or more distributions are mixed (i.e., partly discrete and partly continuous).

I would be very happy for suggestions on how to augment PMETRIC's handling of convolutions, especially those accompanied by Pascal code.

ConvolutionIID(N,RV) This is just an easier way to specify a convolution when all N summed random variables have the same distribution, RV.

```
ConvolutionIID(3,Uniform(0,1))
```

is the same as

```
Convolution(Uniform(0,1),Uniform(0,1),Uniform(0,1))
```

Difference(RV1,RV2) This is the distribution of the difference of two *independent* random variables, RV1 minus RV2, where RV1 and RV2 are each legal distributions in their own right. For example,

```
Difference(Uniform(0,1),Uniform(0,1))
```

specifies a difference between two standard uniform distributions, which ranges from -1 to 1 (not uniformly). PMETRIC handles difference distributions dumbly, like convolutions. The current version can handle differences where both distributions are discrete, both are continuous, or one is discrete and one continuous, but it cannot handle differences in which one or both distributions are mixed (i.e., partly discrete and partly continuous).

Mixture(p1,RV1,p2,RV2,...,pk,RVk) Mixtures are distributions formed by randomly selecting one of a number of random variables. For example, `Mixture(0.5,Normal(0,1),0.5,Uniform(0,1))` defines a random variable that comes from a standard normal half the time and a standard uniform the other half of the time. In general, the format of this distribution is:

`Mixture(`p_1`,BasisDist1(Parms),`p_2`,BasisDist1(Parms),...,`p_k`,BasisDistK(Parms))`

and the p_i's must sum to one (it is also legal to omit p_k).

InfMix(RV1,MixParm,RV2(Parms)) The InfMix distribution is an infinite mixture, formed when a parameter of one distribution is itself randomly distributed according to another distribution. For example,

```
InfMix(Normal(0,5),1,Uniform(10,20))
```

defines a random variable that comes from a normal distribution with standard deviation 5. The first parameter of that distribution (as signified by the "1" between the two distribution names) follows a uniform distribution from 10 to 20. As another example, `InfMix(Normal(0,5),2,Uniform(10,20))` defines a random variable that comes from a normal distribution with mean zero and standard deviation varying uniformly from 10 to 20. In general, the format of this distribution is:

```
InfMix(ParentDist(Parms),MixParm,ParmDist(Parms))
```

where ParentDist is a distribution, MixParm is an integer indicating whether the first, second, ..., parameter of the ParentDist varies randomly, and ParmDist is the distribution of that parameter.

InfMix may be used recursively. For example,

```
InfMix(InfMix(Normal(0,5),1,Uniform(0,2)),2,Uniform(4,6))
```

defines a normal distribution in which the mean is uniform(0,2) and the standard deviation is uniform(4,6).

Limitations: (1) At present, computations of the upper and lower bounds of InfMix distributions assume that the largest and smallest values of the random variable are obtained when the underlying ParmDist is at its two extremes. (2) Extreme caution is needed with these distributions because problems often arise in numerical integration. I have found it helpful to increase the IntegralMinSteps to 10, which was enough in most of the cases I've looked at, but you may need to adjust this up (for precision) or down (for speed) in your cases.

Truncated(RV,Min,Max) A truncated distribution is a conditional distribution, conditioning on the random variable RV falling within the interval from Min to Max. For example, `Truncated(Normal(0,1),-1,1)` defines a random variable that is always between -1 and 1, and which within that interval has relative probabilities defined by the PDF of the standard normal. In general, the format of this distribution is:

```
Truncated(BasisDistribution(Parms),Min,Max)
```

It is sometimes convenient to specify the truncation boundaries in terms the probabilities you want to cut off rather than the scores themselves. For example, you might want to look at the middle 90% of a normal distribution but might not immediately know which scores cut off the top and bottom 5%. For this reason, there is a variant of the command that takes probabilities instead of values for min and max, like this:

```
TruncatedP(BasisDistribution(Parms),0.05,0.95)
```

With `TruncatedP`, PMETRIC will use its InverseCDF function to find the score values that correspond to the cumulative probabilities that you specify, and then truncate at those score values.

Bounded(RV,Min,Max) *I do not know if this is a standard type of distribution or not, and would appreciate any comments on it from those in the know.* A bounded distribution is similar to a truncated distribution in that the random variable must fall within the range of Min to Max. The difference is that all values less than Min are converted to Min, and all values less than Max are converted to Max. Thus, there are discrete masses of probability at Min and Max, and the probability density function between Min and Max is not conditionalized.

For example, consider the distribution `Bounded(Normal(0,1),-1,1)`. This is really a mixture of these three distributions:

Distribution	Mixture Probability
Constant(-1)	0.1587
Truncated(Normal(0,1),-1,1)	0.6826
Constant(1)	0.1587

Note that 0.1587 is the probability that a normal(0,1) score is less than -1, and also the probability that it is greater than 1. Bounding the distribution thus means taking all of the probability density higher than the upper value and massing it at that value.

As with the truncated distribution, there is a form of the Bounded distribution based on probabilities, as in:

```
BoundedP(Normal(0,1),0.1,0.9)
```

.

Order(k,RV1,RV2,RV2,...,RVn) The distribution of this order statistic is the distribution of the k'th largest observation in a sample of n independent observations from the n indicated random variables. For example,

```
Order(2,Normal(0,1),Uniform(0,1),Exponential(1))
```

defines a random variable that is the median (2nd largest) in a sample containing one score from the standard normal, one from the uniform from 0–1, and one from the exponential with rate 1. In general, the format of this distribution is:

`Order(`k`,BasisDist1(Parms),...,BasisDistN(Parms))`

In the special case where the basis distributions are all identical, it is more convenient to use the OrderIID distribution, described next.

OrderIID(k, n,RV) This is the special case of the order distribution in which the basis distributions are identical as well as independent. In general, the format of this distribution is:

`OrderIID(`k`,`N`,BasisDist(Parms))`

It is only necessary to specify the basis distribution once, since all are identical; instead, you have to specify how many there are (N).

OrdExp(i, n, λ) This is the special case of OrderIID in which the basis distribution is an exponential with rate λ, and you want the i'th order statistic in a sample of n ($1 \leq i \leq n$). For this case there are nice fast closed forms for the mean and variance that were given to me by Rolf Ulrich, Dept. of Psychology, Univ. of Wuppertal, Germany.

OrdBinary(i,n1,RV1,n2,RV2) This is an order distribution with two types of underlying RVs. For example,

```
OrdBinary(2,5,Normal(0,1),7,Uniform(0,1))
```

specifies the distribution of the second order statistic in samples of 12 made up of five standard normals and seven standard uniforms.

MinBound(RV1,RV2) Consider two arbitrary random variables X and Y, which may or may not be independent, and let $Z \equiv \min(X, Y)$. The CDFs of these three random variables must obey the inequality

$$F_z(t) \leq F_x(t) + F_y(t) \quad \text{for all } \ t$$

because

$$F_z(t) = F_x(t) + F_y(t) - \Pr(X \leq t \& Y \leq t)$$

Thus, for any two basis RVs X and Y, we can construct the random variable Z which is a lower bound on the distribution of $\min(X, Y)$:

$$F_z(t) = \begin{cases} F_x(t) + F_y(t) & \text{if } \ F_x(t) + F_y(t) < 1 \\ 1 & \text{if } \ F_x(t) + F_y(t) \geq 1 \end{cases}$$

MinBound implements this lower bound distribution for any two arbitrary random variables X and Y.

Because distributions are constructed recursively, it is legal within PMETRIC to construct weird distributions by any combination of the above. For example, this would be legal:

```
Truncated(Mixture(.5,Normal(0,1),.5,OrderIID(4,5,Normal(0,1))),-1,1)
```

and it indicates a truncated mixture of a normal distribution and an order statistic.

It does not appear to me that there will ever be any ambiguity about what distribution is requested within the syntax of PMETRIC, but let me know if you find such a case!

14.5 Bin-Based Distributions

PMETRIC has several bin-based distributions that can be used to construct arbitrary distributions and model any data pattern you like (e.g., ones estimated by tabulating lots of data). Each distribution is made up of NBins adjacent, non-overlapping, equal-width bins, with an arbitary probability of occurrence within its bin. The different bin-based distributions differ in their assumptions about the details of the distribution within each bin, as described below.

Histogram The histogram is a continuous distribution with a flat PDF within each bin.

Polygon The polygon is a continuous distribution with a linear but not necessarily flat PDF within each bin. More specifically, the PDF of the polygon distribution is defined by a series of NBins+1 points; the first point gives the height of the PDF at the lower bound of the distribution, and the remaining NBins points give the heights of the PDF at the top of each bin (the top of the top bin showing the PDF at the upper bound of the distribution). Within each bin, the PDF is a straight line going from the height at the bottom of the bin to the height at the top of the bin.

FreqPolygon The frequency polygon is also a continuous distribution with a linear but not necessarily flat PDF within each bin. Unlike the polygon, though, it keys on the PDF in the middle of each bin rather than the upper edge. More specifically, the PDF of this distribution is defined by a series of NBins+2 points; the first point gives the height of the PDF at the lower bound of the distribution, the last point gives the height of the PDF at the upper bound of the distribution, and the remaining NBins points give the heights of the PDF at the center of each bin. Within each bin, the PDF is formed by straight lines going from the height at the bin's center to the heights at the centers of the adjacent bins.

BinCen The "BinCenters" random variable is a discrete distribution with all of the probability mass associated with each bin assigned to the midpoint of the bin.

The commands `Histogram(FileName)`, `FreqPolygon(FileName)`, `Polygon(FileName)`, and `BinCen(FileNa` tell PMETRIC to read the description of the indicated distribution from the indicated file. The first line in the file must contain three numbers:

1. The minimum value in the distribution (i.e., the lower edge of the lowest bin).
2. The maximum value in the distribution (i.e., the upper edge of the highest bin).
3. The number of bins, NBins.

For example, this line might be

```
-10 100 200
```

to indicate a distribution ranging from -10 to 100, with 200 bins. (Note that in this example each bin would be $[100 - (-10)]/200 = 0.55$ units wide.) Following the first line, there should be NBins+1 additional lines with one number per line. The first line is special: It should be zero for the Histogram and BinCen distributions, but for the Polygon distribution it should be the height of the PDF at the lower bound of the distribution (which may be zero but need not be). The remaining numbers correspond to the probabilities for the successive NBins bins, from smallest to largest. Note that these numbers need not actually *equal* the bin probabilities; it is sufficient for them to *be proportional to* the bin probabilities. PMETRIC automatically rescales them so that the total probability sums to one, so you could input frequency counts or PDF heights instead of actual bin probabilities.

14.6 Approximation Distributions

The above bin-based distributions can also be used as "approximation distributions," the purpose of which is to speed up computations with complicated underlying "basis" distributions. These approximations are particularly useful when (a) you are interested in a basis distribution for which it is time-consuming to compute values, and (b) you want to compute lots of different values from this distribution without changing its parameters. In these cases, initializing the approximation distribution will be a little slower than initializing the basis distribution, but then all further computations will be much faster with the approximation.

Some terminology and notation is used in common across all approximation distributions. Each approximation uses “bins”, which are small, nonoverlapping ranges of the dependent variable. For example, a beta distribution is defined over the range from 0.0 to 1.0, and it might be approximated using 100 bins: 0.00–0.01, 0.01–0.02, ..., 0.99–1.00. The number of bins (100 in this example) will be referred to as “NBins,” and the width of each bin will be referred to as “W.” Of course, the approximation are slower to compute but more accurate with a larger number of bins (smaller W). I find that 200–300 bins is usually enough, and that with approximately symmetric distributions it is generally better to use an odd number of bins.

In practice, it may be somewhat tricky to decide which is the best approximation to use with a given basis distribution. I know of no sure strategy other than trial and error, but offer some comments on the different approximations based on my limited experience with them.

ApprPolygon(RV) This is a continuous approximation that can be used only for a continuous basis distribution, RV. In brief, the PDF of the approximation distribution is a set of NBins straight lines, matched to the height of the basis distribution’s PDF at the bin’s edges (further detail is given below). For example,

```
ApprPolygon(Convolution(Normal(0,1),Beta(2,2)),201)
```

approximates the specified convolution with a set of 201 straight lines.

ApprFreqPolygon(RV) This is a continuous approximation that can be used only for a continuous basis distribution, RV. In brief, the PDF of the approximation distribution is a set of NBins straight lines, matched to the height of the basis distribution’s PDF at the bin’s centers. For example,

```
ApprFreqPolygon(Convolution(Normal(0,1),Beta(2,2)),201)
```

approximates the specified convolution with a set of 201 straight lines.

This is my preferred type of approximation. It is generally quite accurate, and it is often much faster than the other approximations.

Details of construction.

Step 1: The first line starts at X_1=minimum (of the basis distribution) with height PDF at that point and goes to X_2=minimum+W/2 with height PDF=Basis.PDF(X_2). The second line continues from the end of the first line to the point with X_3=minimum+1.5*W and height PDF=Basis.PDF(X_3). And so on, with the final line segment ending at the maximum of the basis distribution and PDF at the maximum.

Step 2: The PDF just constructed is integrated, and the heights are scaled up or down appropriately so that the total area is 1.00.

ApprHistogram(RV) This is a continuous approximation that can be used for either a discrete or a continuous basis distribution, RV. In brief, the PDF is of the approximation distribution is a set of NBins flat lines, as if the basis distribution were uniform within each bin (like in a histogram). For example,

```
ApprHistogram(Convolution(Normal(0,1),Beta(2,2)),201)
```

approximates the specified convolution with a set of 201 bins with equal probability within each bin.

This approximation is more general than ApprPolygon, because it can be used with discrete distributions, and it is less sensitive to abruptly-changing PDFs. But it is usually slower to construct initially, and it is often much slower to do any computations with. The PDF has discontinuities at the bin boundaries (unlike ApprPolygon), and these make numerical integrations converge more slowly.

Details of construction. The CDF of the basis distribution is computed at the top and bottom of each bin, and from these the bin probability is computed. Then, the height of the uniform approximation PDF within that bin is adjusted to give this bin probability.

ApprBinCen(RV) This is a discrete approximation, and it can be used to approximate either a discrete or a continuous basis distribution, RV. In brief, the approximation assumes that all of the probability mass is concentrated in a single point at the center point of each bin; moreover, any value in the bin is treated as if it were that center point.

Details of construction. The CDF of the basis distribution is computed at the top and bottom of each bin, and from these the bin probability is computed. All of this probability mass is assigned to the value at the center of the bin. For purposes of PDF and CDF computations, all values within a bin are treated as equivalent to the center.

Using CUPID, approximations can be written out to files and read in from files. This is useful for saving the time-consuming initialization phase if you want to return to an approximation later, and it also allows you to prepare your own approximation distributions separately and then import them into PMETRIC for further analysis.

To write the current approximation from within CUPID, use the command `BinsWrite(FileName)`. To read the approximation back in, use the command `BinsRead(FileName)`. The format of the approximation file is just the same as that for the bin-based distributions described in the previous section, with one exception: The very first line of the file contains the name of the basis distribution.

14.7 Construction Distributions

Given the approximation distributions described in the previous section, it seemed natural to include another type of approximation distribution that I call "construction distributions." These are approximation distributions constructed simply by tabulating many calls to the random number generator for any arbitrary distribution. I have found them useful mostly in my own simulation work, where I know how to write an RNG for the variable that I am interested in but I don't really know anything else about how to characterize it. In these situations, I have found it convenient to construct an approximation to the true distribution by calling the RNG many times and tabulating the results. The construction distributions simply automate this process. I imagine that they will be valuable only to programmers who can compile their own RNG source code using these routines, but I describe them here for completeness.

There are three types of construction distributions:

ConsHistogramRV A constructed HistogramRV used to approximate a basis distribution.

ConsFreqPolygonRV A constructed FreqPolygonRV used to approximate a basis distribution.

ConsBinCenRV A constructed : BinCenRV used to approximate a basis distribution.

Note that no constructed PolygonRV exists.

14.8 Distributions Arising in Connection with Signal Detection Theory

In addition to the above standard and derived distributions, I have added a few distributions that corresponded to particular projects I happened to be working on. The distributions described in this section arise in connection with signal detection theory experiments, and will be of interest to some psychophysicists and perhaps engineers. If you don't know what signal detection theory is, then it is unlikely that you will care about these. Note: These are all discrete distributions, as each reflects the outcome of one or two binomial-type conditions with a finite number of trials.

ZfromP(SampleSize,TrueP,Adjust) This is the discrete distribution of Z, which is derived from the binomial distribution as follows:

1. For any sample from a Binomial(N, P), convert the number of successes k to the probability of success, $p \equiv k/N$. If $p = 0$, set $p =$ Adjust$/N$; if $p = 1$, set $p = 1 -$ Adjust$/N$. "Adjust" is a parameter between 0 and 1, specified by the user, to indicate how the extreme data values should be treated.
2. Find Z such that $p = \Pr(z \le Z)$, where z is a random variable having the standard normal distribution.

APrime(NSignalTrials,PrHit,NNoiseTrials,PrFalseAlarm) This is the distribution of the sample A' computed from an experiment with NSignalTrials signal trials each having the specified true probability of a hit, and NNoiseTrials noise trials each having the specified true probability of a false alarm. Specifically, A' is the distribution-free estimate of the area under the ROC curve computed using Equations 2 and 9 of Aaronson and Watts (1987).

APrimeSym(NTrials,PC) This is a shortcut for the previous distribution that can be used when there are equal numbers of signal and noise trials and when the probability of a correct response (hit or correct rejection) is the same for both signal and noise trials.

YNdPrime(NSignalTrials,PrHit,NNoiseTrials,PrFalseAlarm,Adjust) This is the distribution of the sample $\hat{d}'$ computed from an experiment with NSignalTrials signal trials each having the specified true probability of a hit, NNoiseTrials noise trials each having the specified true probability of a false alarm, and using the Adjust factor (between 0 and 1) to correct cases with 0% or 100% hits or false alarms (e.g., replace 0 hits with Adjust hits, and replace NSignalTrials hits with [NSignalTrials - Adjust] hits). Programming note: If PDFs are requested, this distribution is implemented using the List (smaller samples) and AppApprCen (larger samples) random variables.

YNdPrimeSym(NTrials,TrueDP,Adjust) This is the special case of YNdPrime in which NSignalTrials = NNoiseTrials and Pr(Hit) = 1 - Pr(FA). Note that the second parameter is the true d' rather than the hit probability.

15 Release History

Version 1.1 was released in September, 2001.

Beta version 1.0 was sent to a few interested parties for preliminary testing in June 2000.

16 Registration and Author Contact Address

I would really like to receive feedback on who is using this software, for what purposes. So, please e-mail me at miller@otago.ac.nz if you found this software useful. If you do, I will add your name to my mailing list and let you know about any new versions, bugs, or new programs that might interest you. If you use this software for any published research, I would greatly appreciate it if you would acknowledge the software in your article (e.g., in a footnote) and email me a citation to the article or, better yet, send me a reprint. Of course I would also welcome bug reports and suggestions for improvement, too, although I can't promise any fast action on those.

Here is how to contact me:

Prof Jeff Miller
Department of Psychology
University of Otago
Dunedin, New Zealand
email: miller@otago.ac.nz
FAX: (64-3)-479-8335

17 OS/2 Versions

I will happily provide executable OS/2 versions of this software to others who still use this operating system, as I do myself. The only difference from the versions described in this documentation is that the "SET CUPID=C:\CUPID" command (if needed) goes in the config.sys file rather than the autoexec.bat file. (Don't forget you have to shutdown and reboot for this to take effect.)

18 Acknowledgements

I have obtained information about probability distributions from a variety of sources, including the internet. The most important are the references listed below. Rolf Ulrich has also supplied quite a lot of information about various

probability distributions and numerical methods. Thanks also to Roman Krejci, from whom I got the pascal code for the Mersenne Twister. Others who have helped, directly or indirectly, include Timo Salmi and Duncan Murdoch. Special credit goes to Alann Lopes, who showed me how object-oriented programming could be useful in the first place.

19 References

References

Aaronson, D., & Watts, B. (1987). Extensions of Grier's computational formulas for A' and B" to below-chance performance. *Psychological Bulletin, 102,* 439–442.

D'Agostino, R. B. (1970). Simple compact portable test of normality: Geary's test revisited. *Psychological Bulletin, 74,* 138–140.

Dallal, G. E., & Wilkinson, L. (1986). An analytic approximation to the distribution of Lilliefors's test statistic for normality. *The American Statistician, 40,* 294–296.

Devroye, L. (1986). *Non-uniform random variate generation.* Berlin: Springer-Verlag.

Evans, M., Hastings, N., & Peacock, B. (1993). *Statistical distributions. (2nd ed.)* New York: Wiley.

Finney, D. J. (1971). *Probit analysis: A statistical treatment of the sigmoid response curve. [3rd ed.].* Cambridge, England: Cambridge University Press.

Finney, D. J. (1978). *Statistical method in biological assay.* London: Charles Griffin.

Gescheider, G. A. (1997). *Psychophysics: The fundamentals. [3rd ed.].* Hillsdale, NJ: Erlbaum.

Guilford, J. P. (1936). *Psychometric methods.* New York: McGraw-Hill.

Johnson, N. L., & Kotz, S. (1970). *Continuous univariate distributions.* New York: Houghton Mifflin.

Matsumoto, M., & Nishimura, T. (1998). Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Transactions on Modeling and Computer Simulation, 8,* 3–30.

Press, W. H., Flannery, B. P., Teukolsky, S. A., & Vetterling, W. T. (1986). *Numerical recipes: The art of scientific computing.* Cambridge, England: Cambridge University Press.

Quick, R. F. (1974). A vector magnitude model of contrast detection. *Kybernetik, 16,* 65–67.

Rosenbrock, H. H. (1960). An automatic method for finding the greatest or least value of a function. *Computer Journal, 3,* 175–184.

Sternberg, S., & Knoll, R. L. (1973). The perception of temporal order: Fundamental issues and a general model. In S. Kornblum (Ed.), *Attention and performance IV* (pp. 629–685). New York: Academic Press.

Strasburger, H. (2001). Converting between measures of slope of the psychometric function. *Perception & Psychophysics, 63,* 1348–1355.

20 Software License

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc. 675 Mass Ave, Cambridge, MA 02139, USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

20.1 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

20.2 Terms of License

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The ”Program”, below, refers to any such program or work, and a ”work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term ”modification”.) Each licensee is addressed as ”you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

> b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
>
> c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

> a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
>
> b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
>
> c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions.

You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and ”any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM ”AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS