A DIAGNOSTIC FUNCTION TO EXAMINE CANDIDATE DISTRIBUTIONS TO MODEL UNIVARIATE DATA

by

JOHN RICHARDS

B.S., Kansas State University, 2008

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Statistics
College of Arts and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2010

Approved by:

Major Professor
Suzanne Dubnicka, Ph.D.

Copyright

Abstract

To help with identifying distributions to effectively model univariate continuous data, the R function `diagnostic` is proposed. The function will aid in determining reasonable candidate distributions that the data may have come from. It uses a combination of the Pearson goodness of fit statistic, Anderson-Darling statistic, Lin's concordance correlation between the theoretical quantiles and observed quantiles, and the maximum difference between the theoretical quantiles and the observed quantiles. The function generates reasonable candidate distributions, QQ plots, and histograms with superimposed density curves. When a simulation study was done, the function worked adequately; however, it was also found that many of the distributions look very similar if the parameters are chosen carefully. The function was then used to attempt to decipher which distribution could be used to model weekly grocery expenditures of a family household.

Table of Contents

List of Figures

List of Tables

Acknowledgements

Dr. Suzanne Dubnica, Dr. Gary Gadbury, Dr. Haiyan Wang, Dr. James Neill, Dr. James Higgins, Dr. Juan Du, Scott Kreider, Heath Yates, Jennifer Richards, Patrick Richards, Irene Richards, Howard Richards, Elda Jean Richards, Adam Wathen

CHAPTER 1 - Introduction

1.1 The Problem

There exist many probability density functions that may be used to model all kinds of univariate data. It is common to not know exactly what probability distribution a particular data set is sampled from, or, at least, an appropriate distribution to model the data. The data analyst may have some preconceived notion of what he might expect the shape of the data to look like; however, many times no assumption about the population distribution is made. This can be problematic because many statistical tests require that data follow a particular distribution. If the data analyst wants to explore what population distribution may have generated the observed data, he would have to examine each candidate probability distribution individually. Given the number of practical distributions, this can be tedious and inefficient. There does not exist a program that takes univariate sample data and provides, to the data analyst, some candidate distributions that can be effectively used to model said data. I propose a function that uses ten commonly used probability distributions and would be used as a diagnostic tool in which the data analyst can explore a few distributions simultaneously that can model the data in question. The function estimates the parameters of each distribution, provides ten histograms of the data with each of the probability density functions superimposed upon each of them, provides probability plots for each of ten distributions, and computes numerical measures of strength to determine reasonable candidate population distributions.

1.2 Ten Commonly Used Distributions

Below is a summary of ten distributions which the function uses. Obviously, this is not an exhaustive list, but the data analyst can use this information to explore other distributions that may look similar to one or more of these ten, but not included in this function. The ten distributions used are the normal, gamma, exponential, logistic, lognormal, Weibull, Cauchy, Laplace, uniform, and Pareto.

1.2.1 Normal Distribution

Because of the widespread usage and familiarity of the normal distribution, the discussion of the normal distribution is minimal. The normal distribution has a location parameter, μ, and a scale parameter, σ. The probability density function of the normal distribution is as follows:

$$f(x \mid \mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]I(-\infty < x < \infty) \tag{1.1}$$

for -∞ < μ < ∞ and σ > 0.

The maximum likelihood estimator of μ is the sample mean. The maximum likelihood estimator of σ^2 is as follows:

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2. \tag{1.2}$$

1.2.2 Gamma Distribution

The gamma distribution is a right-skewed distribution with a shape parameter α and a scale parameter β. The probability density function of the gamma distribution is as follows:

$$f(x \mid \alpha,\beta) = \frac{1}{\Gamma(\alpha)\beta^{\alpha}}x^{\alpha-1}\exp\left(-\frac{x}{\beta}\right)I(x \geq 0) \tag{1.3}$$

for α, β > 0.

The maximum likelihood estimators of α and β are the solutions of the following equations:

$$\begin{aligned}&\frac{1}{n}\sum_{i=1}^{n}\ln(x_i) - \ln(\bar{x}) - \psi(\alpha) + \ln(\alpha) = 0\\ &\alpha\beta - \bar{x} = 0\end{aligned} \tag{1.4}$$

where ψ(α) denotes the digamma function.

This distribution has many applications, including modeling ranges for normal populations (Johnson, Kotz, Balakrishnan, 1994), meteorological precipitation processes (Kotz and Neumann, 1963), and wait times (Krishnamoorthy, 2006). Even though the gamma distribution is right-skewed for all values of the parameters, if α is large enough, the gamma distribution can look very similar to the normal distribution or the lognormal distribution.

1.2.3 Exponential Distribution

The exponential distribution is a special case of the gamma distribution, where α = 1. From this, the probability density function is as follows:

$$f(x \mid \beta) = \frac{1}{\beta}\exp\left(-\frac{x}{\beta}\right)I(x \geq 0) \tag{1.5}$$

for β > 0.

The maximum likelihood estimator of β is simply the sample mean:

$$\hat{\beta} = \bar{x}. \tag{1.6}$$

The classical usage of the exponential distribution is to model wait times of a Poisson process.

1.2.4 Logistic Distribution

The logistic distribution is a symmetric distribution with a location parameter μ and a scale parameter β. The probability distribution function is as follows:

$$f(x \mid \mu,\beta) = \frac{1}{\beta}\frac{\exp[-(x-\mu)/\beta]}{\{1+\exp[-(x-\mu)/\beta]\}^2}I(-\infty < x < \infty) \tag{1.7}$$

for -∞ < μ < ∞ and β > 0.

The maximum likelihood estimators of μ and β are the solutions to the following equations:

$$\sum_{i=1}^{n}\left[1+\exp\left(\frac{x_i-\mu}{\beta}\right)\right]^{-1} = \frac{n}{2}$$

$$\sum_{i=1}^{n}\left(\frac{x_i-\mu}{\beta}\right)\frac{1-\exp\left(\frac{x_i-\mu}{\beta}\right)}{1+\exp\left(\frac{x_i-\mu}{\beta}\right)} = n. \tag{1.8}$$

It is common to compute preliminary estimates of the parameters to aid in the numerical solutions, and then use an optimization technique to arrive at the maximum likelihood estimators. A preliminary estimate for μ is the sample mean. The preliminary estimate of β is as follows:

$$\frac{\sqrt{3}}{\pi} s = \hat{\beta} \tag{1.9}$$

where *s* is the sample standard deviation.

The logistic distribution is often used as a substitute for the normal distribution because of its symmetry (Krishnamoorthy, 2006). The main difference in the shape of the normal distribution as compared to the logistic distribution is that the logistic has heavier tails. A detailed discussion of many applications of the logistic distribution can be found in a book by Balakrishnan (1992).

1.2.5 Lognormal Distribution

The lognormal distribution can be directly derived from the normal distribution. If the natural logarithm of X is normally distributed, then X is lognormally distributed. The probability density function of the lognormal distribution is as follows:

$$f(x \mid \mu, \sigma) = \frac{1}{\sigma x \sqrt{2\pi}} \exp\left[-\frac{(\ln x - \mu)^2}{2\sigma^2}\right] I(-\infty < x < \infty) \quad . \tag{1.10}$$

The application of the lognormal distribution is widespread and mostly used to model positive right-skewed data in which the natural logarithmic transformation is approximately normal. The lognormal distribution can look similar to a normal distribution if the values of the scale parameter is small and the location parameter is large. The lognormal distribution can also look similar to the gamma distribution and is often used as a substitute for the gamma distribution (Krishnamoorthy, 2006).

1.2.6 Weibull Distribution

The Weibull distribution is a right-skewed distribution with a shape parameter γ and a scale parameter η. The probability density function is as follows:

$$f(x \mid \gamma,\eta) = \frac{\gamma}{\eta}\left(\frac{x}{\eta}\right)^{\gamma-1} \exp\left[-\left(\frac{x}{\eta}\right)^{\gamma}\right] I(x \geq 0) \tag{1.11}$$

for $\gamma,\eta > 0$. If $\gamma = 1$, then this reduces to the probability density function exponential distribution.

The maximum likelihood estimators are the solutions to the following equations:

$$\eta = \left[\frac{1}{n}\sum_{i=1}^{n} x_i^{\gamma}\right]^{1/\gamma}$$

$$\gamma = \left[\left(\sum_{i=1}^{n} x_i^{\gamma} \ln x_i\right)\left(\sum_{i=1}^{n} x_i^{\gamma}\right)^{-1} - \frac{1}{n}\sum_{i=1}^{n} \ln x_i\right]^{-1} . \tag{1.12}$$

This distribution has applications in reliability theory (Krishnamoorthy, 2006) and was first used to model the breaking strength of materials (Johnson, Kotz, Balakrishnan, 1992). Like the gamma distribution, if the scale parameter is large, this distribution looks symmetric and can look similar to a normal distribution.

1.2.7 Cauchy Distribution

The Cauchy distribution is a symmetric distribution with extremely heavy tails. It has a location parameter θ and a scale parameter σ. The probability density function is as follows:

$$f(x \mid \theta,\sigma) = \left\{(\pi\sigma)\left[1 + \left(\frac{x-\theta}{\sigma}\right)^2\right]\right\}^{-1} I(-\infty < x < \infty) \tag{1.13}$$

for $-\infty < \theta < \infty$ and $\sigma > 0$.

The maximum likelihood estimators are the solutions of the following equations:

$$\frac{1}{n}\sum_{i=1}^{n} \frac{2}{1 + \left(\frac{x_i - \theta}{\sigma}\right)^2} = 1 ,$$

$$\frac{1}{n}\sum_{i=1}^{n} \frac{2x_i}{1 + \left(\frac{x_i - \theta}{\sigma}\right)^2} = \theta . \tag{1.14}$$

Although the Cauchy distribution is mostly used as a diabolical example, it has been used in applications involving ratios of normally distributed random variables (Casella and Berger, 2002) and in physical sciences (Krishnamoorthy, 2006).

1.2.8 Laplace Distribution

The Laplace distribution is also known as the double exponential distribution. It has a location parameter μ and a scale parameter σ. The probability density function is as follows:

$$f(x \mid \mu,\sigma) = \frac{1}{2\sigma}\exp\left(\frac{-|x-\mu|}{\sigma}\right)I(-\infty < x < \infty) \tag{1.15}$$

for $-\infty < \mu < \infty$ and $\sigma > 0$.

The maximum likelihood estimator of μ is the sample median:

$$\hat{\mu} = median\{x_1,\ldots,x_n\}. \tag{1.16}$$

The maximum likelihood estimator of σ is as follows:

$$\hat{\sigma} = \frac{1}{n}\sum_{i=1}^{n}|x_i - \hat{\mu}|. \tag{1.17}$$

The most striking feature of the Laplace distribution is its peak located at the value of the location parameter; the probability density function is nondifferentiable at $x=\mu$. The classic usage of the Laplace distribution is modeling differences in two independent exponential distributed random variables. It has also been used to model breaking strength data (Krishnamoorthy, 2006).

1.2.9 Uniform Distribution

The uniform distribution has a probability density function as follows:

$$f(x \mid a,b) = \frac{1}{b-a}I(a \le x \le b) \tag{1.18}$$

for $-\infty < a < b < \infty$.

The maximum likelihood estimator of a is simply the sample minimum, and the maximum likelihood estimator of b is the sample maximum.

A common usage of the uniform distribution is modeling p-values of statistical tests assuming the null hypothesis is true. It is also commonly used as a noninformative prior distribution in Bayesian analysis.

1.2.10 Pareto Distribution

The Pareto distribution is a heavy right-skewed distribution with a location parameter α and a scale parameter β. The probability density function is as follows:

$$f\left(x \mid \alpha,\beta\right) = \frac{\beta\alpha^{\beta}}{x^{\beta+1}} I\left(x > \alpha > 0\right) \tag{1.19}$$

for α,β > 0.

The maximum likelihood estimator of α is simply the sample minimum:

$$\hat{\alpha} = \min\left\{x_1,\ldots,x_n\right\}. \tag{1.20}$$

The maximum likelihood estimator of β is as follows:

$$\hat{\beta} = \left\{\ln\left[\frac{\left(\prod_{i=1}^{n} x_i\right)^{1/n}}{\hat{\alpha}}\right]\right\}^{-1}. \tag{1.21}$$

The Pareto distribution is mostly used to model population size and income because of its extremely heavy tail (Krishnamoorthy, 2006).

1.3 Concordance Correlation, Goodness of Fit, Anderson-Darling, and Maximum Distance

The measures of strength that I used are (1) the concordance correlation proposed by Lin (1989), (2) a goodness of fit measure similar to Fisher's Goodness of Fit statistic, (3) Anderson-Darling statistic, and (4) the maximum distance from the theoretical quantiles assuming a particular distribution to the observed order statistic.

1.3.1 Concordance Correlation

Lin (1989) proposed the concordance correlation. This is a measure that was originally used to identify reproducibility of measurements taken from newly developed instruments. It not only measures the strength of linear relationship, but also measures the variation from a line with slope equal to one and intercept equal to zero. If data came from a particular distribution, the theoretical quantiles and the observed quantiles should follow an approximately 45 degree angle line. The concordance correlation is a measure of this deviation from the 45 degree angle line. The theoretical value of the concordance correlation is as follows:

$$\rho_c = 1 - \frac{E\left[\left(X_1 - X_2\right)^2\right]}{\sigma_1^2 + \sigma_2^2 + \left(\mu_1 - \mu_2\right)^2} \quad (1.22)$$

The numerator is the expected squared perpendicular deviation from the 45 degree angle line, and the denominator is the expected squared perpendicular deviation from the 45 degree angle line when X_1 and X_2 have a covariance of zero.

The estimator for the concordance correlation is computed as follows:

$$\hat{\rho}_c = \frac{2S_{12}}{S_1^2 + S_2^2 + \left(\bar{x}_1 - \bar{x}_2\right)^2} \quad (1.23)$$

where

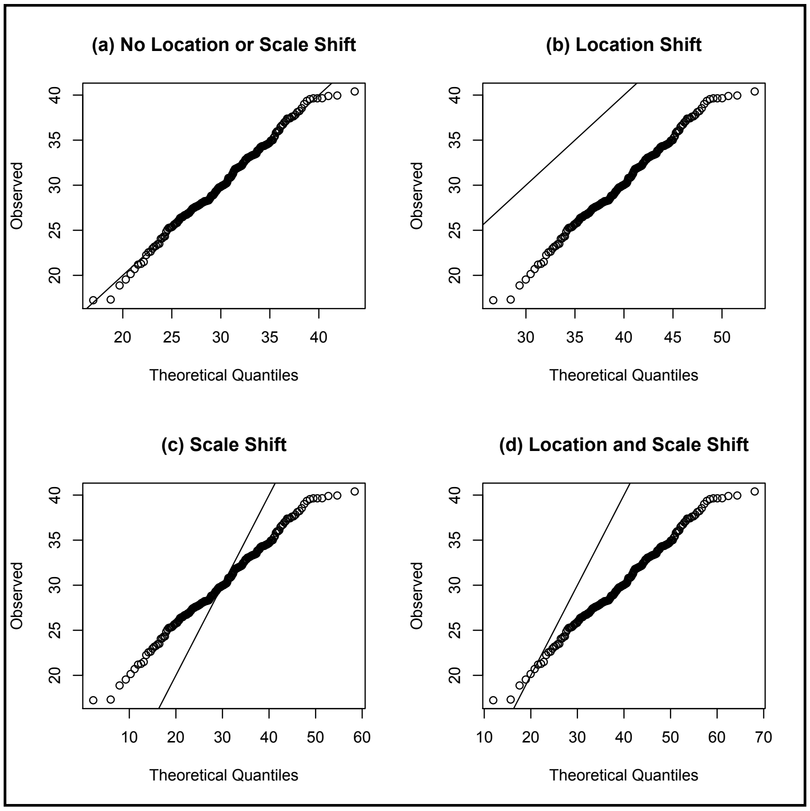
$$\bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$$

$$S_j^2 = \frac{1}{n}\sum_{i=1}^{n}\left(x_{ij} - \bar{x}_j\right)^2$$

$$S_{12} = \frac{1}{n}\sum_{i=1}^{n}\left(x_{i1} - \bar{x}_1\right)\left(x_{i2} - \bar{x}_2\right) \quad (1.24)$$

Because the concordance correlation measures the deviation from the 45 degree line through the origin, the measure is sensitive to shifts in location and scale. Figure 1.1 provides four instances of location and scale shifts. A random sample of 200 observations was generated from a normal population with mean of 30 and standard deviation of 5. From this sample, the mean was estimated to be 30.32 and the standard deviation was estimated to be 4.751. Plot (a) in Figure 1.1 is the QQ plot using the mean and standard deviation estimates to generate

Figure 1.1 Shifts in Location and Scale and Its Effect on QQ Plots



the theoretical quantiles. The concordance correlation was computed as 0.996. Plot (b) in Figure 1.1 is the QQ plot using a mean of 40 and the standard deviation of 4.751 to generate the theoretical quantiles. When only the location was shifted, the ordered pairs shift parallel to the 45 degree angle line. The concordance correlation for this instance was only 0.322. Plot (c) in Figure 1.1 is the QQ plot using a mean of 30.40 and a standard deviation of 10 to generate theoretical quantiles. When only the scale was changed, the ordered pairs rotated at the center. The concordance correlation was 0.773. Finally, Plot (d) in Figure 1.1 is the QQ plot using a mean of 40 and a standard deviation of 10 to generate the theoretical quantiles. When both the location and the scale were changed, the ordered pairs rotated at the base. The concordance correlation was 0.436.

1.3.2 Goodness of Fit

To identify if data follow a particular distribution or not, a goodness of fit test can be used. A common test is the chi-squared goodness of fit test. This test can be used for either discrete data or for continuous data. Since the proposed function only considers continuous distributions, the goodness of fit statistic used is only applicable to continuous data. Because this statistic can only apply to counts, it is necessary to section the data into *k* bins. This means the data are partitioned into separate ranges. Then a count of the observations within each range is recorded. This is then compared to the expected count of the same range, given that the data follow a specific distribution. The test statistic is then given as follows:

$$\chi^2 = \sum_{i=1}^{k} \frac{\left(O_i - E_i\right)^2}{E_i} \tag{1.25}$$

where O_i is the number of observed frequencies within each bin and E_i is the expected frequency in each bin, which is calculated as follows:

$$E_i = n\left[\hat{F}\left(X_u\right) - \hat{F}\left(X_l\right)\right] \tag{1.26}$$

where n is the sample size, $\hat{F}\left(X_u\right)$ is the estimated cumulative probability calculated at the upper limit of the ith bin, and $\hat{F}\left(X_l\right)$ is the estimated cumulative probability calculated at the lower limit of the *i*th bin. The estimated cumulative distribution function $\hat{F}\left(X\right)$ is in the true form of the cumulative distribution function with parameters estimated as discussed earlier.

To do hypothesis testing, this statistic asymptotically follows a χ^2 distribution, provided that each expected count is greater than five. Because the proposed function does not do any hypothesis testing, the expected frequencies may be less than five. However, because the function is computing this statistic for all the ten distributions, the only concern is the value of the test statistic.

For added efficiency, the number of bins selected is the following quantity, provided by the NIST e-Handbook of Statistical Methods (2003), rounded to the nearest whole number: $k = 2n^{2/5} + 1$. This number is then used to arrive at k quantiles to serve as the upper and lower limits of each range such that the probability of observing a value within each range is equal to all other individual ranges. In other words,

$$P[F(X_{ui})-F(X_{li})] = P[F(X_{uj})-F(X_{lj})] \ \forall \ i \neq j, i,j = 1,\dots,k. \quad (1.27)$$

Because there are (k-1) bins and each bin contains equal probability, the expected frequencies for each bin is simply $E_i = n/(k-1)$. The number of observations in each range is then computed, and Equation 1.25 is used to calculate the goodness of fit statistic.

1.3.3 Anderson-Darling Statistic

Anderson and Darling (1952) proposed a statistic which tested if data follow a particular cumulative distribution function. Hypothesis testing can be done with this statistic; however, the statistic itself must be scaled by a factor dependent on what distribution is being tested (Stephens, 1974). Because no hypothesis testing is being done by the function, this scaling factor is unimportant, which makes this statistic appealing to give added information for determining candidate distributions.

Let $X_{(1)}, \dots, X_{(n)}$ denote the order statistics from a random sample with cumulative distribution function *F*. The Anderson-Darling statistic is then calculated as follows:

$$A^2 = -n - \sum_{i=1}^{n} \frac{2i-1}{n}\left[\ln \hat{F}\left(X_{(i)}\right) + \ln\left(1 - \hat{F}\left(X_{(n+1-i)}\right)\right)\right]. \quad (1.28)$$

Again, the estimated cumulative distribution function $\hat{F}(X)$ is estimated by using the true form of the CDF using parameter estimation. If data follow the hypothesized distribution, the value of this statistic should be small.

1.3.4 Maximum Distance

The maximum distance is defined as the maximum difference between an observed value and its theoretical value according to a specified distribution. Often, the difference between the theoretical values and the observed values is greatest in the tails of the distribution. It is likely that two or more distributions may appear appropriate, and the maximum distance measure can maybe provide a little more insight into the actual distribution to be used.

1.4 Interval Estimation

It is well known that the maximum likelihood estimators of the parameters of distributions are asymptotically normally distributed if certain conditions are met. Because maximum likelihood estimators have random error and because the concordance correlation is sensitive to shifts in location and scale, it is appropriate to provide an interval estimate for the true values of the parameters of all the distributions. The proposed function calculates an interval estimate for all the parameters. For all distributions except the Pareto distribution, uniform distribution, and the Laplace distribution, the interval is centered around the maximum likelihood estimate with the upper and lower bounds of the interval being one standard error away from the maximum likelihood estimate. It then selects nine values of equal spacing within the interval for each estimate. To illustrate this, think of a nine by nine grid. The columns represent the nine estimates for the first parameter, and the rows represent the nine estimates for the second parameter. Each cell of this grid now contains a unique combination of the two parameter estimates. The function then selects the combination of parameter estimates that produces the largest concordance correlation between the theoretical quantiles and the observed quantiles. The exponential distribution only has one parameter, so the grid illustration does not apply. Because the interval is two standard errors wide, this is approximately a 68% confidence interval for each of the parameters.

There is a function in R called `fitdistr`. This function obtains the maximum likelihood estimates for all the distributions except the uniform and Pareto distributions. It also obtains the standard errors based on Fisher's Information. For the distributions where the maximum likelihood estimators are not in closed form, it employs numerical techniques to arrive at the estimates by using the function `optim`. The default method in `optim` uses the numerical technique introduced by Nelder and Mead (1965), but `fitdistr` uses other numerical techniques if the Nelder-Mead technique is not suitable. The function `fitdistr` was used to obtain the maximum likelihood estimates and interval estimates for all the parameters of all the distributions except for those of the Pareto distribution, uniform distribution, and the location parameter of the Laplace distribution. The following sections discuss how interval estimates were obtained for the aforementioned parameters.

1.4.1 Interval Estimation of the Parameters of the Uniform Distribution

The maximum likelihood estimators of the parameters of the uniform distribution are the sample minimum and the sample maximum. The parameter a can never be larger than the sample minimum, so the sample minimum is the upper bound of the interval estimate for the parameter a. Likewise, the sample maximum is the lower bound of the interval estimate for the parameter b.

To obtain the interval estimate, the variance of the point estimate must be calculated. Because the maximum likelihood estimates are the sample minimum and maximum, the distribution of the minimum and the maximum can be easily obtained. The distribution for the sample minimum is as follows:

$$f\left(x_{(1)} \mid a,b\right) = \frac{n}{(b-a)^n}\left(b - x_{(1)}\right)^{n-1} I\left(a \le x_{(1)} \le b\right) . \qquad (1.30)$$

The distribution for the sample maximum is as follows:

$$f\left(x_{(n)} \mid a,b\right) = \frac{n}{(b-a)^n}\left(x_{(n)} - a\right)^{n-1} I\left(a \le x_{(n)} \le b\right) . \qquad (1.31)$$

The variances of the distributions of the sample minimum and the sample maximum are equal:

$$Var\left(X_{(1)}\right) = Var\left(X_{(n)}\right) = \frac{n(b-a)^2}{(n+1)^2(n+2)} . \qquad (1.32)$$

The variance is then estimated by replacing the parameters with the maximum likelihood estimates. The interval estimates for the parameters a and b are two standard errors wide, which is consistent with the interval estimates of the parameters of the other distributions.

1.4.2 Interval Estimation of the Parameters of the Pareto Distribution

Because all observations from a Pareto distribution must be greater than the location parameter α, the maximum likelihood estimator is the sample minimum. It is also the upper bound for the interval estimate. The distribution of the sample minimum is as follows:

$$f\left(x_{(1)} \mid \alpha,\beta\right) = \frac{n\beta\alpha^{\beta}}{x_{(1)}^{\beta+1}}\left(\frac{\alpha}{x_{(1)}}\right)^{\beta(n-1)} I\left(x_{(1)} > \alpha > 0\right) \quad (1.33)$$

From this, the variance of the sample minimum is as follows:

$$Var\left(X_{(1)}\right) = \alpha^2\beta n\left(\frac{1}{\beta n - 2} - \frac{\beta n}{\left(\beta n - 1\right)^2}\right) \quad (1.34)$$

for βn > 2. The variance is then estimated by substituting the estimates for the parameters with the maximum likelihood estimates. The interval estimate for the parameter α is two standard errors wide.

Krishnamoorthy (2006) provided a formula for a confidence interval for β:

$$\left(\frac{\hat{\beta}}{2n}\chi^2_{2(n-1),.32/2}, \frac{\hat{\beta}}{2n}\chi^2_{2(n-1),1-.32/2}\right) \quad (1.35)$$

Because the other interval estimates are approximately 68% confidence intervals, I chose 0.32 to be my level of significance as can be seen in the above formula.

1.4.3 Interval Estimation of the Paramaters of the Laplace Distribution

The `fitdistr` function in R can obtain a maximum likelihood estimate and standard error of the scale parameter σ. The standard error of the location parameter estimate, however, is more difficult to obtain. Because the shape of the distribution is centered at a point of nondifferentiability, the Fisher Information matrix estimates the standard error to be zero, so Fisher's Information is inappropriate to calculate the standard errors of the location parameter. With an odd sample size, the distribution of the median is somewhat straightforward to obtain. With an even sample size, however, the joint distribution of the two middle order statistics must be obtained, which is more difficult to derive. Asrabadi (1985) examined the distribution of the median for an even sample size, however, when the formula was used in R, negative variance estimates were sometimes computed. Also, there were discrepancies in the formulas provided in Asrabadi (1985) and Johnson, et. al. (1995). A nonparametric confidence interval was used instead. Let $X_{(1)},\ldots,X_{(n)}$ be the ordered statistics of an observed sample. The approximate 68% confidence interval for the median is the interval:

$$\left(X_{(q-1)}, X_{(n-q)}\right), \tag{1.36}$$

such that *q* is the largest integer such that

$$P\left(X \le q\right) = \sum_{i=0}^{q} \binom{n}{i} (0.5)^{n} \le 0.32/2 \; . \tag{1.37}$$

Eight equidistant values were obtained from this interval, and the sample median was added to this sequence so that nine values can be used as the location parameter estimates.

CHAPTER 2 - The Function and Simulation Results

2.1 The Function `diagnostic`

The proposed R function is only suitable for univariate data. Using maximum likelihood estimation, the function, which is called `diagnostic,` estimates the parameters for each of the aforementioned distributions. It then calculates interval estimates of each parameter of each distribution, selecting a grid of 81 possible combinations of estimates for the two parameters, except for the exponential distribution, which only has one parameter. Theoretical quantiles are computed for each distribution using each of the 81 (or 9 for the exponential distribution) combinations of estimated parameter values. For each combination, the concordance correlation is computed using Equation (1.23) with the theoretical quantiles as the first sample and the observed ordered data as the second sample. This results in 81 different concordance correlations for each distribution. The exponential distribution, only having one parameter, has nine concordance correlations. For all ten distributions, the maximum concordance correlation is obtained, along with the parameter estimates that yielded this value. This information is then stored. Using these parameter estimates, the goodness of fit statistic, Anderson-Darling statistic, and the maximum distance measure is calculated for each distribution.

The function then uses all of this information to rank the distributions. If data follow a particular distribution, the concordance correlation should be large, and the goodness of fit statistic, Anderson-Darling statistic, and the maximum distance should be small. The function ranks each distribution according to each measure. The distribution that yielded the highest concordance correlation receives a one. The distribution with the next highest receives a two, and so on. Then, the distribution with the lowest goodness of fit measure receives a one, and so on. A similar process follows for the Anderson-Darling statistic and the maximum distance. The sum of these four rankings for each distribution is then calculated. The function then sorts the distributions according to the sum of the ranks. The lowest possible sum can be four.

Figure 2.1 An Example of the Input and Output of `diagnostic`

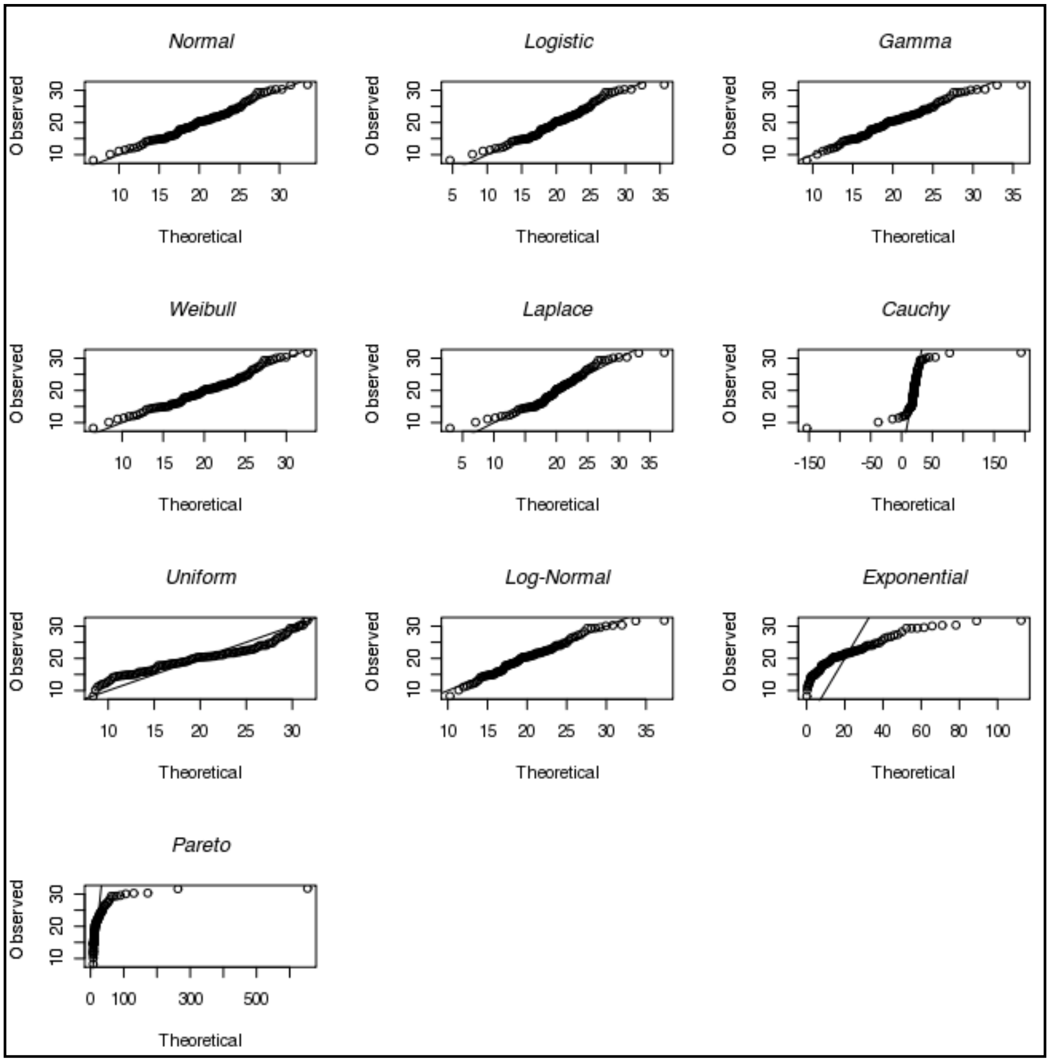
```
> set.seed(6412)
> x=rnorm(100,20,5)
> diagnostic(x)
            Par1       Par2      Con.Corr   Good-Fit A2        Max.dist Sum.Rank
Normal      20.13954   5.185571  0.9927978  12.56    0.4637399 2.104595 10
Gamma       15.01381   0.7459297 0.9921926  11.6     0.3496805 4.229029 10
Weibull     4.313711   22.13904  0.9903245  11.84    0.6943246 2.070418 12
Logistic    20.11907   2.927647  0.9864621  12.32    0.5275203 3.843783 16
Lognormal   2.974558   0.2503853 0.9865312  10.88    0.6189546 5.546976 16
Laplace     20.12635   3.721872  0.9707292  20.72    1.278759  5.493993 24
Uniform     8.260267   31.77221  0.9455263  43       4.398278  3.593482 26
Cauchy      20.08193   2.729113  0.2271609  32       1.369016  162.0362 31
Pareto      8.116331   1.207247  0.07401306 177.68   23.49063  621.92   37
Exponential 0.04717089 NA        0.0508949  188.96   70.27247  80.54955 38
Warning message:
In dgamma(x, shape, scale, log) : NaNs produced
```

The function outputs three windows. The output in the R console is a matrix that has the names of the distributions as its row names and the statistical measures as the column names. It is a ten by seven matrix. The first two columns are the parameter estimates. The third column is the concordance correlation. The fourth column is the goodness of fit measure. The fifth column is the Anderson-Darling statistic. The sixth column is the maximum distance. Lastly, the seventh column is the sum of the ranks.

Two graphics windows are then outputted. The first graphics window contains ten QQ plots for each of the ten distributions. The second graphics window contains ten histograms of the data, each with the probability density curve for each distribution superimposed upon the histogram, using the parameter estimates generated from the function.

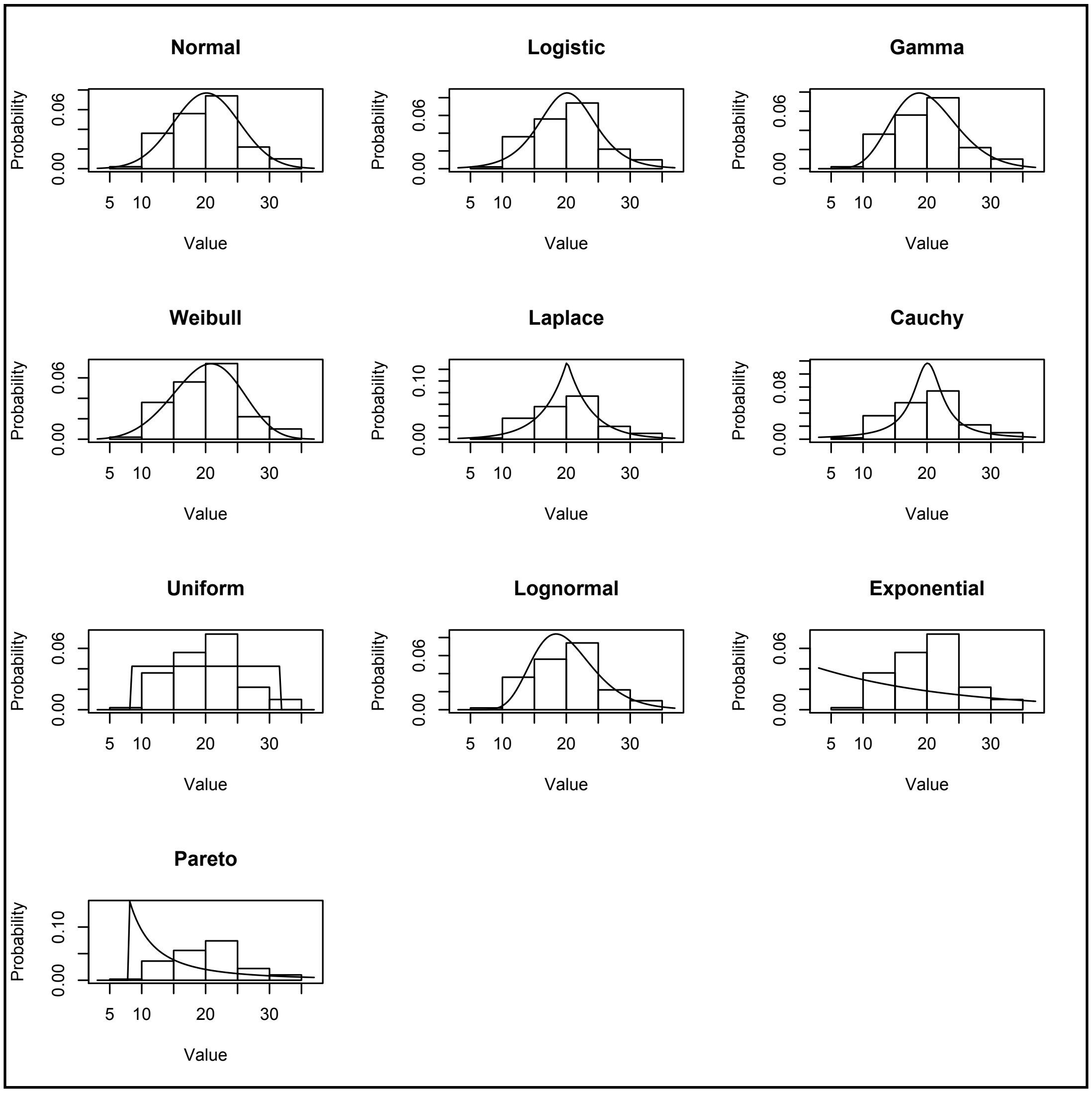
For example, a sample of 100 observations were randomly selected from a normal population with a mean of 20 and a standard deviation of 5. The input and the output are shown in Figures 2.1, 2.2, and 2.3.

Figure 2.2 QQ Plots in Graphical Output



The QQ plots in Figure 2.2 yielded some interesting results. It is quite obvious that the exponental distribution and the Pareto distribution are poor distributions to model these data. However, many distributions appear to follow the 45 degree angle line through the origin. The normal plot, the Weibull plot, and the gamma plot look nearly identical. This may seem unusual because the Weibull and gamma distributions are skewed. It is noteworthy that even though the Weibull and gamma distributions are skewed, the density curves look nearly normal, as can be seen in Figure 2.3. Even the skew of the lognormal distribution is not pronounced (Figure 2.3). In this particular example, the function ranked the normal distribution and the gamma distribution first. The parameter estimates for the normal distribution were close to the true

Figure 2.3 Histograms with Superimposed Density Curves



values of the parameters. The gamma distribution provided the lowest goodness of fit statistic (Figure 2.1); however, the normal distribution had the highest concordance correlation.

2.2 Simulation Results and Discussion

To examine the effectiveness of the function and the reliability of all four statistical measures, a simulation study was done. A thousand different random samples were generated for sample sizes of 20 and 50 for each of the following distributions: Normal(25,5); Laplace(17,1); Logistic(18,1); Gamma(20,2); Gamma(2,1); and Weibull(4,20). In one simulation, the concordance correlation was the only measure used to identify the candidate distributions. In each of the thousand iterations, the distribution that yielded the maximum concordance correlation was recorded. Then, the simulation procedure counted the number of times each distribution yielded the maximum concordance correlation. The counts are represented by “CC” in each subsequent table.

A similar process was also done using the sum of the ranks of only three measures: the concordance correlation, goodness of fit, and maximum distance. “Rank.3” represents when the function estimated the parameters of each distribution such that the maximum concordance correlation was achieved. It then used those parameter estimates to calculate the goodness of fit statistic and the maximum distance.

The third type of simulation was procedurally done the same as the process indicated by “Rank.3,” except the Anderson-Darling statistic was also included in the rankings. This is represented by “Rank.4.”

When adding the frequencies, the sum of the frequencies for the concordance correlation is 1000 because there were no ties. The sum of the frequencies for each distribution yielded by the procedures indicated by “Rank.3” and “Rank.4” will likely sum to more than 1000 because the ranking procedure may pick more than one “best” distribution. An example of this is shown in Figure 2.1 where the normal and the gamma distribution had the same sum of ranks.

2.2.1 Normal(25,5)

Table 2.1 shows the results of the simulation for the Normal(25,5) case. The numbers in each cell represent the counts of the number of times the respective measure selected the corresponding distribution. In the case where n = 20, the concordance correlation was able to select the normal distribution 97 times whereas the ranking procedure without the Anderson-Darling statistic selected the normal distribution 179 times. The ranking procedure using all four measures selected the normal distribution 186 times. The Weibull distribution was counted the most, but as can be seen in Figure 2.3, the Weibull distribution can look very similar to the normal distribution. Because this is a small sample size, we can expect the sample data to be slightly skewed, even though the data follow a symmetric distribution. Because the Weibull distribution always has a right-skew and can also look symmetric, the Weibull can provide a

Table 2.1 Simulation Results for Normal(25,5)

n=20 Norm(25,5)					n=50 Norm(25,5)			
	CC	Rank.3	Rank.4			CC	Rank.3	Rank.4
Normal	97	179	186		Normal	309	370	355
Exponential	0	0	0		Exponential	0	0	0
Gamma	148	182	161		Gamma	168	195	187
Logistic	57	107	92		Logistic	118	139	149
Lognormal	177	188	189		Lognormal	86	90	87
Weibull	297	264	284		Weibull	283	262	267
Cauchy	1	0	1		Cauchy	0	0	0
Laplace	89	85	92		Laplace	25	28	28
Uniform	134	107	57		Uniform	11	10	7
Pareto	0	0	0		Pareto	0	0	0

better fit for the sample data. A similar argument can be made with the gamma distribution and the lognormal distribution.

In the case where n=50, the normal distribution was selected with the highest frequency, however was only selected 37 percent of the time using the three measures instead of four. When the Anderson-Darling was added, the normal distribution actually was counted less.

It is necessary to note that the ranking ranges from 1 to 10. It is likely that two or more measures are close together, but the rankings do not account for this. Just because a particular distribution yields a higher concordance correlation or a lower goodness of fit statistic does not mean that the sample data unequivocally came from that distribution because another distribution can yield a similar measure. For example, in Figure 2.1, the minimum goodness of fit measure was produced for the gamma distribution at 11.6, but the goodness of fit statistic for the normal distribution was very close at 12.56, even if it ranked fifth in the goodness of fit.

Table 2.2 Simulation Results for Laplace(17,1)

n=20 Laplace(17,1)					n=50 Laplace(17,1)			
	CC	Rank.3	Rank.4			CC	Rank.3	Rank.4
Normal	57	93	79		Normal	50	72	54
Exponential	0	0	0		Exponential	0	0	0
Gamma	19	89	46		Gamma	13	36	23
Logistic	72	141	125		Logistic	137	242	218
Lognormal	195	214	195		Lognormal	117	124	104
Weibull	256	223	225		Weibull	143	130	107
Cauchy	44	37	37		Cauchy	1	1	1
Laplace	298	298	323		Laplace	539	487	543
Uniform	29	19	10		Uniform	0	0	0
Pareto	30	13	10		Pareto	0	0	0

2.2.2 Laplace(17,1)

Table 2.2 shows the results for the Laplace(17,1) case. For both sample sizes, the Laplace distribution was counted with the highest frequency, but the Weibull, logistic, and lognormal distributions were also counted with high frequency. This may be expected because the logistic distribution has slightly heavy tails, much like the tails of the Laplace distribution. When all four measures were used, the Laplace distribution was represented with the highest frequency. However, the Laplace distribution was only selected for about half of the samples of size n = 50 and even less for n = 25.

When the sample size increases to n = 50, the logistic and the Weibull distributions were selected approximately the same number of times using only the concordance correlation. When other information was used to arrive at the candidate distribution, the logistic distribution was represented more. The ranking procedure performed worse than merely using the concordance correlation when the sample size was n = 50, but when the Anderson-Darling statistic was

Table 2.3 Simulation Results for Logistic(18,1)

n=20 Logis(18,1)					n=50 Logis(18,1)			
	CC	Rank.3	Rank.4			CC	Rank.3	Rank.4
Normal	112	184	167		Normal	184	217	198
Exponential	0	0	0		Exponential	0	0	0
Gamma	33	109	72		Gamma	53	109	87
Logistic	75	134	120		Logistic	199	247	269
Lognormal	272	250	248		Lognormal	235	217	212
Weibull	263	222	237		Weibull	167	140	141
Cauchy	12	11	11		Cauchy	0	0	0
Laplace	147	142	162		Laplace	159	164	168
Uniform	74	59	38		Uniform	3	4	1
Pareto	12	6	4		Pareto	0	0	0

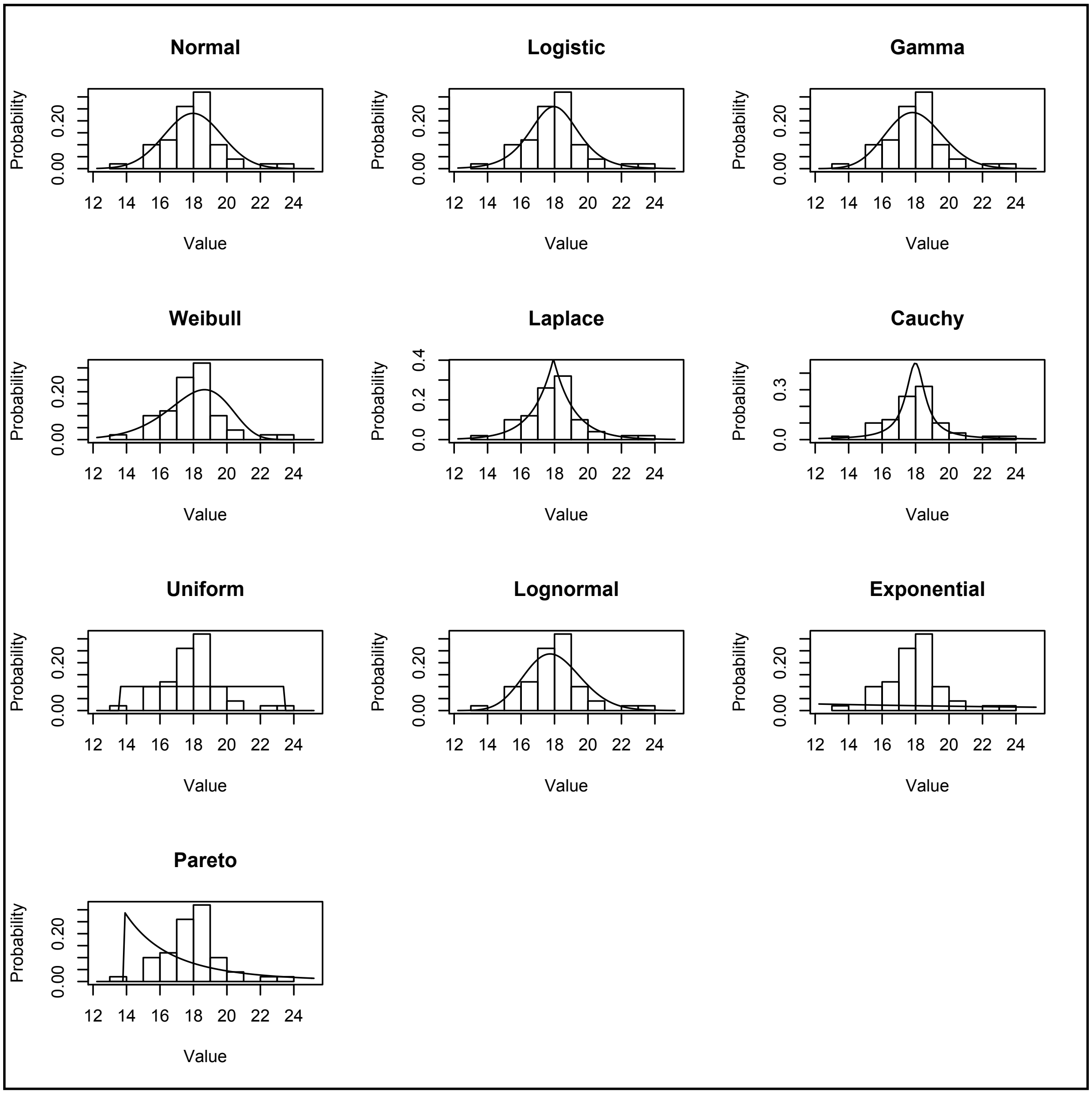
included, the procedure outperformed all other processes.

2.2.3 Logistic(18,1)

Table 2.3 shows the results for a Logistic(18,1) distribution. For a small sample, all three procedures performed poorly in identifying the logistic distribution. Again, the Weibull and lognormal were selected with approximately the same frequency. Even when the sample size is increased to 50 observations, the logistic distribution was only selected approximately 20 percent of the time for the concordance correlation. The ranking procedures, though, identified the logistic distribution with more success. Because the logistic distribution has heavier tails than the normal distribution, the probability of observing values far away from the mean is higher. This could result in a skewed-looking sample, for which distributions like the Weibull and lognormal could provide a better fit to the sample data.

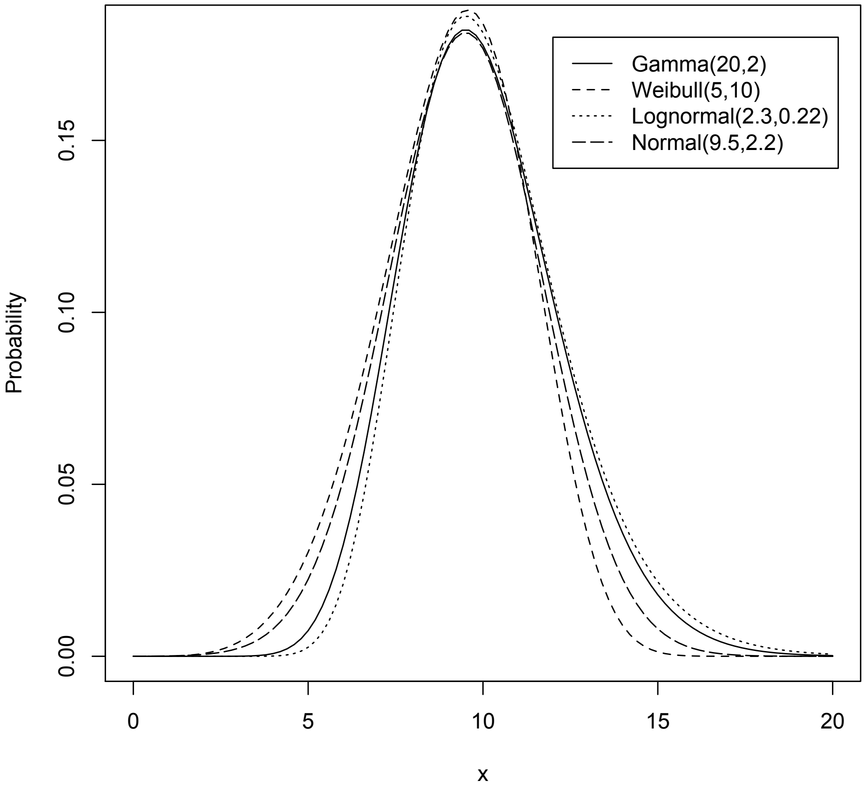
A random sample of 50 observations was taken from a Logistic(18,1) distribution and the proposed function was run. In this case, the function used the ranking procedure with all four

Figure 2.4 Graphical Output from a Sample Taken from Logistic(18,1)



measures to arrive at the candidate distributions. Figure 2.4 provides the histograms with the superimposed distributions. The function estimated the parameters of the logistic distribution to be 17.97 and 0.96, which are very close to the true values of the parameters. The scale parameter of the lognormal distribution is estimated to be quite low at 0.09. When this occurs, the distribution looks nearly symmetric (Figure 2.4). In this particular case, the function selected the Laplace distribution to be the first candidate distribution and the logistic and lognormal

Figure 2.5 Gamma, Weibull, Lognormal, and Normal Distribution Density Curves



distributions were selected second and third, respectively.

2.2.4 Gamma(20,2)

This particular distribution looks nearly symmetric. Figure 2.5 provides the probability density curves of four different distributions. The solid line represents the Gamma(20,2) distribution, the dashed line represents the Weibull(5,10) distribution, the dotted line represents the Lognormal(2.3,0.22) distribution, and the long-dashed line represents the Normal(9.5,2.2) distribution. These four density curves look very similar. A sample drawn from a Gamma(20,2) distribution can easily look like it came from a lognormal or Weibull or normal distribution, especially for small sample sizes. These three densities also look nearly symmetrical, like the normal distribution.

Table 2.4 shows the simulation results from the Gamma(20,2) distribution. When the

Table 2.4 Simulation Results for Gamma(20,2)

n=20 Gamma(20,2)					n=50 Gamma(20,2)			
	CC	Rank.3	Rank.4			CC	Rank.3	Rank.4
Normal	91	159	164		Normal	209	254	259
Exponential	0	0	0		Exponential	0	0	0
Gamma	161	230	203		Gamma	338	382	357
Logistic	52	75	75		Logistic	42	46	62
Lognormal	379	353	360		Lognormal	335	329	321
Weibull	149	143	161		Weibull	57	69	59
Cauchy	0	0	0		Cauchy	0	0	0
Laplace	46	49	60		Laplace	7	13	19
Uniform	120	98	42		Uniform	12	11	6
Pareto	2	1	1		Pareto	0	0	0

sample size was 20, the lognormal was selected with the highest frequency. The gamma and Weibull distributions were selected with approximately the same frequency. Oddly enough, the uniform distribution was selected with an unusually high frequency, but when the Anderson-Darling was included, that frequency was reduced by a fairly large margin.

When the sample size was increased to 50, the gamma distribution and the lognormal distribution had nearly the same frequencies, but the normal distribution was selected quite often. This might be expected based upon the nearly symmetrical density as can be seen in Figure 2.5.

For both sample sizes, the inclusion of the Anderson-Darling statistic reduced the number of times the gamma distribution was selected. This likely occurred because many of the distributions look similar and the four statistical measures were very close together. The ranking process uses equally spaced measures, but the statistical measures are not. As can be seen in Figure 2.1, about five distributions can have each measures very close together, but when ranked, are forced to be equally spaced.

Table 2.5 Simulation Results for Gamma(2,1)

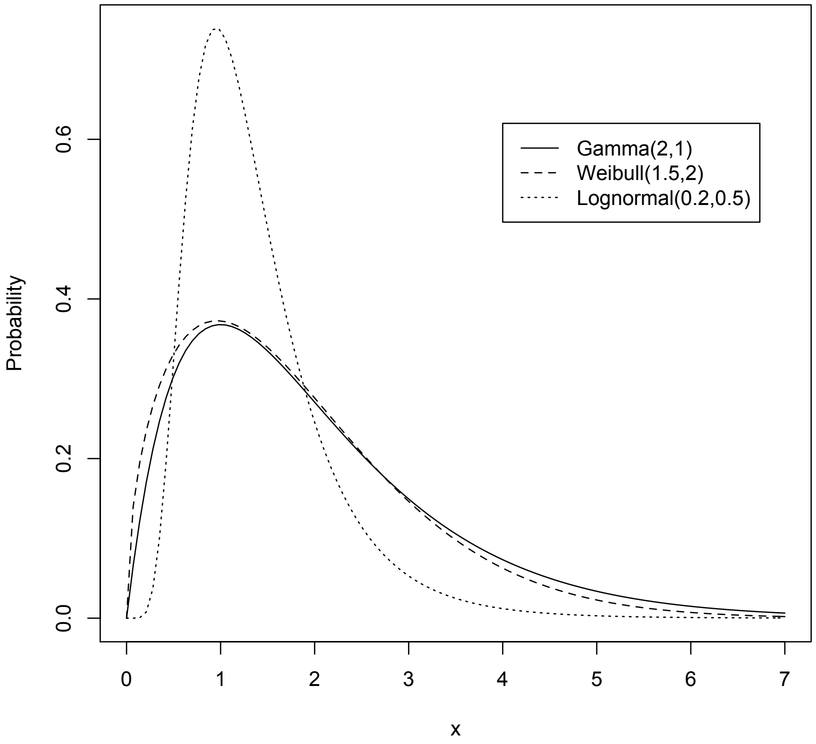
n=20 Gamma(2,1)					n=50 Gamma(2,1)			
	CC	Rank.3	Rank.4			CC	Rank.3	Rank.4
Normal	33	56	77		Normal	3	10	10
Exponential	0	0	0		Exponential	0	0	0
Gamma	176	282	275		Gamma	287	370	440
Logistic	3	7	11		Logistic	0	1	4
Lognormal	232	230	203		Lognormal	169	153	127
Weibull	481	452	441		Weibull	539	555	510
Cauchy	0	0	0		Cauchy	0	0	0
Laplace	3	3	12		Laplace	0	0	1
Uniform	72	69	39		Uniform	2	5	2
Pareto	0	0	0		Pareto	0	0	0

2.2.5 Gamma(2,1)

This distribution is heavily right skewed. The gamma, Weibull, and lognormal are all right skewed distributions. It is obvious, though, that a symmetric distribution cannot appear skewed. The simulation results are consistent with what we might expect when sampling from a Gamma(2,1) distribution. Table 2.5 shows the results.

The gamma, Weibull, and lognormal distributions had high frequencies, whereas the symmetric distributions and highly skewed distributions were not represented with high frequency. The ranking procedures selected the gamma distribution similarly when the sample was size n = 20. When the sample size increased to n = 50, the ranking procedure using the four measures selected the gamma distribution 70 more times than when only using three measures.

Figure 2.6 Gamma, Weibull, and Lognormal Distribution Density Curves II



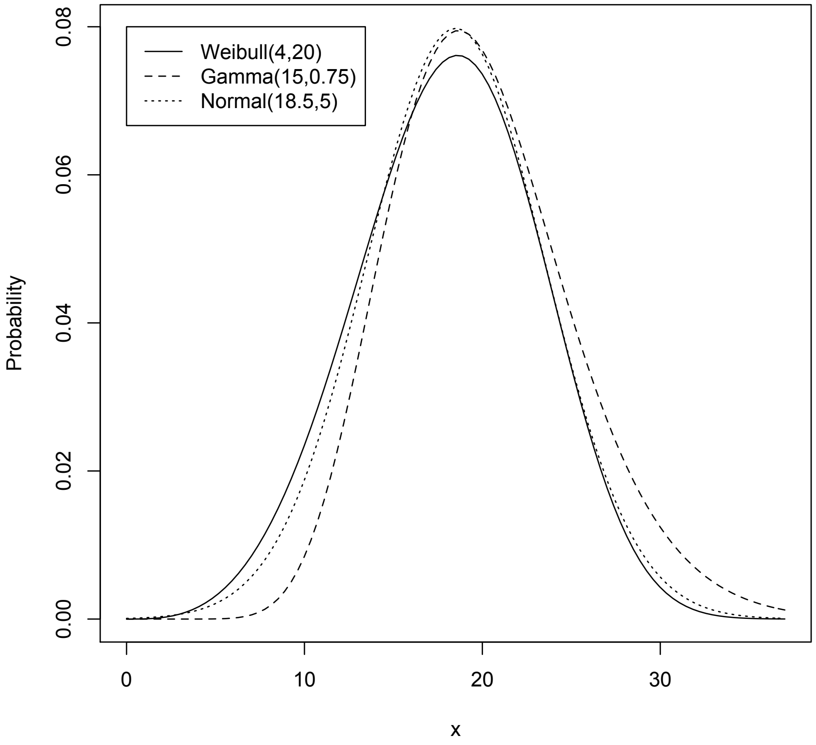
The Weibull distribution is counted with high frequency, however if the parameters are carefully selected, the Weibull distribution can look nearly identical to the Gamma distribution. In Figure 2.6, the Gamma(2,1) density is represented by a solid line, the Weibull (1.5,2) density is represented by a dashed line, and the Lognormal(0.2,0.5) density is represented by a dotted line.

The lognormal has more mass closer to zero; however, this discrepancy may go unnoticed. Also, as can easily be seen, the gamma and Weibull densities are very similar.

2.2.6 Weibull(4,20)

This particular distribution is another case where even though the density is right skewed, it looks nearly symmetric. In Figure 2.7 below, the solid line represents a Weibull(4,20) density, the dashed line represents a Gamma(15,0.75) density, and the dotted line represents a Normal (18.5,5) density. As can be seen, all three densities can be mistaken for one another. The

Figure 2.7 Weibull, Gamma, and Normal Distribution Density Curves



simulations results, though, had a high frequency of times the three procedures selected the Weibull distribution, as is indicated by Table 2.6. However, the function still only chose the correct distribution half of the time.

When the sample size was 20, the Weibull distribution was selected with the highest frequency. The normal and gamma were also selected with high frequency. The normal and gamma distributions are expected to be fairly high, but the uniform distribution had an uncharacteristically high representation. This is probably due to those particular samples having numerous observations close to the center of the density with smaller variability. Given a small sample size, this can be expected.

Table 2.6 Simulation Results for Weibull(4,20)

n=20 Weibull(4,20)					n=50 Weibull(4,20)			
	CC	Rank.3	Rank.4			CC	Rank.3	Rank.4
Normal	110	208	190		Normal	193	278	269
Exponential	0	0	0		Exponential	0	0	0
Gamma	141	144	146		Gamma	102	110	106
Logistic	83	121	111		Logistic	90	100	109
Lognormal	74	86	77		Lognormal	14	24	18
Weibull	357	374	404		Weibull	559	554	556
Cauchy	0	0	0		Cauchy	0	0	0
Laplace	73	66	71		Laplace	12	14	15
Uniform	162	121	64		Uniform	30	23	11
Pareto	0	0	0		Pareto	0	0	0

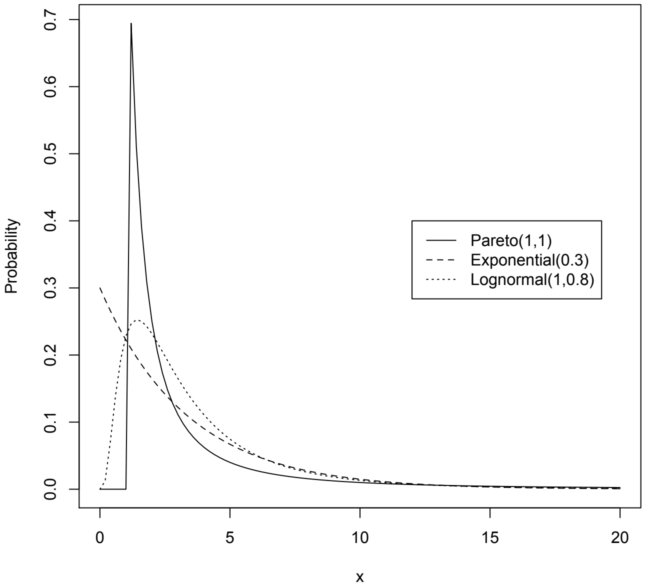
When the sample size increased to 50, the concordance correlation and the ranking procedures performed similarly, except for the ranking procedures also selected the normal distribution with a higher frequency.

The Weibull distribution can take on many shapes which can fit the sample data quite well. Even though the population probability density function of the Weibull(4,20) can look very similar to other distributions, the estimated density curve could take on a minutely different shape such that the density curve fits the sample better. This is why the Weibull distribution was represented highly in the other simulations.

2.2.7 Pareto(1,1) and Pareto(0.1,1)

Simulations using the Pareto(1,1) and Pareto(0.1,1) distributions were attempted, but were not successful. Because these are heavily skewed distribution, some of the other distributions do not provide good fits to the data. In some cases, if this occurs, the `fitdistr`

Figure 2.8 Pareto, Exponential, and Lognormal Distribution Density Curves



function cannot arrive at estimates for the maximum likelihood estimates. Within the `fitdistr` function, the function `optim` is used, which employs a numerical technique to arrive at maximum likelihood estimates. In these numerical techniques, starting values need to be used. In some cases, starting values are not supplied, so the function `fitdistr` automatically selects starting values. However, according to the R documentation, these starting values may not be suitable, especially if the fitted distribution does not provide a good fit. When this occurs, the optimization can fail. In the process of a thousand iterations, the simulation ceased when one iteration had the aforementioned situation occur.

The Pareto distribution is a unique distribution, but the exponential distribution and the lognormal distribution can look similar. In Figure 2.8, the solid line represents the Pareto(1,1) density, the dashed line represents the Exponential(0.3) density, and the dotted line represents the Lognormal(1,0.8) density.

A random sample of 20 observations were generated from a Pareto(1,1) distribution and the function `diagnostic` was run. Figure 2.9, 2.10, and 2.11 provide the output. The function was able to select the Pareto distribution with the parameter estimates fairly close to the true

Figure 2.9 Output for Pareto(1,1)

```
            Par1       Par2      Con.Corr  Good-Fit A2        Max.dist Sum.Rank
Pareto      1.082578   1.224387  0.976729  1        0.1854383 4.031491 5
Lognormal   1.025260   0.8835974 0.907857  7.6      2.137349  4.786667 12
Weibull     0.8770383  3.826469  0.9159424 16       2.277675  3.752983 13
Gamma       0.9638904  0.2486523 0.8774057 14.2     2.241942  6.012214 17
Cauchy      1.883112   0.6620879 0.6314331 7        0.9897855 10.24469 20
Normal      3.55726    4.85301   0.7173243 23.2     3.852603  7.471438 27
Logistic    3.068217   2.083478  0.7025962 22.6     3.258298  9.839254 27
Exponential 0.2496856  NA        0.2968301 17.8     29.10004  5.766325 30
Laplace     2.042172   2.61159   0.6725209 31       2.48352   10.67463 31
Uniform     -0.6475988 20.54042  0.373725  50.8     15.23495  12.82789 38
```

Figure 2.10 QQ Plot in Graphical Output for Pareto(1,1)

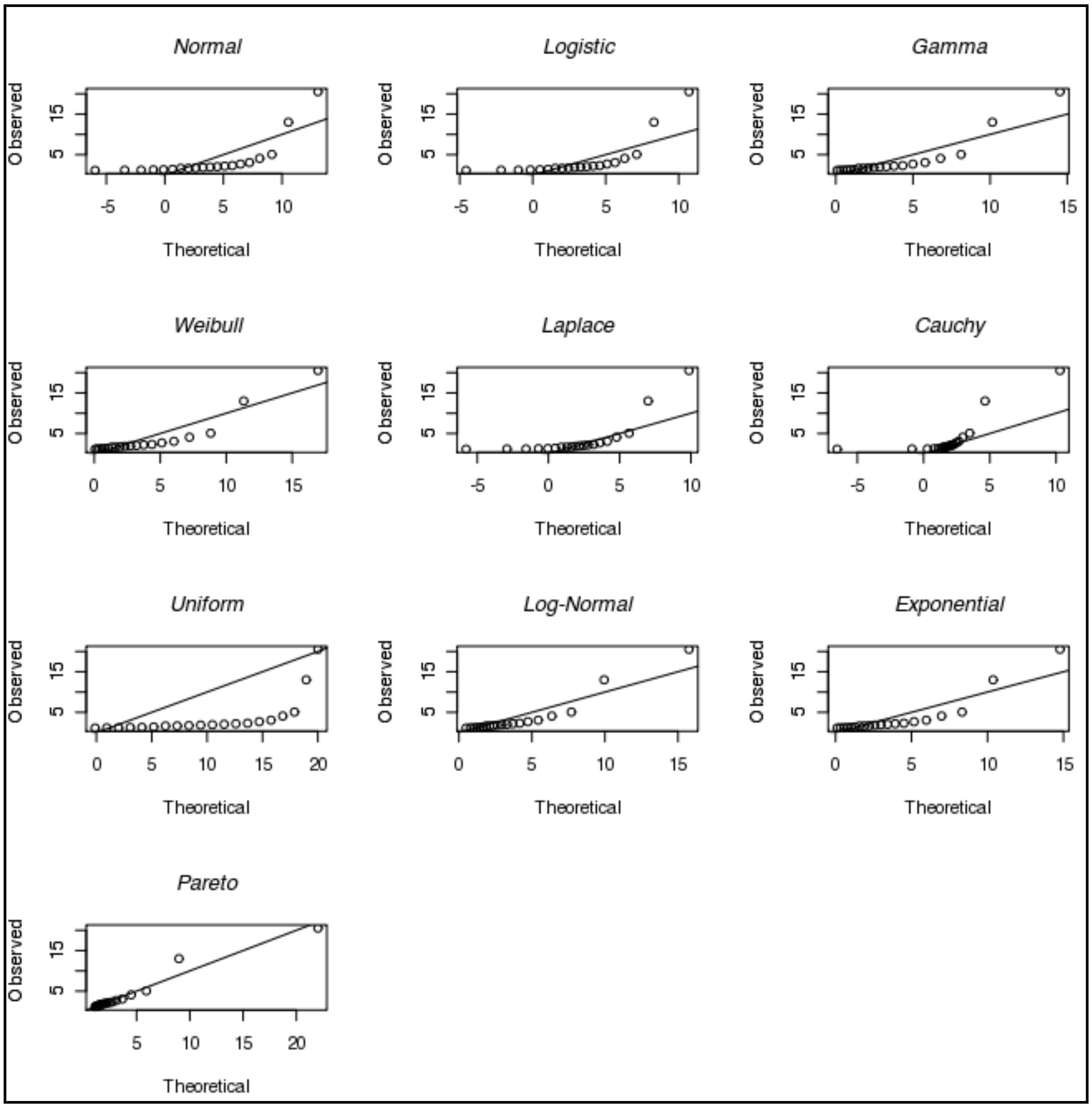
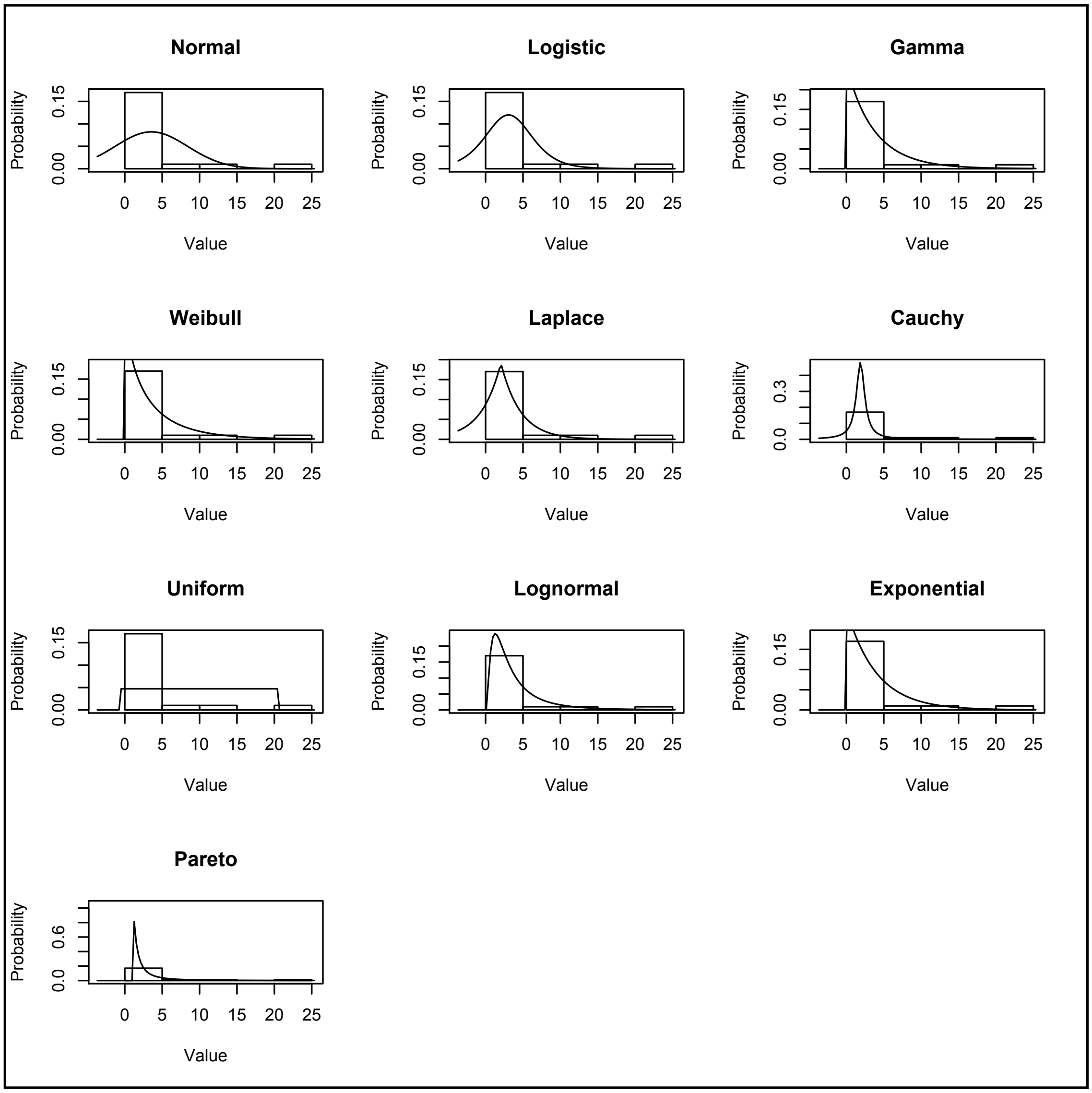


Figure 2.11 Histograms in Graphical Output for Pareto(1,1)



values of the parameters (Figure 2.9). From the QQ plots (Figure 2.10), it is quite obvious that the Pareto distribution displays the straightest line.

CHAPTER 3 - A Real Data Example, Limitations, and Conclusion

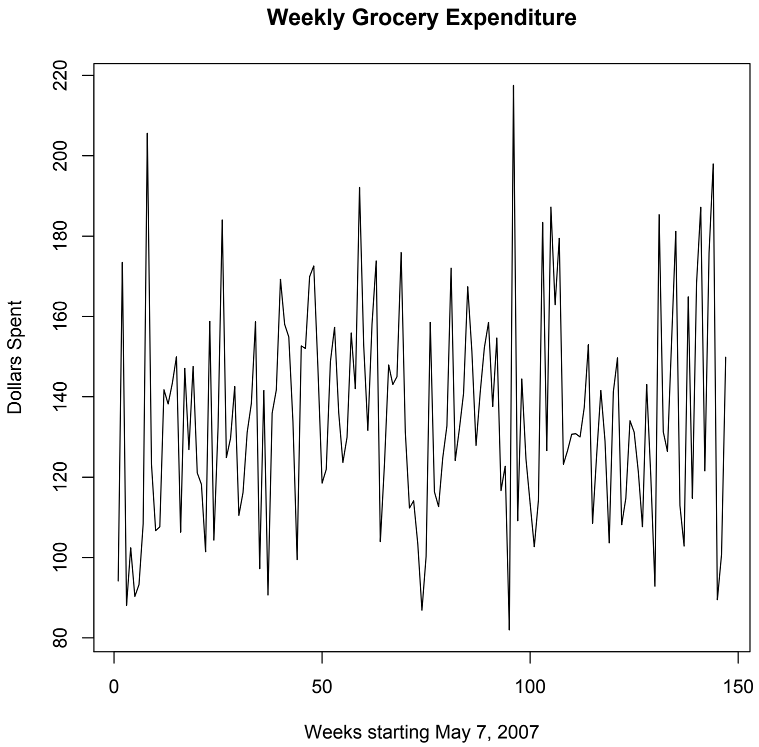
3.1 Weekly Grocery Expenditure of Richards Household

I have records of all the purchases and deposits in our bank account since May 2007. Every week we go grocery shopping, so those transactions have been recorded for a total of 147 grocery shopping trips. The function `diagnostic` was used to arrive at a candidate distribution to help model this particular data set. These data might have some dependence because it is time series data; however, looking at the time series graph (Figure 3.1), an autocorrelation function plot (Figure 3.2), and a partial autocorrelation plot (Figure 3.3), an argument can be made that these time series data are coming from a stationary process. A histogram of the data (Figure 3.4) indicates a slight right skew.

The mean weekly grocery expenditure was \$135.20 with a standard deviation of \$27.06. The range of the expenditures was from \$81.97 to \$217.51. The function `diagnostic` generated output that can be seen in Figures 3.5, 3.6, and 3.7. The lowest possible value of the sum of the ranks is four, and that is what value the gamma distribution produced. The QQ plot for the gamma looks very straight (Figure 3.6) and the density curve seems to fit the histogram quite adequately (Figure 3.7).

Given the large sample size, the goodness-of-fit statistic and the Anderson-Darling statistic should be valid for hypothesis testing. Larsen and Marx (1986) offer the criteria to do hypothesis testing with the goodness-of-fit statistic. The null hypothesis is defined as the data came from a specific distribution. The alternative hypothesis is defined as the data did not come from a specific distribution. Under the null hypothesis, the goodness-of-fit statistic should follow a chi-square distribution with $k-1-r$ degrees of freedom, where k is the number of bins and r is the number of parameters that were estimated for that distribution. The number of bins used for this sample size was 14. For all distributions, except for the exponential, the degrees of freedom are 14 - 1 - 2 = 11. The critical value for a size $\alpha = 0.10$ test is 17.28. Any test statistic value that is greater than this value results in a rejection of the null hypothesis. There are five values that are less than this critical value, which means any of those five values would lead to a failure to

Figure 3.1 Time Series Plot of Weekly Grocery Expenditures



reject the null hypothesis for each of the respective distributions. One of those distributions is the normal distribution. However, looking at the QQ plot in Figure 3.7, the normal distribution shows some slight curvature in the tails, but the gamma and lognormal does not. Using the Anderson-Darling statistic, the critical value for the normal distribution at level α = 0.10 is 0.632 (Shorak and Wellner, 1986). The test statistic for the normal is below this at 0.541, but I would argue that the gamma distribution provides a better fit.

Figure 3.2 Autocorrelation Function Plot of Weekly Grocery Expenditure

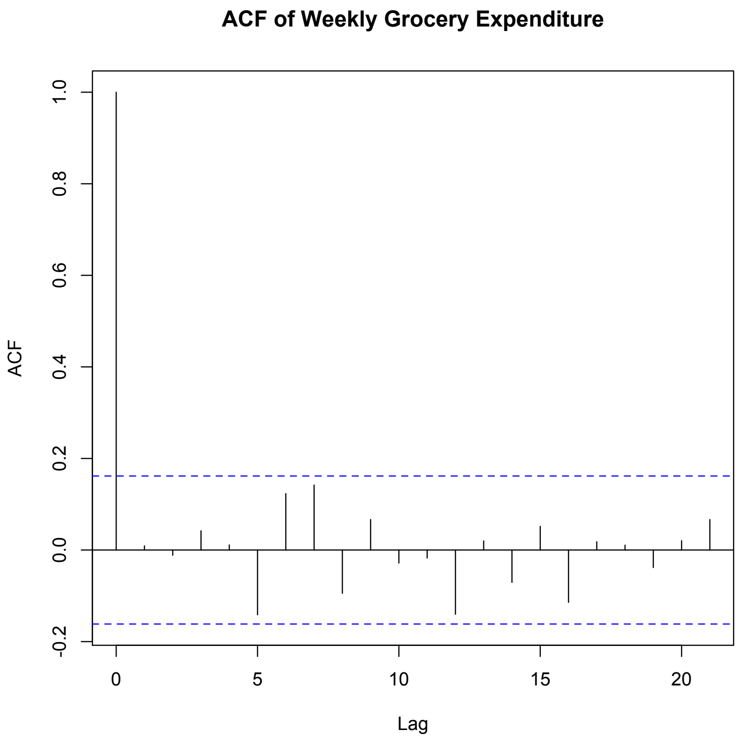


Figure 3.3 Partial Autocorrelation Function Plot of Weekly Grocery Expenditure

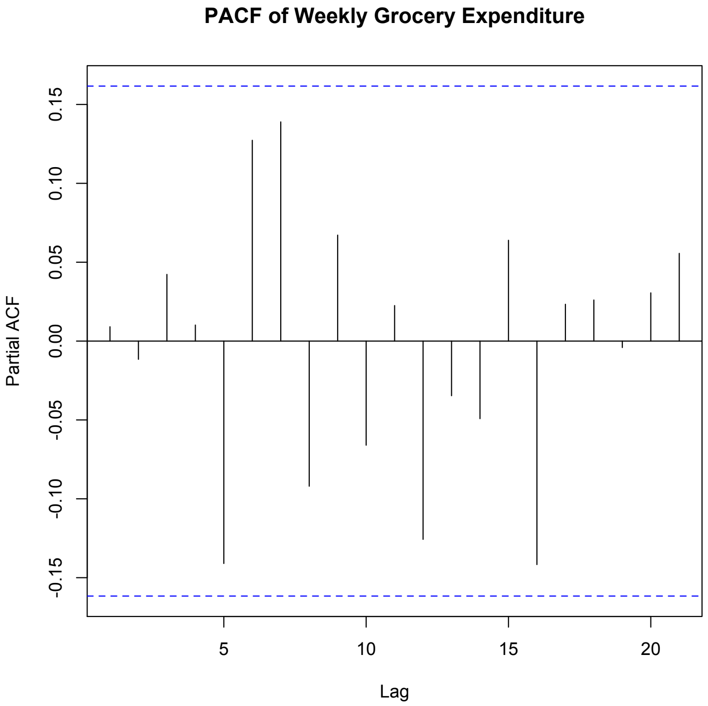


Figure 3.4 Histogram of Weekly Grocery Expenditure

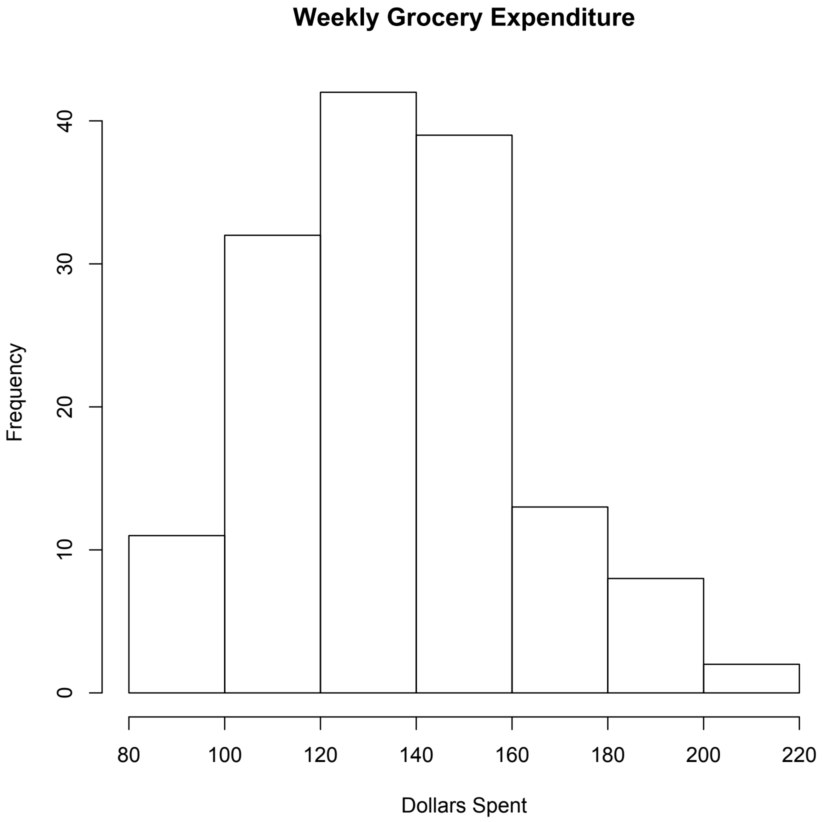


Figure 3.5 Output for Grocery Expenditure from `diagnostic`

```
            Par1        Par2      Con.Corr    Good-Fit A2        Max.dist Sum.Rank
Gamma       24.95837    0.1846079 0.9982464   5.285714 0.1637450 8.678275 4
Lognormal   4.886949    0.1993186 0.9978437   5.47619  0.1811953 9.813802 8
Normal      135.1967    26.96968  0.9921172   7.761905 0.5406865 19.76268 12
Logistic    135.1353    14.96427  0.9865118   10.04762 0.8021629 31.83431 17
Weibull     5.560446    146.3700  0.9796052   13.47619 1.377447  29.28667 19
Laplace     135.0743    19.81241  0.970573    23.38095 1.922842  45.7682  25
Uniform     80.15071    217.51    0.8409656   72.71429 15.39577  36.7912  29
Cauchy      134.3886    14.84865  0.1695811   49.28571 2.174192  1337.111 33
Pareto      81.4305     2.142798  0.3159581   140.5238 23.80584  937.8416 35
Exponential 0.007091597 NA        0.007549144 404.9048 93.57806  583.9428 38
Warning message:
In dweibull(x, shape, scale, log) : NaNs produced
```

Figure 3.6 Graphical Output for Grocery Expenditure

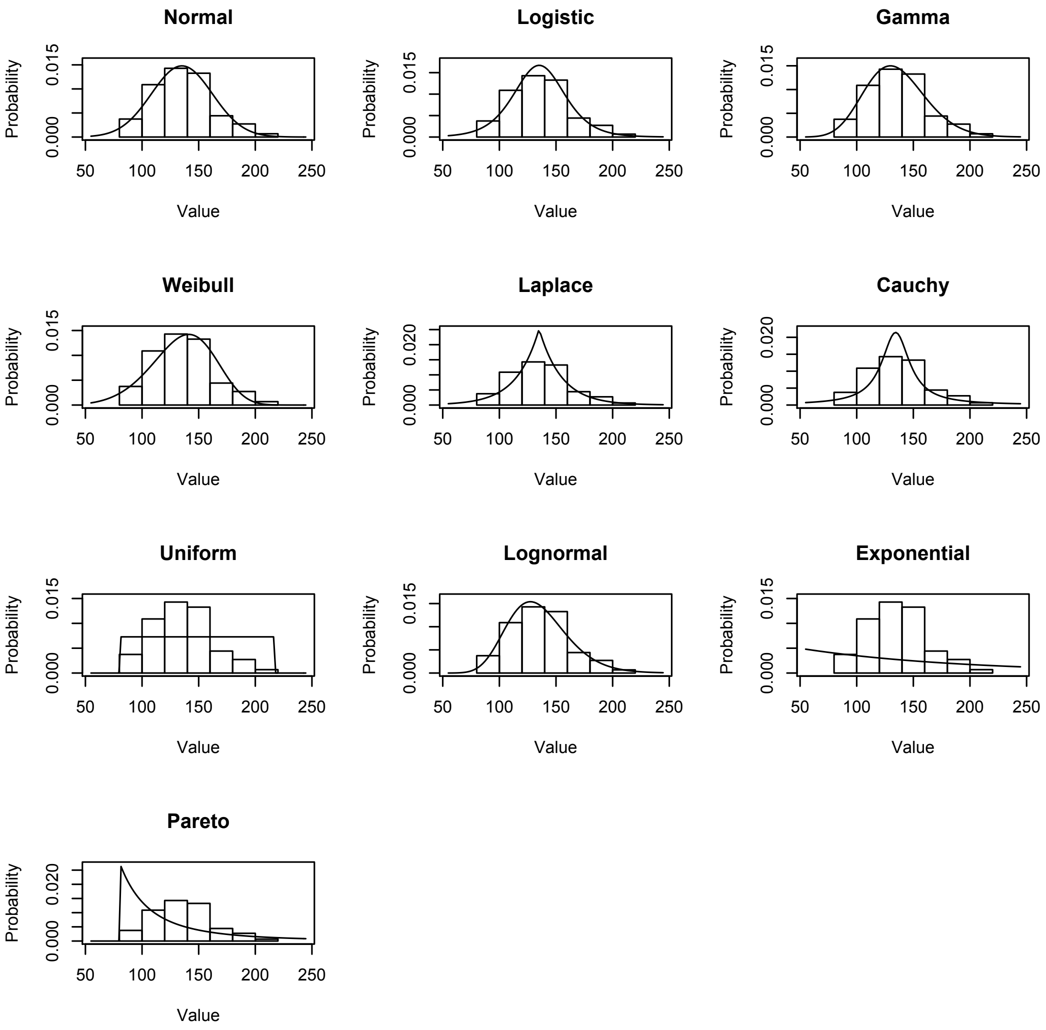
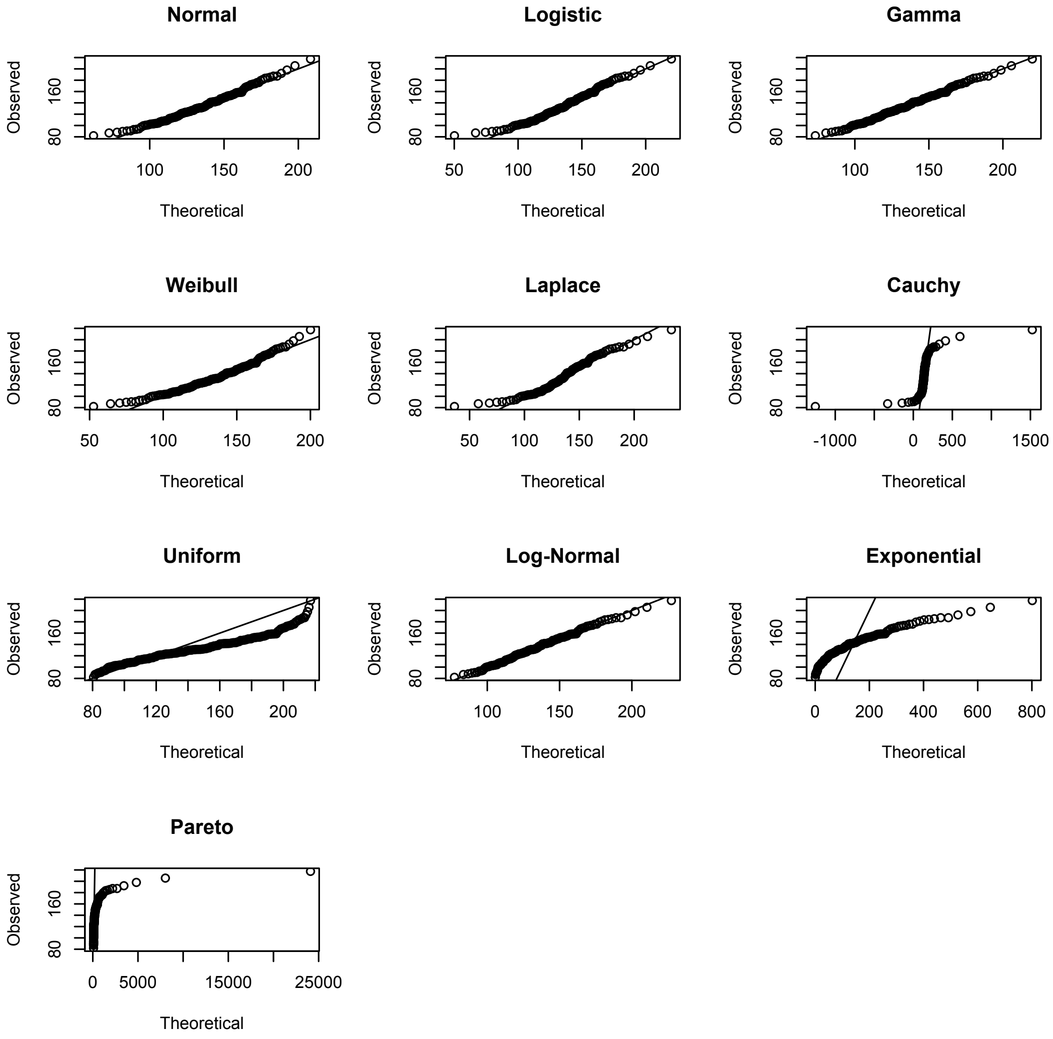


Figure 3.7 Graphical Output for Grocery Expenditure II



3.2 Limitations of `diagnostic`

The first limitation of the proposed function is that whatever data are used, the data have to be within all the support of the random variable X for all the distributions. For example, it is possible to observe negative values for a normal distribution, but not for a Weibull or gamma distribution. The data analyst may need to shift the location of the data before using this function if he suspects that some distribution can provide a good fit if only the data were shifted. However, the argument can be that if data values are outside of the support set for a particular distribution, then that distribution should not be used to fit the data.

Another limitation is that the whole process stops when there is a problem with only one of the distributions. One example is that optimization may fail within the `fitdistr` function. A brief discussion of this can be found in Section 2.2.7. Another example is the parameter estimation technique provided an estimate of a parameter that is impossible to observe, such as what happened in the grocery expenditure data set or calculating a negative estimate of the standard deviation of the normal distribution. Because this function should be general, maximum likelihood estimation may be troublesome for poorly fit distributions. A way to fix this problem would be to not include that particular distribution in the output and give a warning message indicating that distribution was omitted. Another way to fix this problem may be to use another estimation technique, like method of moments, if maximum likelihood methods fail.

In some instances, a warning message appears that looks similar to the following:

```
Warning messages:
1: In dgamma(x, shape, scale, log) : NaNs produced
```

This is a warning message within the function `fitdistr`. This does not seem too problematic, but may have an affect on the validity of the parameter estimates.

The most striking limitation is the number of distributions that are not represented in this function. Distributions like the Gumbel, Rayleigh, extreme value, Wald, and others may provide better fits to the data, but may not be considered due to their absence from this function.

3.3 Further Research

One area that could be examined more is the use of additional statistical measures to arrive at candidate distributions. Statistics such as the Kolmogorov-Smirnov test could be implemented into the function to provide more insight to what distribution data came from. The reason this statistic was not used is because of the suspected sensitivity this statistic had near the center of the hypothesized distribution than at the tails, as outlined by the NIST e-Handbook of Statistical Methods (2003). Appendix C displays four simulations where the function included the Kolmogorov-Smirnov statistic in the ranking procedure. The added statistic did not result in a higher frequency in which the ranking procedure selected the correct distribution. For now the Kolmogorov-Smirnov statistic is omitted; however, this statistic, if used differently, may provide some additional information in assessing which distribution the data follow.

Another area may be to look at some other way of using all the information instead of purely using ranks. More research could be done to arrive at some form of weighted average method of ranking each statistic. As was mentioned earlier, the rankings are forced to be equally spaced, which could cause less accuracy in determining reasonable distributions.

Better methods of parameter estimation could also be employed. Parameters of some distributions have many estimators which work well in some instances, but not in others. Often, the maximum likelihood estimators have either bias or inflated variance which other estimators may correct.

The function uses a method of parameter estimation based on maximum likelihood estimation and using values within an interval to arrive at a large concordance correlation. A similar process can be examined that minimizes the goodness of fit and/or the Anderson-Darling statistic. It is more likely that ties may occur in the goodness of fit statistic for multiple values of parameter estimates because of the binning of the data. The value of the concordance correlation and/or the Anderson-Darling statistic could serve as a tiebreaker to decide which set of parameter estimates to use. Also, choosing the parameter estimates that minimize the Anderson-Darling statistic could be employed.

The concordance correlation is not widespread, but this could be a way to assess the variation within QQ plots. Research can be done to use this for hypothesis testing similar to the tests that use the goodness of fit or Anderson-Darling statistic. No hypothesis testing is done with the function, but maybe some tests can be generated if the proper criteria are met.

3.4 Conclusion

It is important to have an idea what distribution data potentially come from because many statistical tests are based on that notion. Often the data analyst assumes data follow some distribution, but in reality that assumption can be erroneous. The implications of making such an error have many effects on the power, or even validity, of the test. There are statistical tests to test whether data came from a specific distribution or not, but they get underused because it is tedious to do many hypothesis tests and some distributions are more difficult to work with. Many tests, even ones testing whether data follows a distribution, are asymptotic tests that require a large sample size. Many times the data analyst encounters small sample sizes for which the asymptotic properties of the statistical test are not met. It is possible for two or more distributions to look very similar, especially if the sample is not large, so hypothesis testing can also lead to conflicting results. Finding a distribution to model sample data can be challenging, but this proposed function provides a useful tool to at least determine reasonable distributions to model the data. Because of this function's ease of use, hopefully data analysts will not be so quick to make assumptions about the data.

Some distributions are easier to work with than others. If data looks remotely normally distributed, the normal distribution is most often used to model the data. There are, indeed, distributions that can look very much like the normal distribution. If data is shaped in such a way that multiple distributions can effectively model the data, it would be practical to use the distribution that is easiest to work with. Caution, however, should be exercised. Small departures from the assumptions of the data can lead to problems. Many of the statistical tests operate on strict constraints, and if even one of those constraints is violated, the validity of the test must be questioned.

It is my hope that data analysts pause and perhaps question the notion that the data set they are analyzing did come from a certain distribution. This function provides numerous pieces of information to help arrive at candidate distributions.

Bibliography

Anderson, T. W., & Darling, D. A. (1952). Asymptotic theory of certain "goodness-of-fit" criteria based on stochastic processes. *Annals of Mathematical Statistics*, 23, 193-212. doi: 10.1214/aoms/1177729437

Asrabadi, B. R. (1985). The exact confidence interval for the scale parameter and the MVUE of the laplace distribution. *Communications in Statistics - Theory and Methods*, 14, 713-733.

Balakrishnan, N. (Ed.). (1992). *Handbook of the logistic distribution*. New York: Dekker.

Casella, G., & Berger, R. L. (2002). *Statistical inference* (2nd ed.). Pacific Grove, CA: Wadsworth Group.

Chi-square goodness-of-fit test. (2003). *NIST/SEMATECH e-handbook of statistical methods* () Retrieved from http://www.itl.nist.gov/div898/handbook/prc/section2/prc211.htm

General-purpose optimization. R Documentation:

Johnson, N. L., Kotz, S., & Balakrishnan, N. (1995). *Continuous univariate distributions* (2nd ed.). New York: John Wiley & Sons, Inc.

Kolmogorov-Smirnov goodness-of-fit test. (2003). *NIST/SEMATECH e-handbook of statistical methods* () Retrieved from http://www.itl.nist.gov/div898/handbook/eda/section3/eda35g.htm

Kotz, S., & Neumann, J. (1963). On distributions of precipitation amounts for the periods of increasing length. *Journal of Geophysical Research*, 68, 3635-3641.

Krishnamoorthy, K. (2006). *Handbook of statistical distributions with applications*. Boca Raton, FL: Chapman & Hall/CRC Press.

Larson, R.J., & Marx, M.L. (1986). *An introduction to mathematical statistics and its applications* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

Lin, L. (1989). A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, 45, 255-268.

Maximum likelihood fitting of univariate distributions. R Documentation:

Nelder, J.A., & Mead, R. (1965). A simplex algorithm for function minimization. *Computer Journal*, 7, 308-313.

Shorack, G.R., & Wellner, J.A. (1986). *Empirical processes with applications to statistics*. New York: John Wiley & Sons, Inc.

Stephens, M. A. (1974). EDF statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association*, 69, 730-737.

Appendix A - The R Code for the Function `diagnostic`

```
## Before this function can be used, these packages need to be loaded ##
library(MASS)
library(lattice)
library(VGAM)
diagnostic=function(data){
     data=sort(data)
     x=data
     n=length(data)
     p=ppoints(n)
     b=(n-1)/n
     seq=seq(-1,1,length=9)
     fit.norm=fitdistr(data,'normal')
     a.norm=fit.norm$estimate[1]+seq*fit.norm$sd[1]
     b.norm=fit.norm$estimate[2]+seq*fit.norm$sd[2]
     q.norm0=matrix(NA,nrow=81,ncol=1)
     q.norm0[,1]=rep(a.norm,9)
     q.norm1=matrix(NA,nrow=9,ncol=9)
     for(i in 1:9){q.norm1[,i]=rep(b.norm[i],9)}
     q.norm1=matrix(q.norm1,nrow=81,ncol=1)
     q.normf=cbind(q.norm0,q.norm1)
     qq.norm=matrix(NA,nrow=n,ncol=81)
     for(i in 1:81){qq.norm[,i]=qnorm(p,q.normf[i,1],q.normf[i,2])}
          rhoc.norm=numeric(81)
     for(i in 1:81){
     rhoc.norm[i]=(2*b*cov(qq.norm[,i],data))/(b*var(qq.norm[,i])+b*var(data)+(mean
(qq.norm[,i])-mean(data))^2)
     }
     rhoc.norm=as.numeric(rhoc.norm)
     rhoc.norm=as.matrix(rhoc.norm)
     rhocm.norm=matrix(c(q.normf,rhoc.norm),nrow=81,ncol=3)
     rhocm.norm=rhocm.norm[rev(order(rhocm.norm[,3])),,drop=FALSE]
     par.norm=list(rhocm.norm[1,1],rhocm.norm[1,2],rhocm.norm[1,3])
     #names(par.norm)=c('mean.norm','stddev.norm','rc.norm')
     par.normd=data.frame(par.norm)

     fit.exp=fitdistr(data,'exponential')
     a.exp=fit.exp$estimate[1]+seq*fit.exp$sd[1]
     q.exp0=matrix(a.exp,nrow=9,ncol=1)
     qq.exp=matrix(NA,nrow=n,ncol=9)
     for(i in 1:9){qq.exp[,i]=qexp(p,q.exp0[i,1])}
          rhoc.exp=numeric(9)
     for(i in 1:9){
     rhoc.exp[i]=(2*b*cov(qq.exp[,i],data))/(b*var(qq.exp[,i])+b*var(data)+(mean
(qq.exp[,i])-mean(data))^2)
          }
     rhoc.exp=as.numeric(rhoc.exp)
     rhoc.exp=as.matrix(rhoc.exp)
     rhocm.exp=matrix(c(q.exp0,rhoc.exp),nrow=3,ncol=2)
     rhocm.exp=rhocm.exp[rev(order(rhocm.exp[,2])),,drop=FALSE]
     par.exp=list(rhocm.exp[1,1],rhocm.exp[1,2])
     #names(par.exp)=c('rate.exp','rc.exp')
     par.expd=data.frame(par.exp)

     fit.gamma=fitdistr(data,'gamma')
     a.gamma=fit.gamma$estimate[1]+seq*fit.gamma$sd[1]
     b.gamma=fit.gamma$estimate[2]+seq*fit.gamma$sd[2]
     q.gamma0=matrix(NA,nrow=81,ncol=1)
     q.gamma0[,1]=rep(a.gamma,9)
```

```
    q.gamma1=matrix(NA,nrow=9,ncol=9)
    for(i in 1:9){q.gamma1[,i]=rep(b.gamma[i],9)}
    q.gamma1=matrix(q.gamma1,nrow=81,ncol=1)
    q.gammaf=cbind(q.gamma0,q.gamma1)
    qq.gamma=matrix(NA,nrow=n,ncol=81)
    for(i in 1:81){qq.gamma[,i]=qgamma(p,q.gammaf[i,1],q.gammaf[i,2])}
        rhoc.gamma=numeric(81)
    for(i in 1:81){
    rhoc.gamma[i]=(2*b*cov(qq.gamma[,i],data))/(b*var(qq.gamma[,i])+b*var(data)+(mean
(qq.gamma[,i])-mean(data))^2)
    }
    rhoc.gamma=as.numeric(rhoc.gamma)
    rhoc.gamma=as.matrix(rhoc.gamma)
    rhocm.gamma=matrix(c(q.gammaf,rhoc.gamma),nrow=81,ncol=3)
    rhocm.gamma=rhocm.gamma[rev(order(rhocm.gamma[,3])),,drop=FALSE]
    par.gamma=list(rhocm.gamma[1,1],rhocm.gamma[1,2],rhocm.gamma[1,3])
    #names(par.gamma)=c('shape.gamma','rate.gamma','rc.gamma')
    par.gammad=data.frame(par.gamma)

    fit.logis=fitdistr(data,'logistic',list(location=mean(data),scale=(sqrt(3)/pi)*sd
(data)))
    a.logis=fit.logis$estimate[1]+seq*fit.logis$sd[1]
    b.logis=fit.logis$estimate[2]+seq*fit.logis$sd[2]
    q.logis0=matrix(NA,nrow=81,ncol=1)
    q.logis0[,1]=rep(a.logis,9)
    q.logis1=matrix(NA,nrow=9,ncol=9)
    for(i in 1:9){q.logis1[,i]=rep(b.logis[i],9)}
    q.logis1=matrix(q.logis1,nrow=81,ncol=1)
    q.logisf=cbind(q.logis0,q.logis1)
    qq.logis=matrix(NA,nrow=n,ncol=81)
    for(i in 1:81){qq.logis[,i]=qlogis(p,q.logisf[i,1],q.logisf[i,2])}
        rhoc.logis=numeric(81)
    for(i in 1:81){
    rhoc.logis[i]=(2*b*cov(qq.logis[,i],data))/(b*var(qq.logis[,i])+b*var(data)+(mean
(qq.logis[,i])-mean(data))^2)
    }
    rhoc.logis=as.numeric(rhoc.logis)
    rhoc.logis=as.matrix(rhoc.logis)
    rhocm.logis=matrix(c(q.logisf,rhoc.logis),nrow=81,ncol=3)
    rhocm.logis=rhocm.logis[rev(order(rhocm.logis[,3])),,drop=FALSE]
    par.logis=list(rhocm.logis[1,1],rhocm.logis[1,2],rhocm.logis[1,3])
    #names(par.logis)=c('loc.logis','scale.logis','rc.logis')
    par.logisd=data.frame(par.logis)

    fit.lnorm=fitdistr(data,'lognormal')
    a.lnorm=fit.lnorm$estimate[1]+seq*fit.lnorm$sd[1]
    b.lnorm=fit.lnorm$estimate[2]+seq*fit.lnorm$sd[2]
    q.lnorm0=matrix(NA,nrow=81,ncol=1)
    q.lnorm0[,1]=rep(a.lnorm,9)
    q.lnorm1=matrix(NA,nrow=9,ncol=9)
    for(i in 1:9){q.lnorm1[,i]=rep(b.lnorm[i],9)}
    q.lnorm1=matrix(q.lnorm1,nrow=81,ncol=1)
    q.lnormf=cbind(q.lnorm0,q.lnorm1)
    qq.lnorm=matrix(NA,nrow=n,ncol=81)
    for(i in 1:81){qq.lnorm[,i]=qlnorm(p,q.lnormf[i,1],q.lnormf[i,2])}
        rhoc.lnorm=numeric(81)
    for(i in 1:81){
    rhoc.lnorm[i]=(2*b*cov(qq.lnorm[,i],data))/(b*var(qq.lnorm[,i])+b*var(data)+(mean
(qq.lnorm[,i])-mean(data))^2)
    }
    rhoc.lnorm=as.numeric(rhoc.lnorm)
    rhoc.lnorm=as.matrix(rhoc.lnorm)
    rhocm.lnorm=matrix(c(q.lnormf,rhoc.lnorm),nrow=81,ncol=3)
    rhocm.lnorm=rhocm.lnorm[rev(order(rhocm.lnorm[,3])),,drop=FALSE]
    par.lnorm=list(rhocm.lnorm[1,1],rhocm.lnorm[1,2],rhocm.lnorm[1,3])
    #names(par.lnorm)=c('meanlog.lnorm','sdlog.lnorm','rc.lnorm')
    par.lnormd=data.frame(par.lnorm)

    c=((sqrt(6)/pi)*sd(log(data)))^(-1)
    fit.weibull=fitdistr(data,'weibull',list(shape=c,scale=((1/n)*sum(data^c))^(1/
c)))
    a.weibull=fit.weibull$estimate[1]+seq*fit.weibull$sd[1]
    b.weibull=fit.weibull$estimate[2]+seq*fit.weibull$sd[2]
    q.weibull0=matrix(NA,nrow=81,ncol=1)
    q.weibull0[,1]=rep(a.weibull,9)
    q.weibull1=matrix(NA,nrow=9,ncol=9)
    for(i in 1:9){q.weibull1[,i]=rep(b.weibull[i],9)}
    q.weibull1=matrix(q.weibull1,nrow=81,ncol=1)
    q.weibullf=cbind(q.weibull0,q.weibull1)
    qq.weibull=matrix(NA,nrow=n,ncol=81)
    for(i in 1:81){qq.weibull[,i]=qweibull(p,q.weibullf[i,1],q.weibullf[i,2])}
        rhoc.weibull=numeric(81)
    for(i in 1:81){
    rhoc.weibull[i]=(2*b*cov(qq.weibull[,i],data))/(b*var(qq.weibull[,i])+b*var(data)
+(mean(qq.weibull[,i])-mean(data))^2)
    }
    rhoc.weibull=as.numeric(rhoc.weibull)
    rhoc.weibull=as.matrix(rhoc.weibull)
    rhocm.weibull=matrix(c(q.weibullf,rhoc.weibull),nrow=81,ncol=3)
    rhocm.weibull=rhocm.weibull[rev(order(rhocm.weibull[,3])),,drop=FALSE]
    par.weibull=list(rhocm.weibull[1,1],rhocm.weibull[1,2],rhocm.weibull[1,3])
    #names(par.weibull)=c('shape.weibull','scale.weibull','rc.weibull')
    par.weibulld=data.frame(par.weibull)

    fit.cauchy=fitdistr(data,'cauchy')
    a.cauchy=fit.cauchy$estimate[1]+seq*fit.cauchy$sd[1]
    b.cauchy=fit.cauchy$estimate[2]+seq*fit.cauchy$sd[2]
    q.cauchy0=matrix(NA,nrow=81,ncol=1)
    q.cauchy0[,1]=rep(a.cauchy,9)
    q.cauchy1=matrix(NA,nrow=9,ncol=9)
    for(i in 1:9){q.cauchy1[,i]=rep(b.cauchy[i],9)}
    q.cauchy1=matrix(q.cauchy1,nrow=81,ncol=1)
    q.cauchyf=cbind(q.cauchy0,q.cauchy1)
    qq.cauchy=matrix(NA,nrow=n,ncol=81)
    for(i in 1:81){qq.cauchy[,i]=qcauchy(p,q.cauchyf[i,1],q.cauchyf[i,2])}
        rhoc.cauchy=numeric(81)
    for(i in 1:81){
    rhoc.cauchy[i]=(2*b*cov(qq.cauchy[,i],data))/(b*var(qq.cauchy[,i])+b*var(data)+
(mean(qq.cauchy[,i])-mean(data))^2)
    }
    rhoc.cauchy=as.numeric(rhoc.cauchy)
    rhoc.cauchy=as.matrix(rhoc.cauchy)
    rhocm.cauchy=matrix(c(q.cauchyf,rhoc.cauchy),nrow=81,ncol=3)
    rhocm.cauchy=rhocm.cauchy[rev(order(rhocm.cauchy[,3])),,drop=FALSE]
    par.cauchy=list(rhocm.cauchy[1,1],rhocm.cauchy[1,2],rhocm.cauchy[1,3])
    #names(par.cauchy)=c('loc.cauchy','scale.cauchy','rc.cauchy')
    par.cauchyd=data.frame(par.cauchy)

    like.laplace=function(par.lap,x,n){
        a=par.lap[1]
        b=par.lap[2]
        likelaplace=-n*log(2*b)-(1/b)*sum(abs(x-a))
        return(likelaplace)
        }

    o=optim(c(median(data),(1/n)*sum(abs(data-median
(data)))),like.laplace,x=data,n=length(data),control=list(fnscale=-1),hessian=TRUE)
    HI=(-1*o$hessian[2,2])^(-1)
```

```
     se.lap=sqrt(HI)
     loc.lap=o$par[1]
     scale.lap=o$par[2]
     qbl=qbinom(0.32/2,n,0.5)
     a.l=seq(data[qbl-1],data[n-qbl],length=8)

     a.laplace=c(a.l,median(data))
     b.laplace=scale.lap+seq*se.lap
     q.laplace0=matrix(NA,nrow=81,ncol=1)
     q.laplace0[,1]=rep(a.laplace,9)
     q.laplace1=matrix(NA,nrow=9,ncol=9)
     for(i in 1:9){q.laplace1[,i]=rep(b.laplace[i],9)}
     q.laplace1=matrix(q.laplace1,nrow=81,ncol=1)
     q.laplacef=cbind(q.laplace0,q.laplace1)
     qq.laplace=matrix(NA,nrow=n,ncol=81)
     for(i in 1:81){qq.laplace[,i]=qlaplace(p,q.laplacef[i,1],q.laplacef[i,2])}
          rhoc.laplace=numeric(81)
     for(i in 1:81){
     rhoc.laplace[i]=(2*b*cov(qq.laplace[,i],data))/(b*var(qq.laplace[,i])+b*var(data)
+(mean(qq.laplace[,i])-mean(data))^2)
     }
     rhoc.laplace=as.numeric(rhoc.laplace)
     rhoc.laplace=as.matrix(rhoc.laplace)
     rhocm.laplace=matrix(c(q.laplacef,rhoc.laplace),nrow=81,ncol=3)
     rhocm.laplace=rhocm.laplace[rev(order(rhocm.laplace[,3])),,drop=FALSE]
     par.laplace=list(rhocm.laplace[1,1],rhocm.laplace[1,2],rhocm.laplace[1,3])
     #names(par.laplace)=c('loc.laplace','scale.laplace','rc.laplace')
     par.laplaced=data.frame(par.laplace)

     a.hat=min(data)
     b.hat=max(data)
     u.var=(n*(b.hat-a.hat)^2)/((n+1)^2*(n+2))
     se.u=sqrt(u.var)
     seq.l=seq(-2,0,length=9)
     seq.up=seq(0,2,length=9)
     a.unif=a.hat+seq.l*se.u
     b.unif=b.hat+seq.up*se.u
     q.unif0=matrix(NA,nrow=81,ncol=1)
     q.unif0[,1]=rep(a.unif,9)
     q.unif1=matrix(NA,nrow=9,ncol=9)
     for(i in 1:9){q.unif1[,i]=rep(b.unif[i],9)}
     q.unif1=matrix(q.unif1,nrow=81,ncol=1)
     q.uniff=cbind(q.unif0,q.unif1)
     qq.unif=matrix(NA,nrow=n,ncol=81)
     for(i in 1:81){qq.unif[,i]=qunif(p,q.uniff[i,1],q.uniff[i,2])}
          rhoc.unif=numeric(81)
     for(i in 1:81){
     rhoc.unif[i]=(2*b*cov(qq.unif[,i],data))/(b*var(qq.unif[,i])+b*var(data)+(mean
(qq.unif[,i])-mean(data))^2)
     }
     rhoc.unif=as.numeric(rhoc.unif)
     rhoc.unif=as.matrix(rhoc.unif)
     rhocm.unif=matrix(c(q.uniff,rhoc.unif),nrow=81,ncol=3)
     rhocm.unif=rhocm.unif[rev(order(rhocm.unif[,3])),,drop=FALSE]
     par.unif=list(rhocm.unif[1,1],rhocm.unif[1,2],rhocm.unif[1,3])
     #names(par.unif)=c('min.unif','max.unif','rc.unif')
     par.unifd=data.frame(par.unif)

     a.hatp=min(data)
     gm=(prod(data))^(1/n)
     b.hatp=1/((1/n)*sum(log(data))-log(a.hatp))
     par.var=a.hatp^2*b.hatp*n*(1/(b.hatp*n-2)-(n*b.hatp)/(b.hatp*n-1)^2)
     se.p=sqrt(par.var)
     b.lowerp=b.hatp*qchisq(.32,2*(n-1))/(2*n)
     b.upperp=b.hatp*qchisq(1-.32,2*(n-1))/(2*n)
     a.pareto=a.hatp+seq.l*se.p
     b.pareto=seq(b.lowerp,b.upperp,length=9)
     q.pareto0=matrix(NA,nrow=81,ncol=1)
     q.pareto0[,1]=rep(a.pareto,9)
     q.pareto1=matrix(NA,nrow=9,ncol=9)
     for(i in 1:9){q.pareto1[,i]=rep(b.pareto[i],9)}
     q.pareto1=matrix(q.pareto1,nrow=81,ncol=1)
     q.paretof=cbind(q.pareto0,q.pareto1)
     qq.pareto=matrix(NA,nrow=n,ncol=81)
     for(i in 1:81){qq.pareto[,i]=qpareto(p,q.paretof[i,1],q.paretof[i,2])}
          rhoc.pareto=numeric(81)
     for(i in 1:81){
     rhoc.pareto[i]=(2*b*cov(qq.pareto[,i],data))/(b*var(qq.pareto[,i])+b*var(data)+
(mean(qq.pareto[,i])-mean(data))^2)
     }
     rhoc.pareto=as.numeric(rhoc.pareto)
     rhoc.pareto=as.matrix(rhoc.pareto)
     rhocm.pareto=matrix(c(q.paretof,rhoc.pareto),nrow=81,ncol=3)
     rhocm.pareto=rhocm.pareto[rev(order(rhocm.pareto[,3])),,drop=FALSE]
     par.pareto=list(rhocm.pareto[1,1],rhocm.pareto[1,2],rhocm.pareto[1,3])
     #names(par.pareto)=c('loc.pareto','shape.pareto','rc.pareto')
     par.paretod=data.frame(par.pareto)

     par.expdm=list(par.expd[1],NA,par.expd[2])

     par.expdm=data.frame(par.expdm)

     qn=qnorm(p,rhocm.norm[1,1],rhocm.norm[1,2])
     qx=qexp(p,rhocm.exp[1,1])
     qg=qgamma(p,rhocm.gamma[1,1],rhocm.gamma[1,2])
     ql=qlogis(p,rhocm.logis[1,1],rhocm.logis[1,2])
     qlog=qlnorm(p,rhocm.lnorm[1,1],rhocm.lnorm[1,2])
     qw=qweibull(p,rhocm.weibull[1,1],rhocm.weibull[1,2])
     qc=qcauchy(p,rhocm.cauchy[1,1],rhocm.cauchy[1,2])
     qlap=qlaplace(p,rhocm.laplace[1,1],rhocm.laplace[1,2])
     qu=qunif(p,rhocm.unif[1,1],rhocm.unif[1,2])
     qp=qpareto(p,rhocm.pareto[1,1],rhocm.pareto[1,2])

     distn=max(abs(qn-data))
     distx=max(abs(qx-data))
     distg=max(abs(qg-data))
     distl=max(abs(ql-data))
     distlog=max(abs(qlog-data))
     distw=max(abs(qw-data))
     distc=max(abs(qc-data))
     distlap=max(abs(qlap-data))
     distu=max(abs(qu-data))
     distp=max(abs(qp-data))
     dist=data.frame(c
(distn,distx,distg,distl,distlog,distw,distc,distlap,distu,distp))

     g=round(2*n^(2/5))
     pg=ppoints(g,1)
     ppg=c(pg[1]+0.000000000001,pg[2:(g-1)],pg[g]-0.000000000001)
     E=n/(g-1)
     qng=qnorm(pg,rhocm.norm[1,1],rhocm.norm[1,2])
     qxg=qexp(pg,rhocm.exp[1,1])
     qgg=qgamma(pg,rhocm.gamma[1,1],rhocm.gamma[1,2])
     qlg=qlogis(pg,rhocm.logis[1,1],rhocm.logis[1,2])
     qlogg=qlnorm(pg,rhocm.lnorm[1,1],rhocm.lnorm[1,2])
     qwg=qweibull(pg,rhocm.weibull[1,1],rhocm.weibull[1,2])
     qcg=qcauchy(pg,rhocm.cauchy[1,1],rhocm.cauchy[1,2])
     qlapg=qlaplace(pg,rhocm.laplace[1,1],rhocm.laplace[1,2])
```

```
      qug=qunif(pg,rhocm.unif[1,1],rhocm.unif[1,2])
      qpg=qpareto(ppg,rhocm.pareto[1,1],rhocm.pareto[1,2])

      cn=table(cut(data,breaks=qng))
      cx=table(cut(data,breaks=qxg))
      cg=table(cut(data,breaks=qgg))
      cl=table(cut(data,breaks=qlg))
      clog=table(cut(data,breaks=qlogg))
      cw=table(cut(data,breaks=qwg))
      cc=table(cut(data,breaks=qcg))
      clap=table(cut(data,breaks=qlapg))
      cu=table(cut(data,breaks=qug))
      cp=table(cut(data,breaks=qpg))

      chin=sum((cn-E)^2/E)
      chix=sum((cx-E)^2/E)
      chig=sum((cg-E)^2/E)
      chil=sum((cl-E)^2/E)
      chilog=sum((clog-E)^2/E)
      chiw=sum((cw-E)^2/E)
      chic=sum((cc-E)^2/E)
      chilap=sum((clap-E)^2/E)
      chiu=sum((cu-E)^2/E)
      chip=sum((cp-E)^2/E)
      chi=data.frame(c(chin,chix,chig,chil,chilog,chiw,chic,chilap,chiu,chip))



      Sn=vector()
      for(i in 1:n){Sn[i]=((2*i-1)/n)*(log(pnorm(x[i],rhocm.norm[1,1],rhocm.norm[1,2]))
+log(1-pnorm(x[n+1-i],rhocm.norm[1,1],rhocm.norm[1,2]))))}
      An=-n-sum(Sn)
      Sx=vector()
      for(i in 1:n){Sx[i]=((2*i-1)/n)*(log(pexp(x[i],rhocm.exp[1,1],rhocm.exp[1,2]))
+log(1-pexp(x[n+1-i],rhocm.exp[1,1],rhocm.exp[1,2]))))}
      Ax=-n-sum(Sx)
      Sg=vector()
      for(i in 1:n){Sg[i]=((2*i-1)/n)*(log(pgamma(x[i],rhocm.gamma[1,1],rhocm.gamma
[1,2]))+log(1-pgamma(x[n+1-i],rhocm.gamma[1,1],rhocm.gamma[1,2]))))}
      Ag=-n-sum(Sg)
      Sl=vector()
      for(i in 1:n){Sl[i]=((2*i-1)/n)*(log(plogis(x[i],rhocm.logis[1,1],rhocm.logis
[1,2]))+log(1-plogis(x[n+1-i],rhocm.logis[1,1],rhocm.logis[1,2]))))}
      Al=-n-sum(Sl)
      Slog=vector()
      for(i in 1:n){Slog[i]=((2*i-1)/n)*(log(plnorm(x[i],rhocm.lnorm[1,1],rhocm.lnorm
[1,2]))+log(1-plnorm(x[n+1-i],rhocm.lnorm[1,1],rhocm.lnorm[1,2]))))}
      Alog=-n-sum(Slog)
      Sw=vector()
      for(i in 1:n){Sw[i]=((2*i-1)/n)*(log(pweibull(x[i],rhocm.weibull
[1,1],rhocm.weibull[1,2]))+log(1-pweibull(x[n+1-i],rhocm.weibull[1,1],rhocm.weibull
[1,2]))))}
      Aw=-n-sum(Sw)
      Sc=vector()
      for(i in 1:n){Sc[i]=((2*i-1)/n)*(log(pcauchy(x[i],rhocm.cauchy[1,1],rhocm.cauchy
[1,2]))+log(1-pcauchy(x[n+1-i],rhocm.cauchy[1,1],rhocm.cauchy[1,2]))))}
      Ac=-n-sum(Sc)
      Slap=vector()
      for(i in 1:n){Slap[i]=((2*i-1)/n)*(log(plaplace(x[i],rhocm.laplace
[1,1],rhocm.laplace[1,2]))+log(1-plaplace(x[n+1-i],rhocm.laplace[1,1],rhocm.laplace
[1,2]))))}
      Alap=-n-sum(Slap)
      Su=vector()
      for(i in 1:n){Su[i]=((2*i-1)/n)*(log(punif(x[i],rhocm.unif
[1,1]-0.00001,rhocm.unif[1,2]+0.00001))+log(1-punif(x[n+1-i],rhocm.unif
[1,1]-0.00001,rhocm.unif[1,2]+0.00001)))}
      Au=-n-sum(Su)
      Sp=vector()
      for(i in 1:n){Sp[i]=((2*i-1)/n)*(log(ppareto(x[i],rhocm.pareto
[1,1]-0.00001,rhocm.pareto[1,2]))+log(1-ppareto(x[n+1-i],rhocm.pareto
[1,1]-0.00001,rhocm.pareto[1,2])))}
      Ap=-n-sum(Sp)
      A2=data.frame(c(An,Ax,Ag,Al,Alog,Aw,Ac,Alap,Au,Ap))


      rA2=rank(A2)
      rchi=rank(chi)
      rdist=rank(dist)
      r=data.frame(c(par.normd[3],par.expdm[3],par.gammad[3],par.logisd[3],par.lnormd
[3],par.weibulld[3],par.cauchyd[3],par.laplaced[3],par.unifd[3],par.paretod[3]))
      r=-r
      rrho=rank(r)

      rn=sum(rchi[1],rdist[1],rrho[1],rA2[1])
      rx=sum(rchi[2],rdist[2],rrho[2],rA2[2])
      rg=sum(rchi[3],rdist[3],rrho[3],rA2[3])
      rl=sum(rchi[4],rdist[4],rrho[4],rA2[4])
      rlog=sum(rchi[5],rdist[5],rrho[5],rA2[5])
      rw=sum(rchi[6],rdist[6],rrho[6],rA2[6])
      rc=sum(rchi[7],rdist[7],rrho[7],rA2[7])
      rlap=sum(rchi[8],rdist[8],rrho[8],rA2[8])
      ru=sum(rchi[9],rdist[9],rrho[9],rA2[9])
      rp=sum(rchi[10],rdist[10],rrho[10],rA2[10])

      parmat=matrix(c
(par.normd,chin,An,distn,rn,par.expdm,chix,Ax,distx,rx,par.gammad,chig,Ag,distg,rg,par
.logisd,chil,Al,distl,rl,par.lnormd,chilog,Alog,distlog,rlog,par.weibulld,chiw,Aw,dist
w,rw,par.cauchyd,chic,Ac,distc,rc,par.laplaced,chilap,Alap,distlap,rlap,par.unifd,chiu
,Au,distu,ru,par.paretod,chip,Ap,distp,rp),nrow=10,ncol=7,byrow=TRUE,dimnames=list(c
('Normal','Exponential','Gamma','Logistic','Lognormal','Weibull','Cauchy','Laplace','U
niform','Pareto'),c('Par1','Par2','Con.Corr','Good-Fit','A2','Max.dist','Sum.Rank')))

dn=dnorm(x,rhocm.norm[1,1],rhocm.norm[1,2])+0.001
dl=dlogis(x,rhocm.logis[1,1],rhocm.logis[1,2])+0.001
dg=dgamma(x,rhocm.gamma[1,1],rhocm.gamma[1,2])+0.001
dw=dweibull(x,rhocm.weibull[1,1],rhocm.weibull[1,2])+0.001
dlap=dlaplace(x,rhocm.laplace[1,1],rhocm.laplace[1,2])+0.001
dc=dcauchy(x,rhocm.cauchy[1,1],rhocm.cauchy[1,2])+0.001
du=dunif(x,rhocm.unif[1,1],rhocm.unif[1,2])+0.001
dlog=dlnorm(x,rhocm.lnorm[1,1],rhocm.lnorm[1,2])+0.001
dx=dexp(x,rhocm.exp[1,1])+0.001
dp=dpareto(x,rhocm.pareto[1,1],rhocm.pareto[1,2])+0.001


his=hist(data,plot=F)
his=max(his$density)+0.001
par(mfrow=c(4,3))
curve(dnorm(x,rhocm.norm[1,1],rhocm.norm[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Normal',ylab='Probability',xlab='Value',ylim=c(0,max(his,dn)))
hist(data,probability=T,add=T)
curve(dlogis(x,rhocm.logis[1,1],rhocm.logis[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Logistic',ylab='Probability',xlab='Value',ylim=c(0,max(his,dl)))
hist(data,probability=T,add=T)
curve(dgamma(x,rhocm.gamma[1,1],rhocm.gamma[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Gamma',ylab='Probability',xlab='Value',ylim=c(0,max(his,dg)))
hist(data,probability=T,add=T)
```

```
curve(dweibull(x,rhocm.weibull[1,1],rhocm.weibull[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Weibull',ylab='Probability',xlab='Value',ylim=c(0,max(his,dw)))
hist(data,probability=T,add=T)
curve(dlaplace(x,rhocm.laplace[1,1],rhocm.laplace[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Laplace',ylab='Probability',xlab='Value',ylim=c(0,max(his,dlap)))
hist(data,probability=T,add=T)
curve(dcauchy(x,rhocm.cauchy[1,1],rhocm.cauchy[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Cauchy',ylab='Probability',xlab='Value',ylim=c(0,max(his,dc)))
hist(data,probability=T,add=T)
curve(dunif(x,rhocm.unif[1,1],rhocm.unif[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Uniform',ylab='Probability',xlab='Value',ylim=c(0,max(his,du)))
hist(data,probability=T,add=T)
curve(dlnorm(x,rhocm.lnorm[1,1],rhocm.lnorm[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Lognormal',ylab='Probability',xlab='Value',ylim=c(0,max(his,dlog)))
hist(data,probability=T,add=T)
curve(dexp(x,rhocm.exp[1,1]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Exponential',ylab='Probability',xlab='Value',ylim=c(0,max(his,dx)))
hist(data,probability=T,add=T)
curve(dpareto(x,rhocm.pareto[1,1],rhocm.pareto[1,2]),from=min(x)-sd(x),to=max(x)+sd
(x),main='Pareto',ylab='Probability',xlab='Value',ylim=c(0,max(his,dp)))
hist(data,probability=T,add=T)

x11()
     #QQ Plots

     par(mfrow=c(4,3))
     plot(qn,data,main="Normal",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(ql,data,main="Logistic",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(qg,data,main="Gamma",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(qw,data,main="Weibull",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(qlap,data,main="Laplace",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(qc,data,main="Cauchy",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(qu,data,main="Uniform",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(qlog,data,main="Log-Normal",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(qx,data,main="Exponential",xlab="Theoretical",ylab="Observed")
     abline(0,1)
     plot(qp,data,main="Pareto",xlab="Theoretical",ylab="Observed")
     abline(0,1)





     parmat=parmat[(order(as.numeric(parmat[,7]))),,drop=FALSE] #Sorting the matrix by
rank.

     parmat

     }
```

Appendix B - The R Code for the Simulations

For the purpose of simulations, the same code (Appendix A) was used with the following exceptions: (1) no graphical output was used, (2) no output was generated, and (3) only the distribution with the lowest sum of ranks was recorded. The following is the amended code which selects the distribution with the lowest sum of ranks along with an example of a simulation.

```
...
parmat=matrix(c
(par.normd,chin,An,distn,rn,par.expdm,chix,Ax,distx,rx,par.gammad,chig,Ag,distg,rg,par
.logisd,chil,Al,distl,rl,par.lnormd,chilog,Alog,distlog,rlog,par.weibulld,chiw,Aw,dist
w,rw,par.cauchyd,chic,Ac,distc,rc,par.laplaced,chilap,Alap,distlap,rlap,par.unifd,chiu
,Au,distu,ru,par.paretod,chip,Ap,distp,rp),nrow=10,ncol=7,byrow=TRUE)

     sNorm=ifelse(min(as.numeric(parmat[,7]))==parmat[1,7],1,0)
     sExp=ifelse(min(as.numeric(parmat[,7]))==parmat[2,7],1,0)
     sGamma=ifelse(min(as.numeric(parmat[,7]))==parmat[3,7],1,0)
     sLogis=ifelse(min(as.numeric(parmat[,7]))==parmat[4,7],1,0)
     sLnorm=ifelse(min(as.numeric(parmat[,7]))==parmat[5,7],1,0)
     sWeibull=ifelse(min(as.numeric(parmat[,7]))==parmat[6,7],1,0)
     sCauchy=ifelse(min(as.numeric(parmat[,7]))==parmat[7,7],1,0)
     sLaplace=ifelse(min(as.numeric(parmat[,7]))==parmat[8,7],1,0)
     sUnif=ifelse(min(as.numeric(parmat[,7]))==parmat[9,7],1,0)
     sPareto=ifelse(min(as.numeric(parmat[,7]))==parmat[10,7],1,0)

     ind=data.frame(c
(sNorm,sExp,sGamma,sLogis,sLnorm,sWeibull,sCauchy,sLaplace,sUnif,sPareto))

     names(ind)='ind'

     ind

     }

srNorm=numeric(1000)
srExp=numeric(1000)
srGamma=numeric(1000)
srLogis=numeric(1000)
srLnorm=numeric(1000)
srWeibull=numeric(1000)
srCauchy=numeric(1000)
srLaplace=numeric(1000)
srUnif=numeric(1000)
srPareto=numeric(1000)
q=matrix(seq(1:20000),nrow=20,ncol=1000)
set.seed(0315)
for(i in 1:1000){
     q[,i]=rnorm(20,25,5)
     srNorm[i]=srankdiagnostic(q[,i])$ind[1]
     srExp[i]=srankdiagnostic(q[,i])$ind[2]
srGamma[i]=srankdiagnostic(q[,i])$ind[3]
srLogis[i]=srankdiagnostic(q[,i])$ind[4]
srLnorm[i]=srankdiagnostic(q[,i])$ind[5]
srWeibull[i]=srankdiagnostic(q[,i])$ind[6]
```

```
srCauchy[i]=srankdiagnostic(q[,i])$ind[7]
srLaplace[i]=srankdiagnostic(q[,i])$ind[8]
srUnif[i]=srankdiagnostic(q[,i])$ind[9]
srPareto[i]=srankdiagnostic(q[,i])$ind[10]

     }

sum(srNorm)
[1] 186
sum(srExp)
[1] 0
sum(srGamma)
[1] 161
sum(srLogis)
[1] 92
sum(srLnorm)
[1] 189
sum(srWeibull)
[1] 284
sum(srCauchy)
[1] 1
sum(srLaplace)
[1] 92
sum(srUnif)
[1] 57
sum(srPareto)
[1] 0
```

Appendix C - The Kolmogorov-Smirnov Statistic and Simulations

The traditional usage of the Kolmogorov-Smirnov test is to decide if data follow a fully specified distribution. Because no hypothesis testing is done in the function `diagnostic`, this statistic can be computed with estimated parameters. The statistic is computed as follows:

$$D = \max_{1 \le i \le n}\left(\hat{F}\left(X_i\right) - \frac{i-1}{n}, \frac{i}{n} - \hat{F}\left(X_i\right)\right) \tag{A.1}$$

where the cumulative probability function is calculated with estimated parameters, and *n* is the sample size. If data follow a particular distribution, this statistic should be small.

Table A.1 displays the four simulations when this statistic was included in the ranking procedure and when the ranking procedure used only the concordance correlation, goodness-of-fit, Anderson-Darling, and maximum distance. For the normal distribution and gamma distribution, the added Kolmogrov-Smirnov statistic made the ranking procedure perform worse than when it was omitted. For the Weibull distribution, the added statistic yielded approximately the same results.

Table A.1 Simulation Results Including Kolmogorov-Smirnov Statistic

	n=20 Norm (25,5)			n=50 Norm (25,5)			n=50 Gamma (2,1)			n=50 Weibull (4,20)	
	Rank.4	K-S		Rank.4	K-S		Rank.4	K-S		Rank.4	K-S
Normal	186	169		355	335		10	9		269	255
Exponential	0	0		0	0		0	0		0	0
Gamma	161	160		187	179		440	419		106	98
Logistic	92	83		149	156		4	4		109	110
Lognormal	189	188		87	87		127	124		18	18
Weibull	284	294		267	257		510	495		556	552
Cauchy	1	1		0	0		0	0		0	0
Laplace	92	95		28	29		1	3		15	19
Uniform	57	60		7	6		2	2		11	9
Pareto	0	0		0	0		0	0		0	0