National Aeronautics and
Space Administration

Lyndon B. Johnson Space Center
White Sands Test Facility
P.O. Box 20
Las Cruces, NM 88004-0020

ENTERED

August 27, 2014

RECEIVED

AUG 28 2014

NMED
Hazardous Waste Bureau

Reply to Attn of: RE-14-101

Mr. John E. Kieling, Chief
New Mexico Environment Department
Hazardous Waste Bureau
2905 Rodeo Park Drive East, Building 1
Santa Fe, NM 87505

Subject: Response to NMED Disapproval – NASA WSTF Soil Background Study Investigation Report

NASA submitted the Soil Background Study Investigation Report to NMED on March 27, 2014. NMED reviewed the report and issued a Notice of Disapproval (NOD) with comments on June 26, 2014. In the NOD, NMED directed NASA to address all comments in a response letter no later than September 1, 2014. This submittal provides the required response.

All four comments in the NOD pertained to the statistical evaluation of the soil background analytical data, which was performed by a subcontracted professional statistician. To fully address these comments, NASA requested that a detailed analysis of the comments be performed by the subcontracted statistician. The professional statistician evaluated NMED comments and prepared an independent report to address NMED's concerns.

Enclosure 1 provides a printed table that lists each numbered NMED comment and a summary of the full response to each comment. Enclosure 2 is a CD-ROM that includes an electronic copy of the table and hyperlinks to the full responses in the applicable sections of the statistician's report.

I certify under penalty of law that this document and all attachments were prepared under my direction or supervision in accordance with a system designed to assure that qualified personnel properly gather and evaluate the information submitted. Based on my inquiry of the person or persons who manage the system, or those persons directly responsible for gathering the information, the information submitted is, to the best of my knowledge and belief, true, accurate, and complete. I am aware that there are significant penalties for submitting false information, including the possibility of fine and imprisonment for known violations.

If you have any questions or comments, please contact Tim Davis of my staff at 575–524–5024.

Radel Bunker-Farrah
Chief, Environmental Office

2 Enclosures

cc:
Mr. Dan Comeau
New Mexico Environment Department
Hazardous Waste Bureau
2905 Rodeo Park Drive East, Building 1
Santa Fe, NM 87505

White Sands Test Facility Soil Background Study Investigation Report – Disapproval Comment Resolutions

August 2014

NMED Comment Number	Specific Comment	NASA Response
	NMED Overall Comment: Overall, there is a major concern that the statistics provide over-estimations of the background data and as ProUCL or another commercial software program was not used, the results cannot be reviewed to see what specific evaluations and decisions were made.	This overall comment is addressed with the following summary responses and detailed responses in the statistician's report that is Enclosure 2.
1	**NMED Comment:** The statistical estimates of the upper tolerance limits (UTLs) for the background data were not determined using ProUCL or other commercial software but rather a proprietary code. While it is noted that the statistician followed the ProUCL guidance, an independent look at the outputs and verification of the results could not be conducted.	**Detailed Response** **Summary Response:** While ProUCL is a substantial software resource to assist in the analysis of environmental data, like most software packages, it has limitations that leave gaps in a complete analysis strategy. R software provides great flexibility not only because it gives the user access to the many and growing number of freely available packages of methodologies, but also because it allows the development of needed methods that are not readily available. All statistical outputs presented in the report for the present soil study were calculated in R. While an outline of the analysis strategy was presented in that report, it lacked specific implementation details. These details have been provided, and a statistician familiar with R should be able to reproduce the statistical outputs in the report with this information.

NMED Comment Number	Specific Comment	NASA Response
2	**NMED Comment:** There is general concern that practically 100% of the UTLs are greater than the maximum detected background concentrations for all the background units. While it is anticipated that UTLs could be greater than the maximum detected, it is not typical to see 100% of the UTLs greater than background with many of them being two to three times greater than the maximum detected concentration. It appears that there is unaccounted for bias in the data, possibly the inclusion of outliers or highly skewed data. Inclusion of an outlier would result in a high UTL. Discussion of whether outliers were evaluated and excluded is warranted as well as a discussion as to why all the UTLs are high (and in many cases two to three times higher and a few greater than three times higher) compared to the maximum detected concentration.	**Detailed Response** **Summary Response:** Logically it makes sense that, for small samples, any UTL achieving at least 95%-confidence for 95%-coverage will generally be larger than the maximum sample data value – since the expected value of the highest order statistic will not even reach the 95th percentile of the population's distribution when the sample size is small. Simulations also demonstrate that, for small samples, UTL's often exceed the maximum data value in the sample used to calculate the UTL – sometimes by a factor of more than three for 95%-confidence with 95%-coverage, and sometimes by more than a factor of five or even six for 99%-coverage. The commonly discussed nonparametric method of using the highest or second-highest order statistic would ensure that the UTL does not exceed the highest sample value. However, this method requires substantially large sample sizes –much larger than is available in the present study. Additionally, with the required larger samples comes a larger expected maximum sample data value since better representation of the population is achieved with bigger samples. Regarding outliers, it is important to realize that available outlier detection methods tend to produce large numbers of false detections. Thus, available methods tend to flag relatively extreme observations as outliers even though they often are part of the primary underlying population of interest and provide unique and valuable information about that

NMED Comment Number	Specific Comment	NASA Response
		population. It is also important to realize that most of the available methods make assumptions about the underlying population's distribution, and in particular, most of them assume normality. Hence the advice of most statistical experts is to check observations that are flagged by outlier detection methods, but not to discard them in analyses unless they are found to have identifiable coding or sampling anomalies. Dixon's Test was used to examine outliers in the present soil study, as well as strip plots for visual inspection, and extreme values were checked for potential anomalies.
3	**NMED Comment:** For the statistical calculations of the UTLs, it appears the data were "forced" into one of the four distribution types: normal, lognormal, gamma, or exponential. It is not clear why the data distributions were limited to only these four types, rather than allowing the program (or ProUCL) to determine the best-fit distribution. Further, it appears that even if the p-values were greater than 10%, when nonparametric distributions may be applied, the data were still forced into one of the four distributions. Discuss how outliers also affect the chosen distribution, as it does not appear that any tests were conducted for outliers or any data excluded as an outlier. Provide discussion of this issue.	**Detailed Response** **Summary Response:** In the analysis of the present soil study, determination of the best characterizing distribution for a population was not forced into any particular type, but followed the same process other software does in determining a best fitting distribution. To evaluate this, it is important to understand the null and alternative hypotheses in the GOF tests, and the fact that a low p-value implies that the distribution being tested is not appropriate for characterizing the population, while high p-values imply that the distribution is potentially a viable distribution for describing the population. If all p-values are low, this implies none of the considered distributions are viable, and nonparametric methods were then considered for further analyses.

NMED Comment Number	Specific Comment	NASA Response
4	**NMED Comment:** Similar to above, it appears that if none of the four distributions fit, an upper prediction level (UPL) was calculated and used as the UTL. The prediction level is an estimate of what a future value will be while the tolerance limit is representative of a population characteristic. Typically the UPL is used to compare data to background (such as compliance monitoring) and not to establish a soil background range. Justify why the UPL is an adequate substitution for a UTL, and why the UPL is more appropriate than using non-parametric or other statistical methods for estimating the UTL. Also, it is not clear from the summary tables if the result is a UPL or a UTL. Provide discussion of this issue.	**Detailed Response** **Summary Response:** Due to the small sample size, no standard nonparametric methods for calculating a UTL were viable. Using a bootstrap methodology with a nonparametric formulation for calculating a UPL provides a very conservative calculation for a UTL. While the resulting UTL value is quite large, it is the only available nonparametric option given the limited sample sizes.

Response to NMED 2014-Jun-26 Letter Regarding:
Notice of Disapproval – Soil Background Study Investigation Report

August 18, 2014

Prepared for:
Pamela Egan
Navarro Research and Engineering, Inc.
NASA-White Sands Test Facility
P.O. Box 20
Las Cruces, NM 88004
(575) 524-5351

Prepared by:
David L. Daniel, Ph.D.
Chief Statistician,
The Data Toolbox Company
2324 Tuscan Hills Lane
Las Cruces, NM 88011
(575) 644-7499

Introduction

On 2014-Mar-21, I provided a report to NASA White Sands Test Facility (NASA WSTF) titled *Statistical Development of Soil Background Concentrations*. This report was used in NASA WSTF's report to the New Mexico Environmental Department (NMED) titled Soil Background Study Investigation Report, which NMED received on 2014-Mar-31. NMED responded to the NASA WSTF with a Notice of Disapproval, and enumerated four comments related to the statistical development, to which NMED requested responses. This document provides those responses.

Response to Comment #1

Comment #1 states:

> The statistical estimates of the upper tolerance limits (UTLs) for the background data were not determined using ProUCL or other commercial software but rather a proprietary code. While it is noted that the statistician followed the ProUCL guidance, an independent look at the outputs and verification of the results could not be conducted.

While ProUCL Version 4.1.00 provides access to a substantial number of statistical methodologies for addressing environmental analyses, it doesn't have a sufficiently complete repertoire of methods to provide a gapless analysis strategy. For example, while it provides Regression on Order Statistics (ROS) methodology as a means for imputing values where there are non-detects (ND's), the authors of the ProUCL Version 4.1.00 Technical Guide have cautioned against using ROS. Also, Section 4.3.5.4 of the technical guide acknowledges the circular problem of performing ROS for the gamma distribution, where (unlike the case for the normal or lognormal distributions) the gamma parameters must first be estimated in order to do the ROS, but to estimate parameters the full data set is needed. They refer the reader to Singh, Singh, and Iaci (2002) for details, but this paper does not address issues with ND's. A similar gap is present in the goodness of fit test for the gamma distribution. A solution to this is to the Expectation Maximization Algorithm (EM) or the Markov Chain Monte Carlo methods for determining values to impute in place of the ND's (see Section 13.3.2 in Helsel, 2012), but neither of these are implemented in ProUCL.

ProUCL is primarily focused on providing upper confidence limits (UCL's) and secondarily on upper prediction limits (UPL's). It does, however, have a weak strategy for addressing UTL's. In particular, there are two reasons this is true.

First, ProUCL's use of the Gamma ROS (GROS) method to implement a parametric methodology when ND's are present is lacking because it attempts to estimate the gamma parameters with only the detected observations. To illustrate the difference, some simulations were conducted using each of three different gamma distributions whose parameters were selected to typify the estimated gamma distributions in the present soil study (see Figure 1 for these parameter combinations plotted amongst most of the soil study's parameter combinations that were estimated for samples that appeared to be from gamma distributions). These gamma distributions are:

1. Gamma(3, 3),
2. Gamma(7, 1), and
3. Gamma(1, 0.2),

and Figure 2 shows their probability density functions (pdf's). From this plot one can see that the second distribution is approaching symmetry and somewhat approximates a normal distribution. Figures 3a–3c show some results of the simulations, implementing both the Gamma ROS method and the EM methodology. In the simulations, samples of size n=12 (the same sample size as in our present soil study) were generated with the lowest three values being replaced by ND's. In Figures 3a–3c, the pdf of the original gamma distribution from which simulation data were generated is shown with a black line-type with circles on it; eight pdf's for gamma distributions whose parameters were estimated with the EM method are displayed in a solid green line-type; and eight pdf's for gamma distributions whose parameters were estimated with the GROS method are displayed in a blue line-type with x's on it. The eight gamma distributions that are displayed for each of the estimating methods were chosen by sorting the estimated parameter sets first by the shape estimate and then by the rate estimate (which is equal to one divided by the scale parameter estimate), and the eight parameter sets were chosen to be equally spaced in the sorted listing. This spacing helps in providing a reasonably broad representation of the estimates in the simulation for each method. In each of Figures 3a–3c, the pdf's obtained by EM method more closely approximated the original pdf than the pdf's obtained from the GROS method. Not only do the simulations show better representation, but the GROS method is intuitively flawed because it doesn't attempt to take into account the ND's at the stage that the gamma parameters are estimated, which will necessarily distort the estimates. The simulation code is available upon request, and it specifies a random number seed so that the results can be reproduced.

Second, when nonparametric methods are called for, ProUCL's sole use of high order statistics as a UTL is limiting since it can require fairly large sample sizes to obtain any estimate at all. ProUCL does provide percentile-based bootstrap methods for estimating UTL's, but with small sample sizes, typically the result is the same – select the maximum sample value, so that it has the same limitation (note that bootstrap methods should not be used in such cases as they give a false representation of the confidence level for the UTL). There are not many other nonparametric options, so this is not a direct shortcoming of ProUCL, but does compound the issue of not providing better methods for gamma-distributed populations when ND's are present. This is because when a sample that comes from a gamma distribution cannot be properly identified as such because of poor parameter estimates, only nonparametric options remain.

ProUCL also has very limited outlier detection methods – Dixon's Test and Rosner's test, and the ProUCL Technical Guide points to a number of other methods that it does not implement. Outlier detection methods, in general, tend to produce a large number of false positives or false negatives, so the lack of inclusion of supposedly more sophisticated methods is not a substantial issue. As a whole, ProUCL is a substantial software resource that provides access to many statistical methodologies useful in environmental analyses. However, it will likely never provide all of the most current methodologies that are available or even desirable and, hence, there will always be a need for other software resources to fill these gaps.

The UTL's and other calculations conducted for the present soil study were performed using R software (Version 3.1.0). R is open source and available on every mainstream computing platform. As such, use of R has become very wide spread for doing statistical modeling, conducting hypothesis tests, obtaining statistical estimates, and creating graphics. R is taught and used in almost every university statistics department in the country, is a very commonly requested skill in job advertisements for statisticians, and was listed as the single highest paying information technology skill out of 200 skills surveyed by Dice.com in January of 2014. Its adoption is growing rapidly not only among statisticians, but in other areas such wildlife science, biology, environmental science, and even linguistics. One of the strengths of R is the large library of packages that have been developed to

provide various functionalities. Most of the statistical outputs given in the report on the present soil study were calculated using the packages freely available on the Comprehensive R Archive Network (CRAN) web site, http://cran.r-project.org. While the soil study report provided an outline of the logic used in conducting the statistical analyses, it did fail to give specifics on the R packages used and the specifics of the methodologies used within those packages. Herein such details are provided. Where freely available R packages were not used, more implementation details are provided.

Data imputation for ND's was performed using an EM methodology. For this methodology, a method in a CRAN package was modified. In particular, the *fitdistr()* method in the *MASS* R package was modified to update the estimates of the imputed data values based on the most recent updates of the parameter estimates in each iteration of the estimation process. Code for this new method, named *fitdistr.nds()*, is provided in Appendix A, and is available electronically upon request. If this process did not converge for a particular distribution, UTL methodologies based on that distribution were deemed "not viable" and excluded from the selection process and no goodness of fit (GOF) test was performed for that distribution. If it did converge, then distribution GOF tests were conducted using the sample data with the newly imputed values using the *shapiro.test()* method in the *stats* package for normal and lognormal distributions; for gamma distributions using the *ks.test()* method with argument *y="pgamma"* and also arguments *shape* and *rate* set to the estimated gamma parameters – also in the *stats* package; and with *gofExp.test()* in the *Renext* package for exponential distributions. Once a distribution was chosen based upon the results of the GOF tests, UTL's were calculated using *normtol.int()* in the *tolerance* package for normal and lognormal distributions (and the UTL was back-transformed in the case of a lognormal distribution), and using *gamtol.int()* in the same package for gamma distributions. As the UTL's are one-sided tolerance calculations, the *method* argument in each of these methods is not relevant. For a nonparametric UTL, the custom method *bootstrap.km.percent()* was used with argument *num.samples* set to 2,000. The code for this and the other method it requires are listed in Appendix B, and is available electronically upon request. More about this technique and the limitations of nonparametric UTL's in the present soil study are presented in the Section titled *Response to Comment #4* below.

Given these implementation details and the process outline provided in the original report, verification of the results should now be possible. However, due to the complex nature of the implementation, verification may require the efforts of a statistician familiar with R. Parametric-based tolerance intervals calculated in ProUCL should give essentially identical results when provided with the data imputed for the ND's via the EM method. Requests for any further information about the process or methodology are welcome.

Response to Comment #2

Comment #2 states:

> There is general concern that practically 100% of the UTLs are greater than the maximum detected background concentrations for all the background units. While it is anticipated that UTLs could be greater than the maximum detected, it is not typical to see 100% of the UTLs greater than background with many of them being two to three times greater than the maximum detected concentration. It appears that there is unaccounted for bias in the data, possibly the inclusion of outliers or highly skewed data. Inclusion of an outlier would result in a high UTL. Discussion of whether outliers were evaluated and excluded is warranted as well as a discussion

as to why all the UTLs are high (and in many cases two to three times higher and a few greater than three times higher) compared to the maximum detected concentration.

For small sample sizes, it is not only common, but it is typical for a UTL to exceed the maximum value in a data set – often by a substantial amount. Consider a sample of size n=12 from a normal distribution. The expected value of the maximum for this sample is approximately at the 92nd percentile (approximately given by: $1 - 1/(n+1) \approx 0.92$) of the normal distribution. Not only does this maximum value not give 95% coverage, but it also does not provide the additional buffer needed to obtain 95% confidence.

To further illustrate this, simulations were conducted in the R software (Version 3.1.0). In the simulations, samples of n=12 were drawn from a distribution, and the 95%-confidence 95%-coverage UTL's were calculated. The UTL of the sample was scaled by the maximum data value for the sample – hence, the values studied are: how many times bigger the UTL is than the maximum value in the sample. This was done for each of three different gamma distributions whose parameters were selected to typify the estimated gamma distributions in the present soil study – the same distributions as for the simulations discussed in the Section titled *Response to Comment #2* (Figure 1 illustrates these parameter combinations plotted amongst most of the soil study's parameter combinations that were estimated for samples that appeared to be from gamma distributions and Figures 3a – 3c shows their pdf's). From this plot one can see that the second distribution is approaching symmetry and somewhat approximates a normal distribution. Figure 4a shows histograms of these UTL's scaled by the maximum value in the sample. It shows that *very few* of the UTL's calculated were as small as the maximum value in the sample from which it was calculated – 0.33% in all. In total, for the 95%-confidence 95%-coverage UTL's, 9.6% were at least two times the maximum sample value, 1.6% were at least one and a half times the maximum sample value, and 0.1% were at least three times the maximum sample value. 95%-confidence 99%-coverage UTL's were also calculated in these simulations and are shown in Figure 4b, and are even more extreme in order to get the greater coverage. For the 95%-confidence 99%-coverage UTL's, 54.9% were at least two times the maximum sample value, 29.2% were at least one and a half times the maximum sample value, and 14.2% were at least three times the maximum sample value. None of the 99%-coverage UTL's were less than their maximum sample value. Table 1 gives the minimum, maximum, and average of these scaled UTL's for each distribution. For the 95%-coverage UTL's, the average ranged from 1.33 to 1.77, while for the 99%-coverage UTL's, the average ranged from 1.72 to 2.90. Further simulations indicated that it would require sample sizes in the range of 45-65 before the 95%-confidence 95%-coverage UTL's would typically exceed the maximum data value in the sample only 50% of the time. This simulation code is also available upon request, and it also specifies a random number seed so that the results can be reproduced.

In addition to the small sample size issue, perhaps some of the confusion that commonly arises over the magnitude of UTL's relative to the sample data stems from certain nonparametric UTL calculations. A commonly cited formulation for obtaining a nonparametric UTL is to use the largest or second largest sample value for the UTL. However, like most nonparametric methods, this formulation has requirements that are often overlooked or misstated. In particular, in order to use the maximum value in a data set as the 95%-confidence 95%-coverage UTL, a minimum sample size of n=59 is required, and for 95%-confidence 99% coverage the minimum sample size is much larger – n=299 (Hahn and Meeker, 1991, Eq. 5.4). This confusion may be compounded by the fact that the example in Section 3.4.5.4 of the ProUCL Version 4.1.00 Technical Guide (Draft), is incorrect – giving a hugely over-stated value for some of the "Achieved Confidence" levels in Table 3-2, and implying that UTL's with high coverage and a high confidence level can be achieved from small

sample sizes. The reason the example is incorrect is that the cumulative binomial formula above Equation 3-10 at the bottom of page 87 and the statement of how to use it are incorrect. First, the summation should be from i=0 to r-1, not from i=0 to r; the reasoning is that the r^{th} order statistic must *exceed r-1 observations*. Equivalently, it could be written as one minus the cumulative binomial formula summed over r to n – again with the reasoning being that it has to be above r-1 observations. This incorrect formulation is how the authors arrived at the incorrect and extremely over-stated values of "Achieved Confidence" in Table 3-2 of the ProUCL Technical Guide (except that the second value of 92.2% achieved confidence has had a digit dropped – according to the authors' formula it should be 92.02%). The stated confidence level and the correct confidence levels are as follows:

Stated confidence:	96.7%,	92.2%,	100%,	100%
Correct confidence:	88.5%,	70.8%,	70.8%,	21.4%

The correct values listed here were calculated using a cumulative binomial formula in its correct form. As verification of their correctness, the formula in Equation 5.4 of Hahn and Meeker (1991) indicates that the best confidence that could be obtained, using the highest order statistic, for p=99% coverage with n=24 is:

$$\text{confidence} = 1 - p^n = 1 - (0.99)^{24} = 0.2143, \text{ or } 21.43\%,$$

which is indeed the fourth of the above correct confidence levels given. For 95%-coverage, the formula gives:

$$1 - p^n = 1 - (0.95)^{24} = 0.7080, \text{ or } 70.80\%,$$

which is the third value listed above for correct confidence levels. Further, the statement in the ProUCL Technical Guide that, "r … is chosen such that the cumulative binomial probability … becomes as close as possible to $(1 - \alpha)$," is incorrect. Rather, it is necessary for the cumulative binomial probability to equal or exceed $(1 - \alpha)$. The way the authors have stated it would imply that regardless of sample size, one can find a value of r such that the r^{th} order statistic will suffice as a UTL with the specified coverage, but this is not at all true. In fact, in order for the highest order statistics in a sample to obtain 95%-confidence with 95%-coverage as the UTL, the sample size must be (rearranging Equation 5.4 from Hahn and Meeker, 1991):

$$n = \log(1 - \text{confidence}) / \log(p) = \log(0.05) / \log(0.95) = 58.40, \text{ so } n=59,$$

and for 99% coverage:

$$n = \log(1 - \text{confidence}) / \log(p) = \log(0.05) / \log(0.99) = 298.07, \text{ so } n=299.$$

The ProUCL Version 5.0.00 Technical Guide has removed the incorrect calculations and now correctly shows only the coverage given by using the highest order statistic, but it's cumulative binomial formula with accompanying statement of how to use it is still incorrect. It is worth noting that the text by Hahn and Meeker (1991) that was referenced above is a well-known and commonly referenced source regarding various statistical intervals.

Also, while this nonparametric method will yield a UTL that is no more than the largest sample observation value, an often overlooked issue associated with it is that, due to its required high sample size, the expected value of the highest and second-highest order statistics for the sample will increases

substantially with the increased sample size. Hence, while this method has the appeal of getting a UTL that is less than or equal to the maximum sample value, it is often the case that as the sample size is increased, the relevant parametric method begins to yield a UTL that is less than the highest sample value before enough samples are obtained to make this nonparametric method valid.

When no suitable distribution can be identified for a sample, a nonparametric method is needed. However, without a sufficiently large sample size, the nonparametric method of using a high order statistic for the UTL is not viable. As stated above in the response to Comment #1, the ProUCL Version 4.1.00 software provides no nonparametric solution in this case. Its technical guide gives little useful guidance in this situation, though it discusses techniques that hint at a solution, with the solution being to bootstrap another nonparametric upper limit, such as an upper prediction limit (UPL). With small sample sizes such as those in the present study, the specific implementations of this method almost inevitably always yield a UTL estimate that is equal to the maximum value in the sample. This UTL formulation will almost never achieve 95% coverage 95% of the time, and hence is inadequate. This issue is discussed further in the response to Comment #4.

Regarding outliers, ProUCL offers only two tests for their detection – Dixon's Test and Rosner's Test. Section 7.1.1 of the ProUCL Technical Guide states that Dixon's Test can be used for $n \leq 25$, while Section 7.1.2 states that Rosner's Test can be used for $n \geq 25$. However, both of these tests assume normality of the primary underlying populations being tested – aside from the outlier. Most of the samples in the present study do not appear to come from normal distributions. Also, in Section 7.1 of the ProUCL Technical Guide, the authors state that both of these methods suffer from masking effects. They refer the reader to "more effective robust outlier identification procedures (Singh and Nocerino, 1995)," which they state "are beyond the scope of ProUCL 4.0." The primary technique implemented in the referenced paper by Singh and Nocerino (1995) is a method referred to as the PROP robust method ("PROP" apparently being an abbreviation for "Proposed") based on the PROP influence function from another paper. The authors state that the PROP method "works quite effectively at identifying multiple outliers in univariate as well as multivariate data sets of all sizes," and that "no tuning constants except for an appropriate choice of an α-value, are needed." However, this method, like virtually all outlier detection tests, depends on an assumption about the distribution of the data. In particular, it too assumes normality of the distribution of the primary underlying population from which the sample came from, excepting the outliers.

As our samples in the present soil study have n=12, both Dixon's Test and the PROP method were considered. A primary concern about the identification of outliers is that one cannot know how to define an outlier without knowing what the underlying distribution is. Skewed distributions will generally produce the occasional value from the tail of the distribution that may appear to be an outlier when compared to a distribution that is less skewed. Hence, the detection of outliers is a bit of a circular problem. That is, more extreme values are only outliers relative to a particular distribution, but those extreme values will not be out of the ordinary for some other distributions with a longer tail, yet the underlying distribution generally cannot be known in advance. Hence, it becomes virtually impossible to determine if extreme values are part of a primary distribution that is somewhat skewed, or if they should be considered outliers because they come from a secondary distribution and are contaminating the sample by mixing with observations came from a primary distribution of interest. This is why most statisticians and statistics texts recommend not removing outliers unless one can demonstrate that such observations were miscoded or had some other identifiable reason to be anomalous, such as a change in the experimental conditions or sampling methods (see, for example,

Kutner, et. al, 2004, Section 3.3 and Ofungwu, 2014, Section 6.6). Additionally, the EPA guidance states (US EPA, 2009, Section 6.3.3):

> If either Dixon's or Rosner's test identifies an observation as a statistical outlier, the measurement should not be treated as such *until* a specific physical reason for the abnormal value can be determined. ... If no error in the value can be documented, it should be assumed that the observation is a true but extreme value. In this case, it should not be altered or removed. However, it may helpful to obtain another observation in order to verify or confirm the initial measurement.

Simulations were conducted to evaluate the ability of both Dixon's Test and the PROP method, again utilizing the three gamma distributions discussed earlier as they are reasonably typical of the estimated distributions for the present study. Each simulation conducted N=1,000 replications, with each replication generating twelve random observations from a specified distribution, and then tested the sample for outliers using both detection methods. Results of the simulations are reported for each of the three distributions in Table 2. Dixon's Test simply tests whether the highest value is an outlier, while the PROP method gives two cut-off values – one for "potential outliers" and a second for "clear outliers." Hence, the numbers of samples having outliers are reported separately for Dixon's Test, for the PROP method potential outliers, and for the PROP method clear outliers. Since the PROP method also flags specific observations as potential or clear outliers, the number of each category of outlier is also reported. Table 2 indicates that, for the three simulation distributions, Dixon's test flagged between 15.7% and 38.1% of the samples as having outliers despite the fact that the data within each sample were drawn from the same population. Hence, these percentages represent the percent of false positives from Dixon's test. The PROP method flagged between 29.0% and 41.8.4% of the samples as having potential outliers, and flagged 68.7% to 88.7% of the samples as having clear outliers, again representing the percent of the samples having false positives. Hence, despite the fact that the observations in a sample were drawn from the same population, both methods flagged large percentages of samples as having outliers. While this may be due in part to the fact that the underlying distributions are not normal distributions, keep in mind that distribution #2 (Gamma(7, 1)) is fairly symmetric and its density function somewhat approximates a normal density function, yet it had the highest number of samples reported to have PROP potential outliers and the highest average number of potential outliers per sample (0.484); and the second highest number of samples reported to have PROP clear outliers (71.7%) as well as the second highest average number of clear outliers per sample (2.119). Bar charts showing the distribution of the PROP potential outliers and PROP clear outliers for each of the distributions are shown in Figures 5a–5c. Here too, this simulation code is also available upon request and it also specifies a random number seed so that the results can be reproduced.

Despite both detection methods having severely high false positives with our test distributions, being overly aggressive with detecting outliers from homogeneous samples, Dixon's Test was applied to the sample data sets in the present soil study. Dixon's Test was chosen because (1.) its false detection rate, while still extremely high, appears to be somewhat lower than for the PROP method; (2.) it is discussed and implemented by ProUCL; and (3.) in the PROP method, the convergence values of the parameter estimates and the calculated values used to identify outlying observations varied substantially depending on what starting values were used in the method. Results of Dixon's Test for samples flagged as having an outlier at a 5% significance level are shown in Tables 3a–3c (including the p-values), and strip charts showing the data for these samples (scaled by their maximum value so that the pattern of all observations in each sample is spread out enough to be discernable) are displayed in Figures 6a–6d. Dixon's Test identified 139 of the 510 samples as having an outlier, which is 27.25% of the samples. This percentage is within the range of false detections it flagged for in the

simulations, so it is not a surprising number even if there is no contamination of any of the populations (although it should be noted that there are numerous samples where all the observations have the same value, including having all non-detects, which trivially did not reject The Dixon's Test null hypothesis of no outliers).

Note that the data used in Dixon's Test are the original data, without imputed values (such as the values used to replace non-detects that are obtained by the ROS method or, as was used in the present study, the EM Algorithm). Hence, some of the samples were likely flagged by Dixon's Test because of one or two detected values among numerous non-detects, which are recorded as zeros. In examining Figures 6a–6d, several samples stand out as having potential outliers that could perhaps be meaningful. They include Sample ID's: 22, 23, 24, 27, 43, 77, 97, 106, and 137. Review of the highest observations in each of these samples for possible anomalous sampling conditions and proper data entry has been conducted. Having been unable to identify any such anomalies, the advice given in Kutner, et. al. (2004, Section 3.3), Ofungwu (2014, Section 6.6), EPA guidance (US EPA, 2009, Section 6.3.3) and many other statistical resources – to leave the observations in for analysis purposes as they may represent the truth about the underlying population – has been followed.

Response to Comment #3

Comment #3 states:

> For the statistical calculations of the UTLs, it appears the data were "forced" into one of the four distribution types: normal, lognormal, gamma, or exponential. It is not clear why the data distributions were limited to only these four types, rather than allowing the program (or ProUCL) to determine the best-fit distribution. Further, it appears that even if the p-values were greater than 10%, when nonparametric distributions may be applied, the data were still forced into one of the four distributions. Discuss how outliers also affect the chosen distribution, as it does not appear that any tests were conducted for outliers or any data excluded as an outlier. Provide discussion of this issue.

The null and alternative hypotheses for a GOF test of a particular distribution type is

H_0: The population sampled from follows the specified distribution.
H_a: The population sampled from does not follow the specified distribution.

As is usual in hypothesis testing, the null hypothesis is rejected when the p-value calculated from the test statistic is less than or equal to the significance level, α. Consequently, when the p-value for a GOF test is more than our significance level of 10%, the null hypothesis is *not* rejected, thus making the specified/test distribution appear to be a viable option. It is when the p-value is less than or equal to the 10% significance level that the tested distribution is deemed not appropriate. Hence, nonparametric methods would only be indicated if all of the GOF tests have p-values less than or equal to 10%, and all of their specified distributions are thus rejected, leaving no viable distribution options. If only one distribution has a GOF test p-value greater than 10%, then that distribution is the only viable distribution, so it is assumed to be the distribution most appropriate to characterize the population. If, however, multiple GOF tests have p-values greater than 10%, then the distribution having the highest GOF p-value is assumed to be the best distribution to characterize the population (since the GOF with the highest p-value is the furthest from indicating its respective distribution is

rejected as being the population's characterizing distribution). This process is essentially what every statistical software package does to arrive at a "best fitting distribution," including ProUCL. The present soil study examined four possible distributions. Examination of the exponential distribution was technically unnecessary since it is a special case of the gamma distribution when the shape parameter is equal to one. However, the exponential was specifically examined because it is easier to work with (in terms of estimation and testing) than the gamma distribution since it has fewer parameters that need to be estimated. That is, if a population's distribution was determined to be the exponential distribution, it would probably also indicate that a gamma distribution was viable as well, but it would be easier to use methodology specific to exponential distributions than to have to estimate both parameters of the gamma distribution.

Ultimately, when GOF p-values were greater than the 10% significance level, the population's assumed distribution was not forced into one of the four distributions considered but, rather, this situation indicated that one or more of the distributions were not rejected as being the population's characterizing distribution, and therefore were considered as viable distributions to be used in further analysis of the sample data. Also, ProUCL considers exactly the same distributions as what was considered in the present soil study, except that it does not examine the exponential except indirectly through its consideration of the gamma distribution. Discussion of outlier detection and follow-up was presented in the above Section titled *Response to Comment #2*. In particular, an observation should be considered an outlier if it is extreme relative to the primary underlying distribution. However, for common distributions with long tails, outlier detection methods tend to have large numbers of false detections. That is, they tend to identify "relatively extreme" observations as outliers even though they were actually sampled from the primary underlying distribution and therefore provide useful information about the primary underlying distribution. Hence, most statistical texts that address outlier detection provide the advice of reviewing potential outliers for coding errors and sampling inconsistencies, but leaving those observations in the data sample for further analysis unless such an anomaly is discovered. This has been done in the present soil study.

Response to Comment #4

Comment #4 states:

> Similar to above, it appears that if none of the four distributions fit, an upper prediction level (UPL) was calculated and used as the UTL. The prediction level is an estimate of what a future value will be while the tolerance limit is representative of a population characteristic. Typically the UPL is used to compare data to background (such as compliance monitoring) and not to establish a soil background range. Justify why the UPL is an adequate substitution for a UTL, and why the UPL is more appropriate than using non-parametric or other statistical methods for estimating the UTL. Also, it is not clear from the summary tables if the result is a UPL or a UTL. Provide discussion of this issue.

When the p-values of the GOF tests for each of the four distributions considered were less than or equal to the 10% significance level, the analysis process dictated that nonparametric methods for calculating the UTL were appropriate. However, using Equation 5.4 in Hahn and Meeker (1991), it was calculated that for the small sample size of n=12 and for a 95%-confidence level, the amount of coverage that would be obtained by using the sample's largest data value, $x_{(12)}$, is:

$$p = (1 - \text{confidence})^{1/n} = (1 - 0.95)^{1/12} = 0.779, \text{ or } 77.9\%,$$

which is far short of the desired 95% and 99% coverage levels. Hence alternative methods were considered. As was calculated earlier in the Section titled *Response to Comment #1*, it would require a sample of n=59 to achieve 95% coverage, and a sample of n=299 to achieve 99% coverage – each using the highest data value in the sample of that size as the UTL. As this was not feasible, other approaches to a nonparametric UTL had to be investigated. With the recent release of ProUCL Version 5.0.00, its associated technical guide has added four subsections to Section 3.4.4, the last two of which describe the calculation of nonparametric UTL calculations based upon bootstrapping estimates the percentiles. This was also the second option discussed in *Appendix A – A Comment on Choices for Nonparametric UTL's* in the 2014-March statistical analysis report on the present soil study. However, as discussed in that Appendix, the sample size in the present study is not sufficient to do anything more than give the highest data value in the sample, and one must be careful to recognize this as an intrinsic limitation in the bootstrap methodology and not to conclude that the initially specified confidence and coverage are correct simply because the method will calculate a number of the UTL. In reality, the coverage will be the same as for the method that directly uses the sample's highest data value, that is, 77.9% in for the present study's sample size of n=12.

This left no standard nonparametric techniques for calculating a UTL. However, UPL's and UTL's are not completely unrelated. A 95%-UPL is an estimate of an upper bound on the next occurring single sample value and thus will have a 95% chance that the random sample drawn to use for the calculation will be sufficiently representative of the population, and that the future observation is not so extreme, so that the calculated UPL will indeed bound the future observation. There are also UPL's that attempt to capture *k* future observations instead of a single future observation – these are discussed in Section 3.5.5 of the ProUCL Version 5.0.00 Technical Guide. In a somewhat similar manner, a 95%-confidence 90%-confidence UTL has a 95% chance that the sample it is calculated from will yield an interval that will provide an upper bound for 90% of the population – including future observations. If a UPL for a single future observation could be constructed so that it provided an upper bound for 90% of all future single observations with 95% confidence, then it would be serving the same function as a 95%-confidence 90%-confidence UTL. And that is one of the primary objectives of what bootstrapping attempts to do. The caveat to this approach is that it *may* not be as efficient as other mechanisms for calculating a UTL, but given that there are no other nonparametric options available, it was the only remaining choice. To construct the UTL in a nonparametric manner, the UPL that was used is that given in Equation 5.2 of the ProUCL Version Technical Guide, which is based upon the Chebyshev inequality, and KM estimates were used as best as possible to obtain the sample statistic quantities specified in the formula (see Appendix A of the original analysis report for some additional comment on this). This method yields a relatively large UTL – often approximately 3.5 times the size of the maximum data value. But earlier calculations show that the maximum data used as a UTL would only provide 77.9% coverage, and there are no other apparent options for constructing a nonparametric UTL without additional sample data.

It should be noted that in Appendix A of the original analysis report, the formulas given in Equations (1) and (2) are missing a "power of 2" on the $\hat{\sigma}$ term. Compare this term in Equation (1) with the s_x term in Equation 5.2 of the ProUCL Technical Guide, where the s_x term will have a power of two when brought under the square-root radical. However, the formula used in the calculation code has been checked and found to be correct. The R software code used to compute this UTL is provided in Appendix C, and is available electronically upon request. The method called to calculate the UTL is *bootstrap.km.chebyshev.percent()* and it was called with the argument *num.samples* set to 2,000.

Summary of Responses

Comment #1

While ProcUCL is a substantial software resource to assist in the analysis of environmental data, like most software packages, it has limitations that leave gaps in a complete analysis strategy. R software provides great flexibility not only because it gives the user access to the many and growing number of freely available packages of methodologies, but also because it allows the development of needed methods that are not readily available. All statistical outputs presented in the report for the present soil study were calculated in R. While an outline of the analysis strategy was presented in that report, it lacked specific implementation details. These details have been provided, and a statistician familiar with R should be able to reproduce the statistical outputs in the report with this information.

Comment #2

Logically it makes sense that, for small samples, any UTL achieving at least 95%-confidence for 95%-coverage will generally be larger than the maximum sample data value – since the expected value of the highest order statistic will not even reach the 95^{th} percentile of the population's distribution when the sample size is small. Simulations also demonstrate that, for small samples, UTL's often exceed the maximum data value in the sample used to calculate the UTL – sometimes by a factor of more than three for 95%-confidence with 95%-coverage, and sometimes by more than a factor of five or even six for 99%-coverage. The commonly discussed nonparametric method of using the highest or second-highest order statistic would ensure that the UTL does not exceed the highest sample value. However, this method requires substantially large sample sizes –much larger than is available in the present study. Additionally, with the required larger samples comes a larger expected maximum sample data value since better representation of the population is achieved with bigger samples.

Regarding outliers, it is important to realize that available outlier detection methods tend to produce large numbers of false detections. Thus, available methods tend to flag relatively extreme observations as outliers even though they often are part of the primary underlying population of interest and provide unique and valuable information about that population. It is also important to realize that most of the available methods make assumptions about the underlying population's distribution, and in particular, most of them assume normality. Hence the advice of most statistical experts is to check observations that are flagged by outlier detection methods, but not to discard them in analyses unless they are found to have identifiable coding or sampling anomalies. Dixon's Test was used to examine outliers in the present soil study, as well as strip plots for visual inspection, and extreme values were checked for potential anomalies.

Comment #3

In the analysis of the present soil study, determination of the best characterizing distribution for a population was not forced into any particular type, but followed the same process other software does in determining a best fitting distribution. To evaluate this, it is important to understand the null and alternative hypotheses in the GOF tests, and the fact that a low p-value implies that the distribution being tested is not appropriate for characterizing the population, while high p-values imply that the

distribution is potentially a viable distribution for describing the population. If all p-values are low, this implies none of the considered distributions are viable, and nonparametric methods were then considered for further analyses.

Comment #4

Due to the small sample size, no standard nonparametric methods for calculating a UTL were viable. Using a bootstrap methodology with a nonparametric formulation for calculating a UPL provides a very conservative calculation for a UTL. While the resulting UTL value is quite large, it is the only available nonparametric option given the limited sample sizes.

As always, if you have any questions or comments, please let me know.

Figure 1. Scatterplot of most of the parameter estimate combinations determined for gamma distributions used in the soil study (just under 30% of the parameter combinations are beyond the bounds of the plot). Square boxes indicate the parameter combinations used in the simulations.

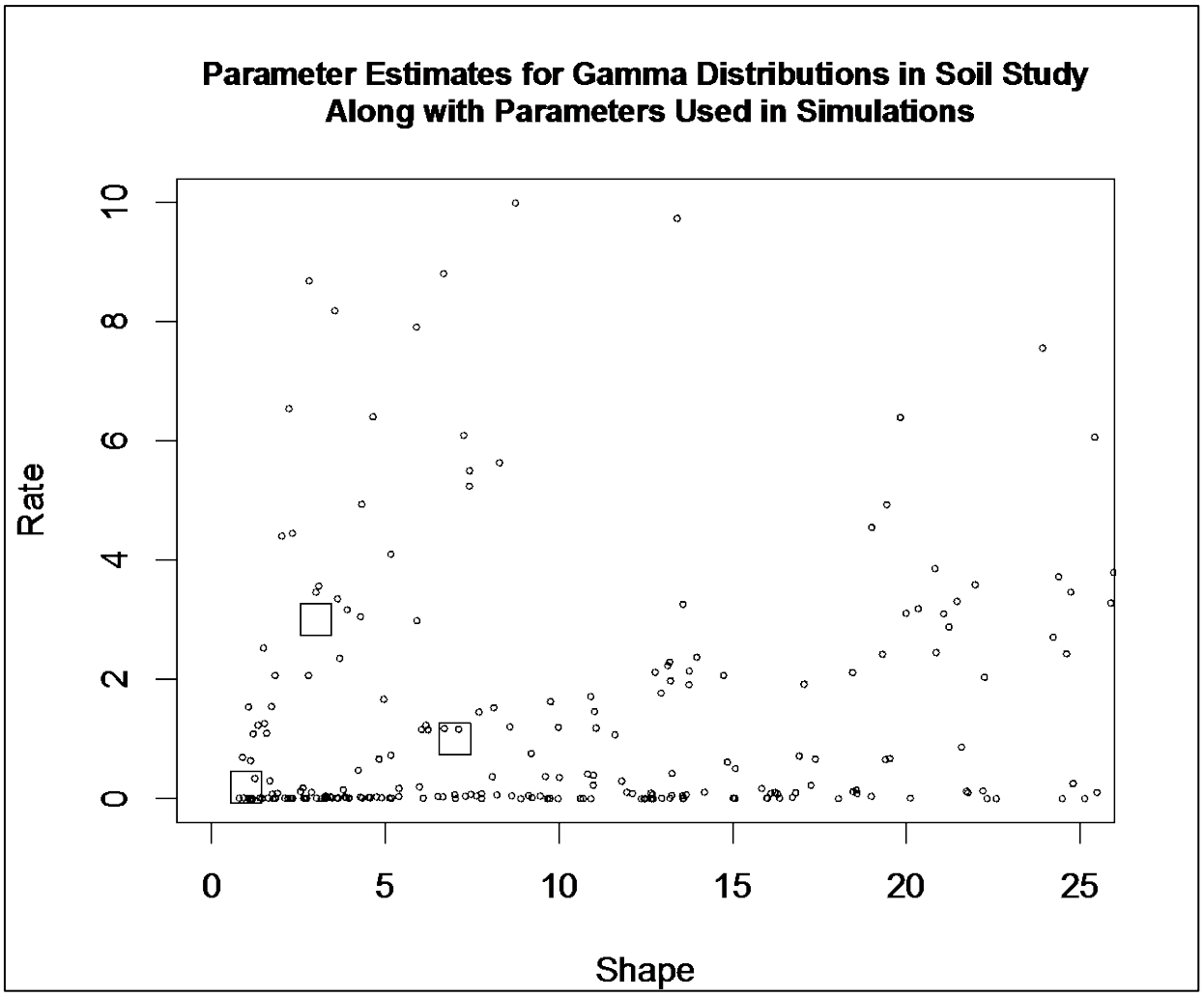


Figure 2. Probability density functions (pdf's) of the gamma distributions used in the simulations.

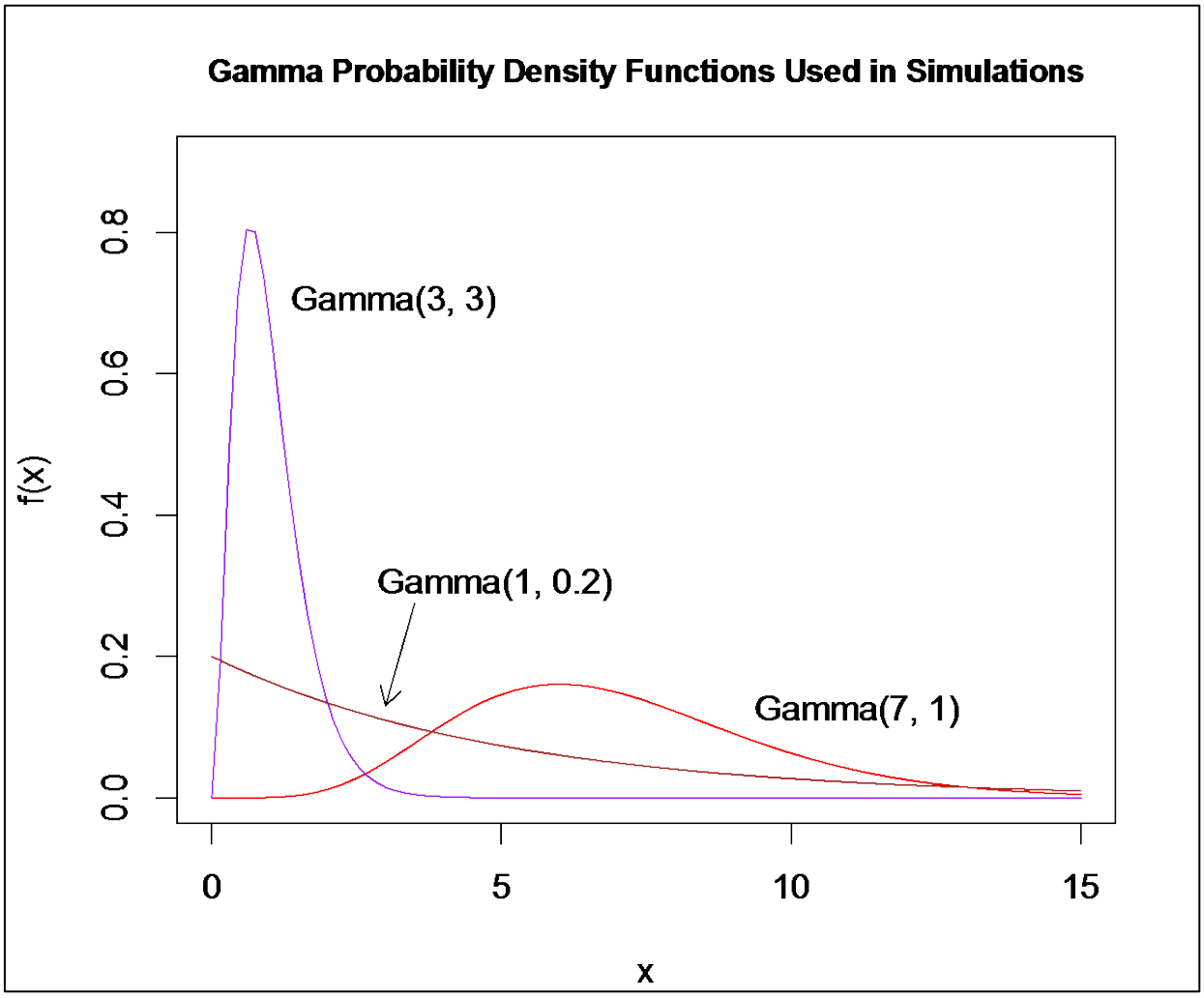


Figure 3a. Plot of the original gamma pdf used in a simulation to randomly generate n=12 observations with the lowest three observations converted to ND's, along with the pdf's for a random sample of eight realizations from the simulation for both the EM method and the GROS method of estimating the gamma parameters (shape and rate).

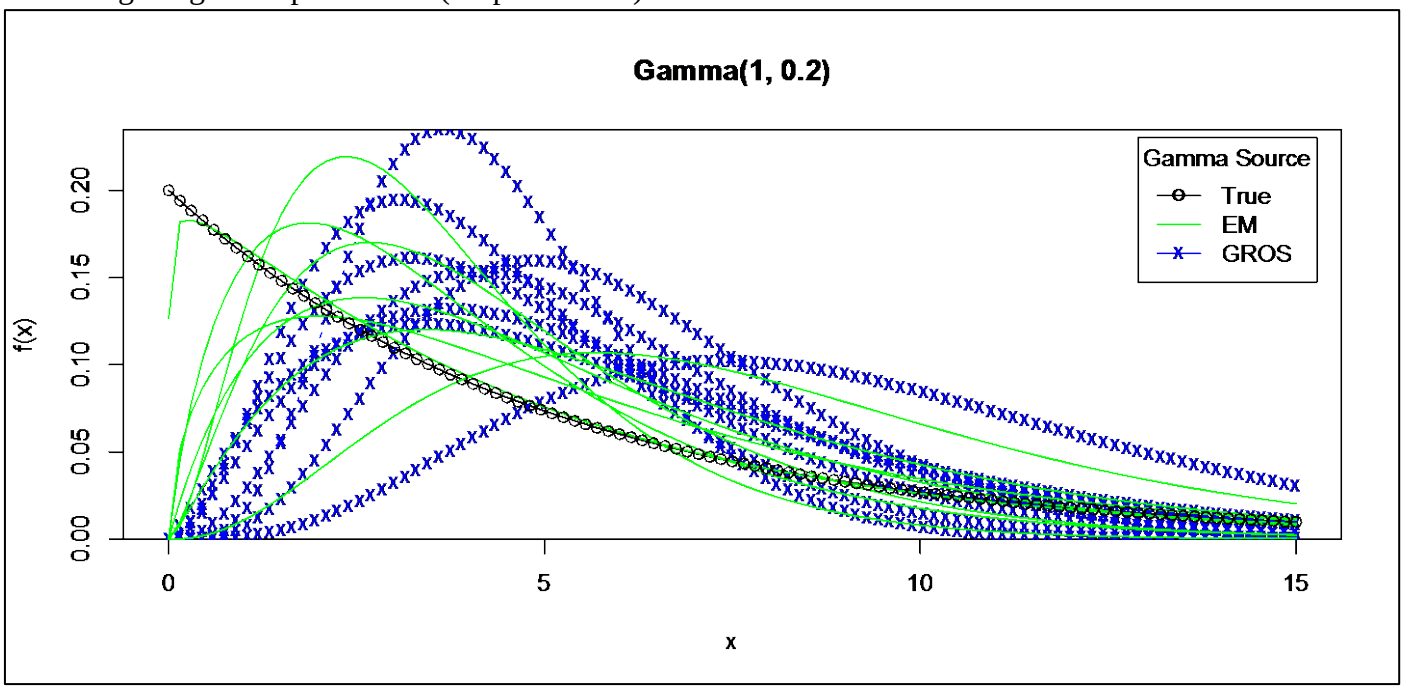


Figure 3b. Plot of the original gamma pdf used in a simulation to randomly generate n=12 observations with the lowest three observations converted to ND's, along with the pdf's for a random sample of eight realizations from the simulation for both the EM method and the GROS method of estimating the gamma parameters (shape and rate).

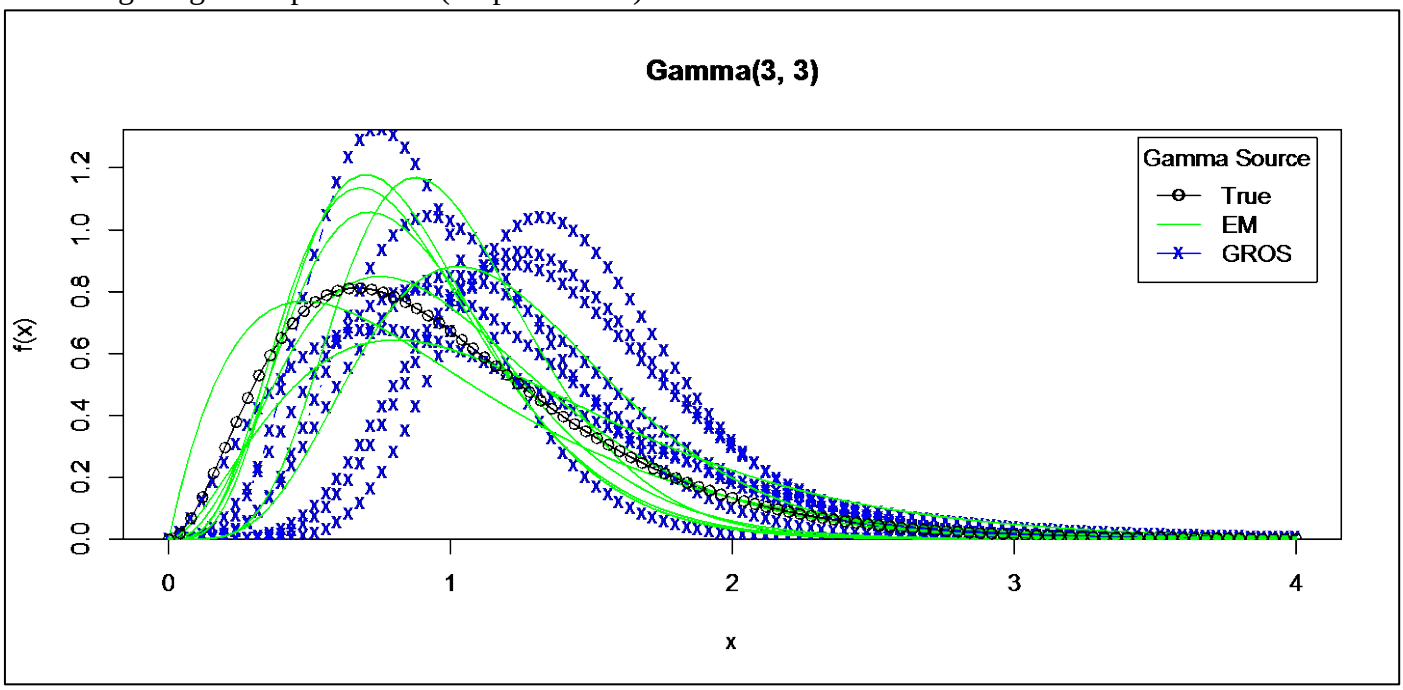


Figure 3c. Plot of the original gamma pdf used in a simulation to randomly generate n=12 observations with the lowest three observations converted to ND's, along with the pdf's for a random sample of eight realizations from the simulation for both the EM method and the GROS method of estimating the gamma parameters (shape and rate).

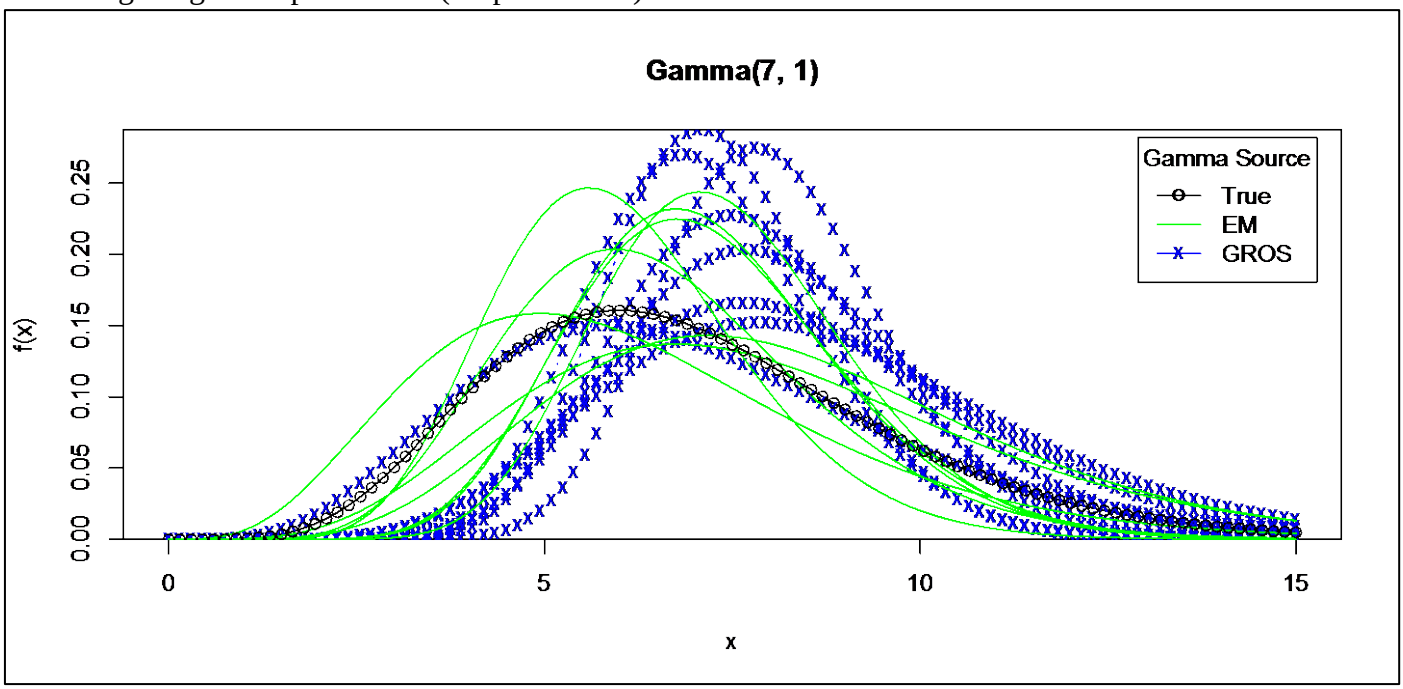


Figure 4a. Histograms from simulations with N=1,000 of the ratio of the calculated 95%-confidence 95%-coverage UTL to the maximum data value in the sample used to calculate the UTL using data randomly generated from each of three distributions.

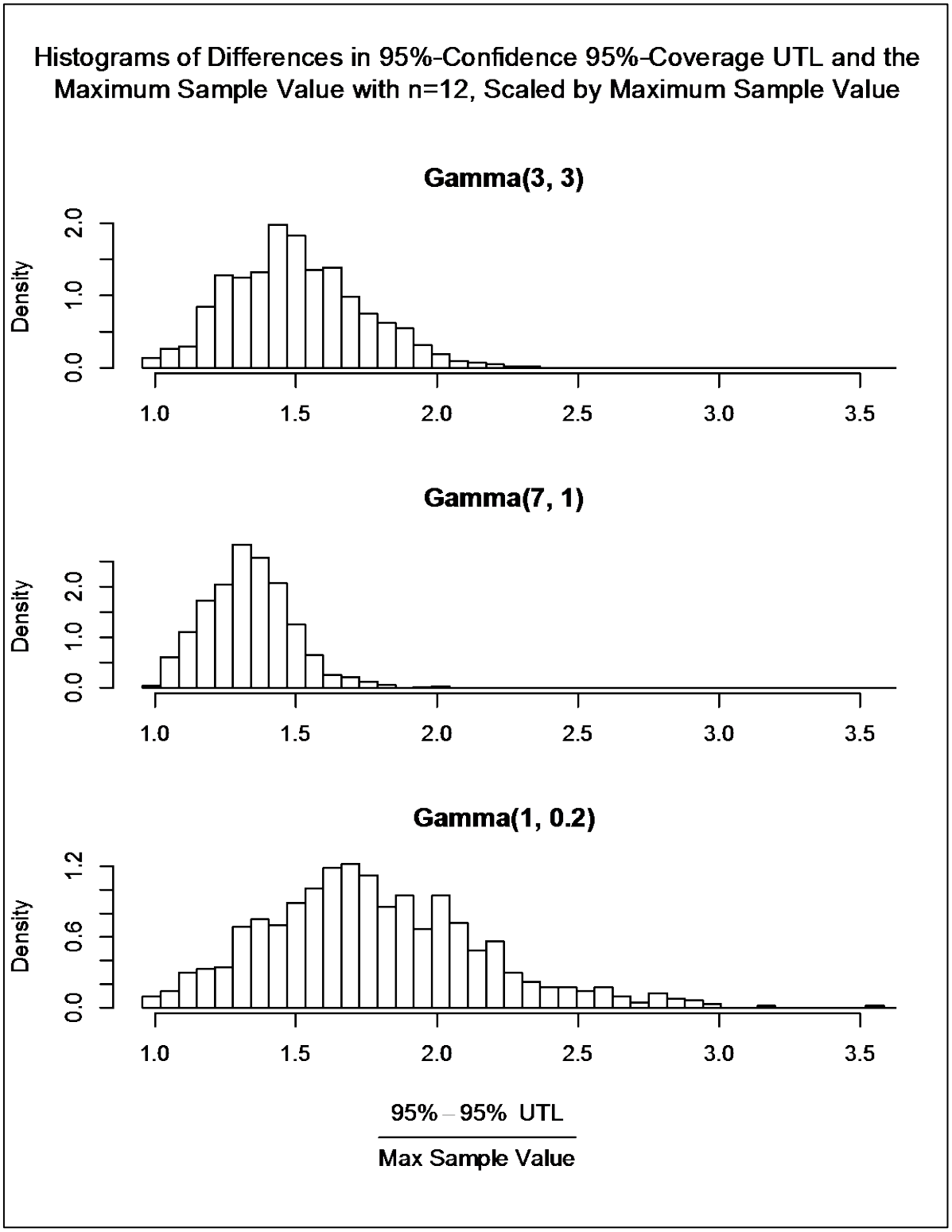


Figure 4b. Histograms from simulations with N=1,000 of the ratio of the calculated 95%-confidence 99%-coverage UTL to the maximum data value in the sample used to calculate the UTL using data randomly generated from each of three distributions.

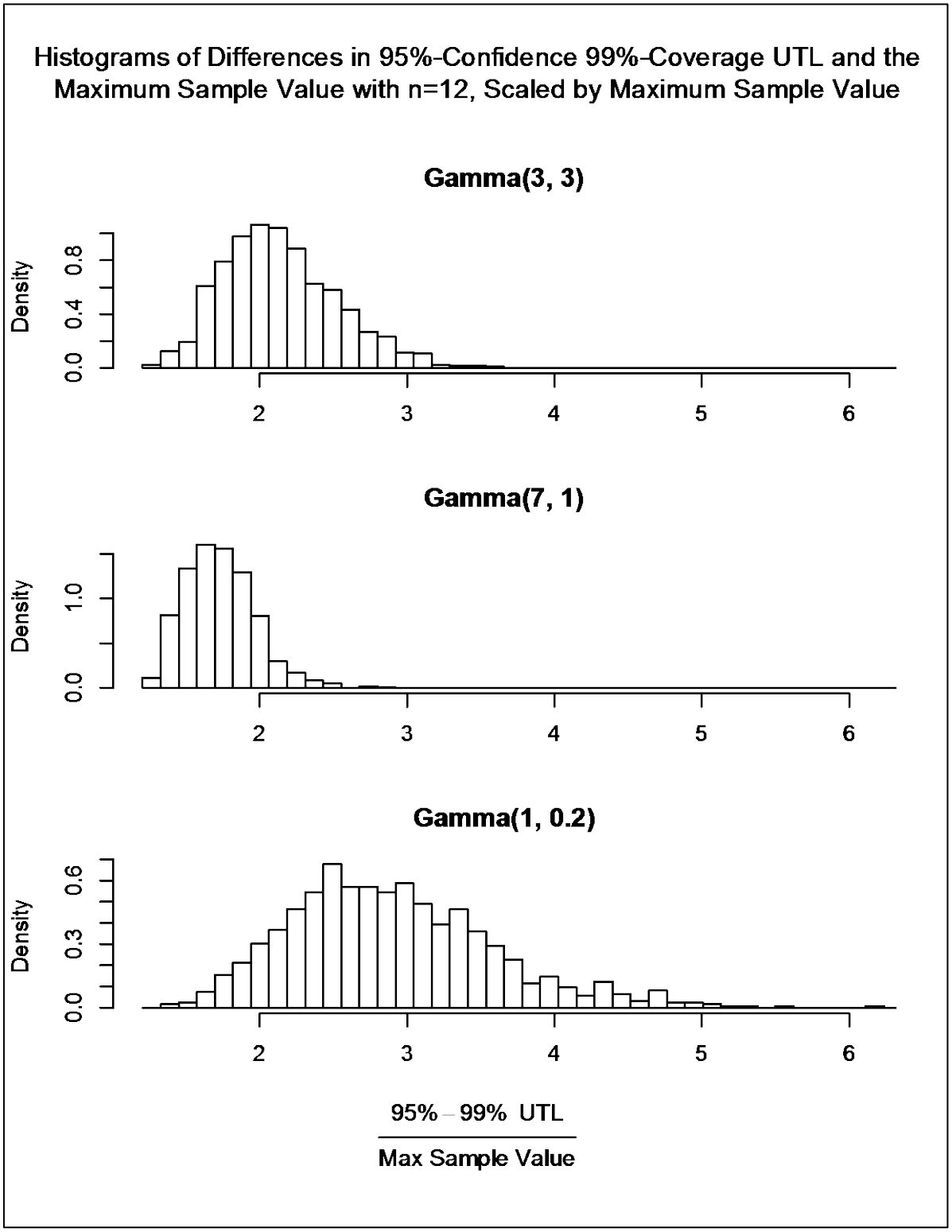


Table 1. Table for simulations with N=1,000 replications of the minimum, maximum, and average difference between the calculated 95%-confidence and both 95%- and 99%-coverage UTLs and the maximum data value from the sample that was used to calculate the UTL using data randomly generated from each of three distributions. Differences are scaled by the maximum sample value.

	95%-Coverage			99%-Coverage		
Distribution	Min	Max	Avg	Min	Max	Avg
Gamma(3, 3)	0.9682	2.3038	1.5012	1.3149	3.5495	2.1386
Gamma(7, 1)	1.0004	2.0186	1.3329	1.2045	2.8511	1.7297
Gamma(1, 0.2)	0.9558	3.5206	1.7748	1.3594	6.1127	2.8985

Table 2. Table of simulation results – using the Dixon outlier test and the PROP outlier test with data generated from one of three different gamma distributions, the second one nearing the shape of a normal distribution. All three gamma distributions used are typical of commonly fit distributions in our current study.

	Gamma(3,3)		Gamma(7, 1)		Gamma(1, 0.2)	
Outlier Test	Samples with Outliers	Average per Sample	Samples with Outliers	Average per Sample	Samples with Outliers	Average per Sample
Dixon outliers	210		157		381	
PROP potential outliers	394	0.442	418	0.484	290	0.309
PROP clear outliers	687	1.654	717	2.119	887	2.698

Figure 5a. Bar charts of simulation results – using the Dixon outlier test and the PROP outlier test with data generated from one of three different gamma distributions, the second one nearing the shape of a normal distribution. All three gamma distributions used are typical of commonly fit distributions in our current study.

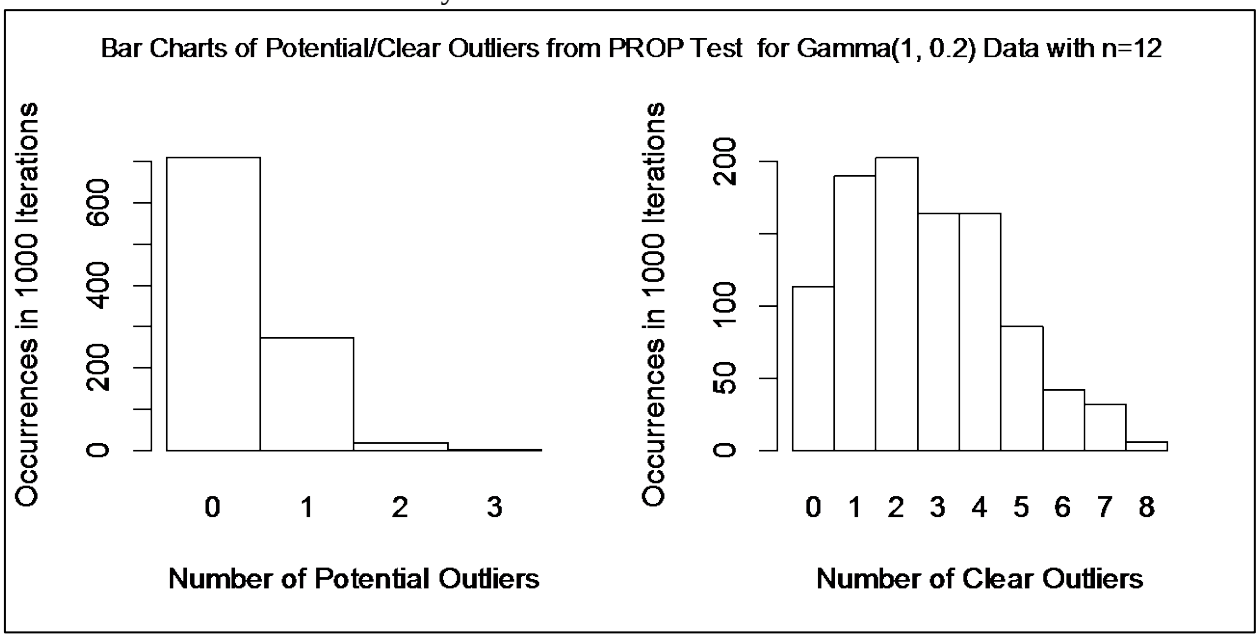


Figure 5b. Bar charts of simulation results – using the Dixon outlier test and the PROP outlier test with data generated from one of three different gamma distributions, the second one nearing the shape of a normal distribution. All three gamma distributions used are typical of commonly fit distributions in our current study.

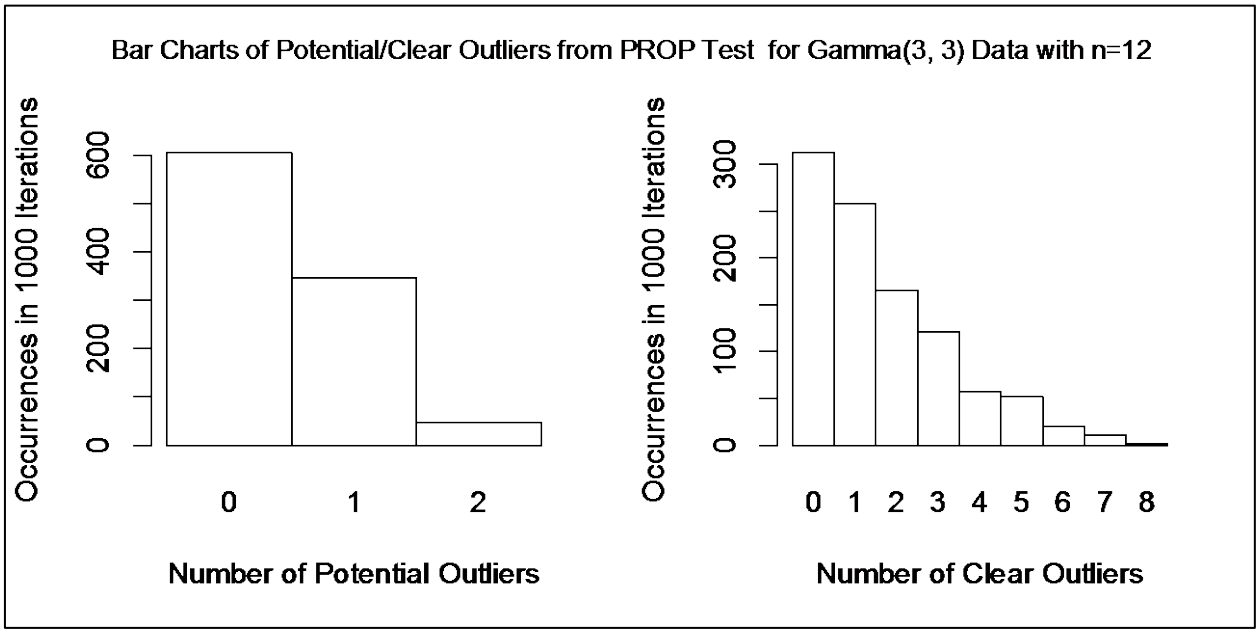


Figure 5c. Bar charts of simulation results – using the Dixon outlier test and the PROP outlier test with data generated from one of three different gamma distributions, the second one nearing the shape of a normal distribution. All three gamma distributions used are typical of commonly fit distributions in our current study.

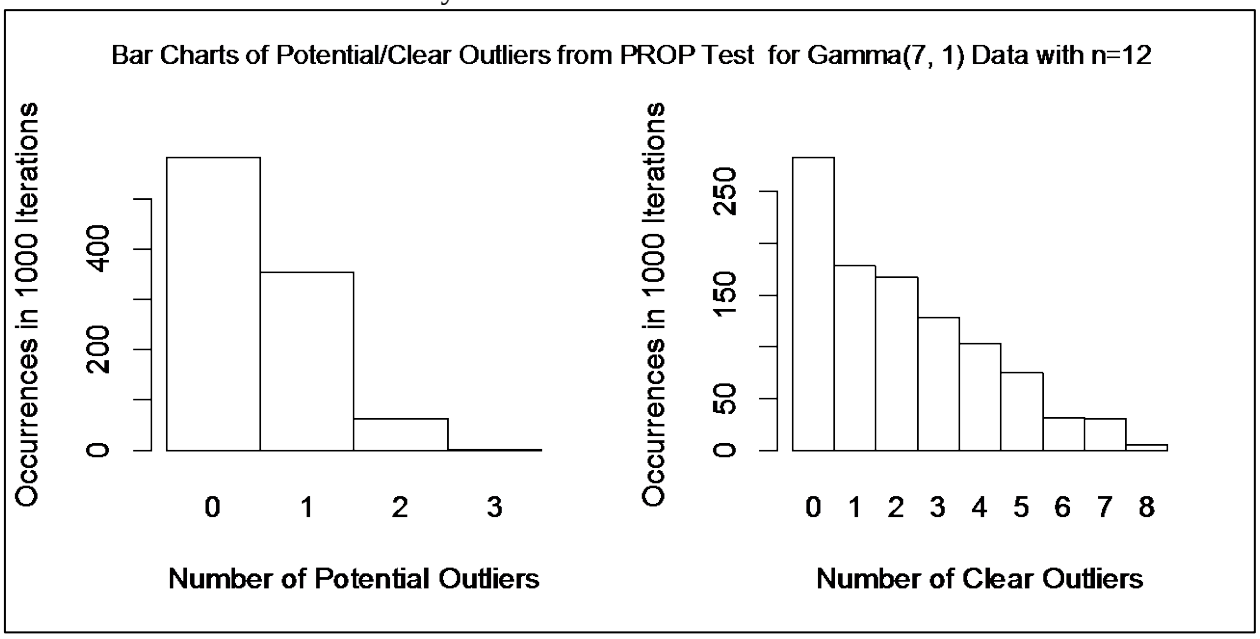


Table 3a. Table of the 139 study sample data sets that, according to the Dixon test with α=0.05, have outliers.

Sample ID	Area	Depth	Analyte	p-value
1	1	shallow	Barium, Total	0.0000
2	1	shallow	Cobalt, Total	0.0379
3	1	shallow	Iron, Total	0.0182
4	1	shallow	Vanadium, Total	0.0431
5	1	middle	Barium, Total	0.0380
6	1	middle	Cyanide, Total	0.0000
7	1	middle	Iron, Total	0.0003
8	1	middle	Molybdenum, Total	0.0347
9	1	middle	Strontium, Total	0.0477
10	1	middle	Titanium, Total	0.0355
11	1	deep	Aluminum, Total	0.0000
12	1	deep	Arsenic, Total	0.0000
13	1	deep	Beryllium, Total	0.0151
14	1	deep	Copper, Total	0.0065
15	1	deep	Iron, Total	0.0000
16	1	deep	Manganese, Total	0.0000
17	1	deep	Molybdenum, Total	0.0112
18	1	deep	Nitrogen	0.0347
19	1	deep	Sodium, Total	0.0495
20	1	deep	Titanium, Total	0.0260
21	1	deep	Vanadium, Total	0.0000
22	2	shallow	Barium, Total	0.0034
23	2	shallow	Chloride	0.0000
24	2	shallow	Chromium, Hexavalent	0.0000
25	2	shallow	Chromium, Total	0.0468
26	2	shallow	Iron, Total	0.0000
27	2	shallow	Mercury, Total	0.0000
28	2	shallow	Molybdenum, Total	0.0012
29	2	shallow	Nickel, Total	0.0462
30	2	shallow	Selenium, Total	0.0000
31	2	shallow	Sodium, Total	0.0000
32	2	shallow	Vanadium, Total	0.0086
33	2	shallow	Zinc, Total	0.0087
34	2	middle	Aluminum, Total	0.0000
35	2	middle	Beryllium, Total	0.0012
36	2	middle	Chloride	0.0055
37	2	middle	Chromium, Total	0.0088
38	2	middle	Iron, Total	0.0000
39	2	middle	Mercury, Total	0.0000
40	2	middle	Nitrogen	0.0060
41	2	middle	Potassium, Total	0.0017
42	2	middle	Selenium, Total	0.0000
43	2	middle	Tin, Total	0.0000
44	2	middle	Titanium, Total	0.0291
45	2	deep	Aluminum, Total	0.0159
46	2	deep	Arsenic, Total	0.0106
47	2	deep	Barium, Total	0.0370

Table 3b. Table of the 139 study sample data sets that, according to the Dixon test with α=0.05, have outliers.

Sample ID	Area	Depth	Analyte	p-value
48	2	deep	Iron, Total	0.0000
49	2	deep	Lead, Total	0.0000
50	2	deep	Magnesium, Total	0.0135
51	2	deep	Manganese, Total	0.0193
52	2	deep	Molybdenum, Total	0.0000
53	2	deep	Nitrogen	0.0040
54	2	deep	Potassium, Total	0.0066
55	2	deep	Selenium, Total	0.0000
56	2	deep	Strontium, Total	0.0253
57	2	deep	Tin, Total	0.0000
58	2	deep	Titanium, Total	0.0000
59	2	deep	Zinc, Total	0.0056
60	3	shallow	Boron, Total	0.0422
61	3	shallow	Cobalt, Total	0.0007
62	3	shallow	Cyanide, Total	0.0000
63	3	shallow	Lead, Total	0.0384
64	3	shallow	Molybdenum, Total	0.0000
65	3	shallow	Nickel, Total	0.0249
66	3	shallow	Perchlorate	0.0297
67	3	shallow	Strontium, Total	0.0063
68	3	shallow	Tin, Total	0.0000
69	3	middle	Arsenic, Total	0.0000
70	3	middle	Beryllium, Total	0.0452
71	3	middle	Cadmium, Total	0.0000
72	3	middle	Cobalt, Total	0.0000
73	3	middle	Copper, Total	0.0163
74	3	middle	Cyanide, Total	0.0000
75	3	middle	Iron, Total	0.0000
76	3	middle	Manganese, Total	0.0000
77	3	middle	Mercury, Total	0.0000
78	3	middle	Nickel, Total	0.0263
79	3	middle	Sodium, Total	0.0035
80	3	middle	Tin, Total	0.0000
81	3	middle	Uranium, Total	0.0090
82	3	middle	Zinc, Total	0.0000
83	3	deep	Cadmium, Total	0.0000
84	3	deep	Chloride	0.0113
85	3	deep	Cyanide, Total	0.0000
86	3	deep	Nickel, Total	0.0150
87	3	deep	Strontium, Total	0.0000
88	3	deep	Tin, Total	0.0000
89	3	deep	Vanadium, Total	0.0118
90	4	shallow	Antimony, Total	0.0000
91	4	shallow	Arsenic, Total	0.0232
92	4	shallow	Barium, Total	0.0157
93	4	shallow	Beryllium, Total	0.0080
94	4	shallow	Chloride	0.0000

Table 3c. Table of the 139 study sample data sets that, according to the Dixon test with α=0.05, have outliers.

Sample ID	Area	Depth	Analyte	p-value
95	4	shallow	Chromium, Total	0.0482
96	4	shallow	Magnesium, Total	0.0000
97	4	shallow	Perchlorate	0.0000
98	4	shallow	Potassium, Total	0.0171
99	4	shallow	Sodium, Total	0.0000
100	4	shallow	Strontium, Total	0.0419
101	4	shallow	Titanium, Total	0.0214
102	4	shallow	Uranium, Total	0.0105
103	4	middle	Antimony, Total	0.0000
104	4	middle	Barium, Total	0.0000
105	4	middle	Cadmium, Total	0.0000
106	4	middle	Chromium, Hexavalent	0.0000
107	4	middle	Copper, Total	0.0164
108	4	middle	Cyanide, Total	0.0000
109	4	middle	Mercury, Total	0.0473
110	4	middle	Nickel, Total	0.0098
111	4	middle	Uranium, Total	0.0000
112	4	deep	Barium, Total	0.0000
113	4	deep	Cadmium, Total	0.0000
114	4	deep	Calcium, Total	0.0060
115	4	deep	Copper, Total	0.0218
116	4	deep	Iron, Total	0.0078
117	4	deep	Manganese, Total	0.0447
118	4	deep	Strontium, Total	0.0281
119	5	shallow	Aluminum, Total	0.0170
120	5	shallow	Cadmium, Total	0.0000
121	5	shallow	Chloride	0.0066
122	5	shallow	Chromium, Hexavalent	0.0000
123	5	shallow	Manganese, Total	0.0000
124	5	shallow	Nitrogen	0.0000
125	5	shallow	Sodium, Total	0.0106
126	5	middle	Barium, Total	0.0313
127	5	middle	Cadmium, Total	0.0000
128	5	middle	Calcium, Total	0.0159
129	5	middle	Cyanide, Total	0.0000
130	5	middle	Molybdenum, Total	0.0139
131	5	middle	Tin, Total	0.0000
132	5	deep	Beryllium, Total	0.0272
133	5	deep	Cadmium, Total	0.0000
134	5	deep	Chromium, Hexavalent	0.0000
135	5	deep	Magnesium, Total	0.0441
136	5	deep	Manganese, Total	0.0171
137	5	deep	Nitrogen	0.0052
138	5	deep	Sodium, Total	0.0373
139	5	deep	Strontium, Total	0.0278

Figure 6a. Strip chart of analyte concentration values (scaled to have the same maximum) by analyte Sample ID number given in Table 2.

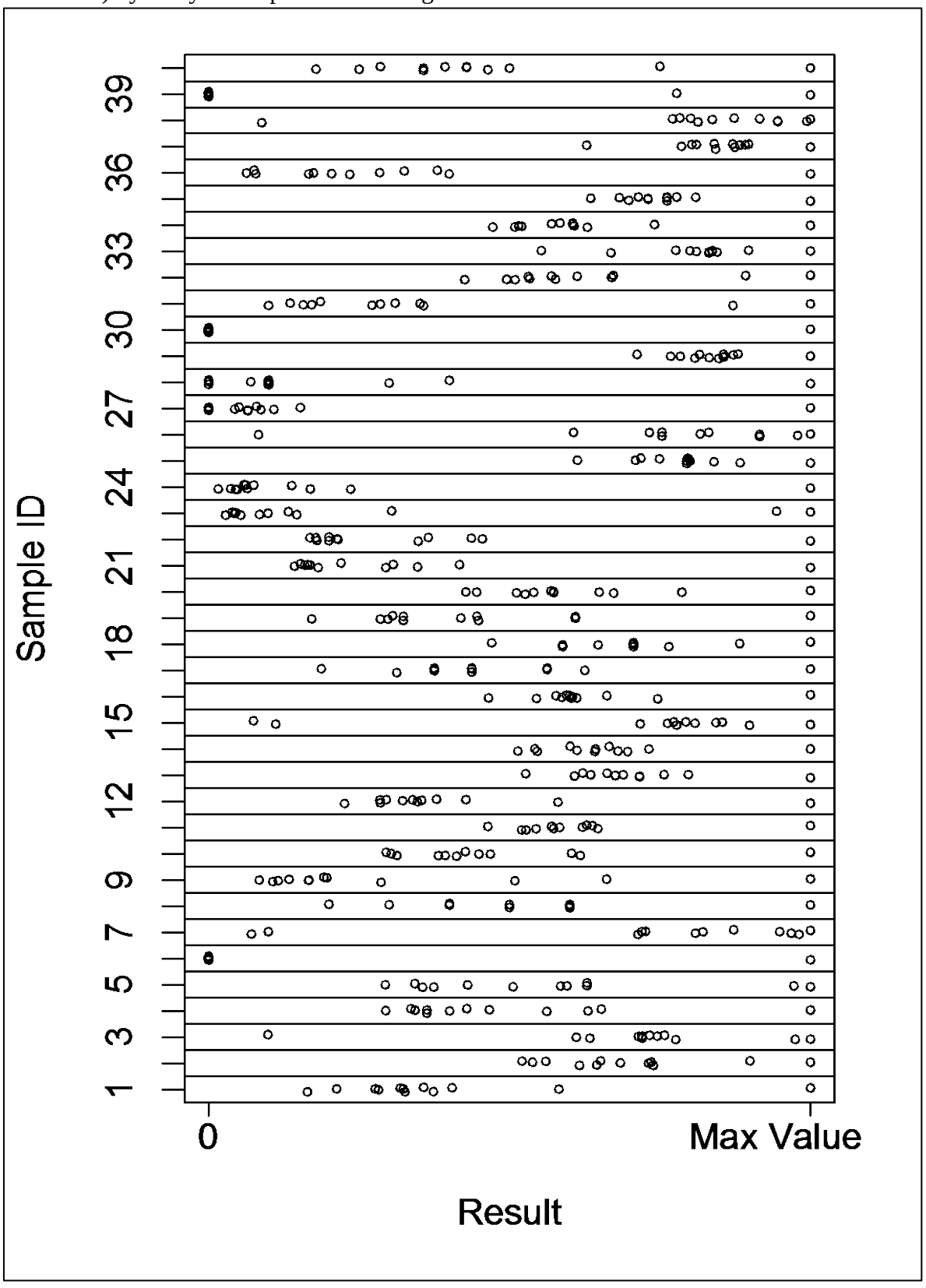


Figure 6b. Strip chart of analyte concentration values (scaled to have the same maximum) by analyte Sample ID number given in Table 2.

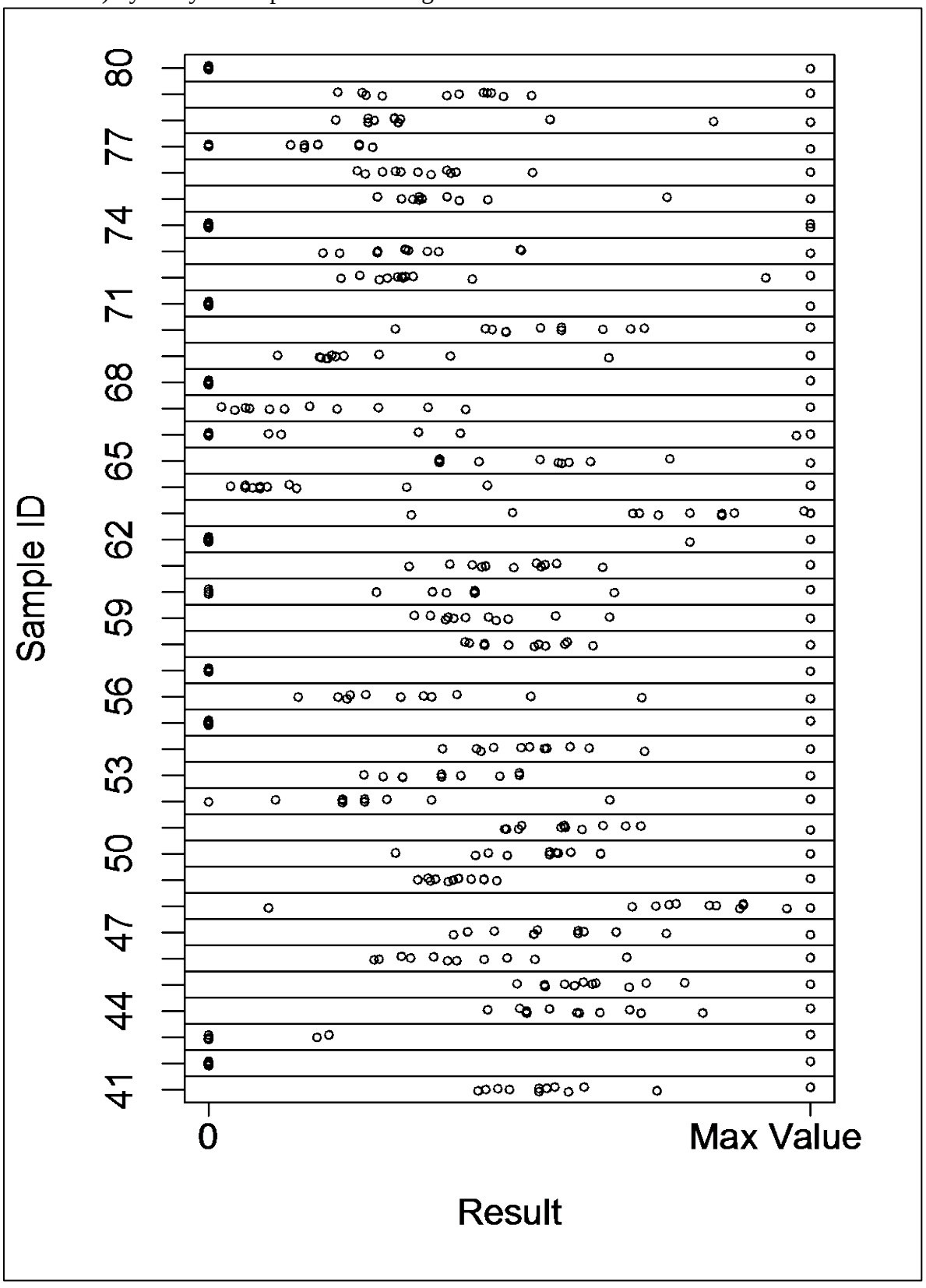


Figure 6c. Strip chart of analyte concentration values (scaled to have the same maximum) by analyte Sample ID number given in Table 2.

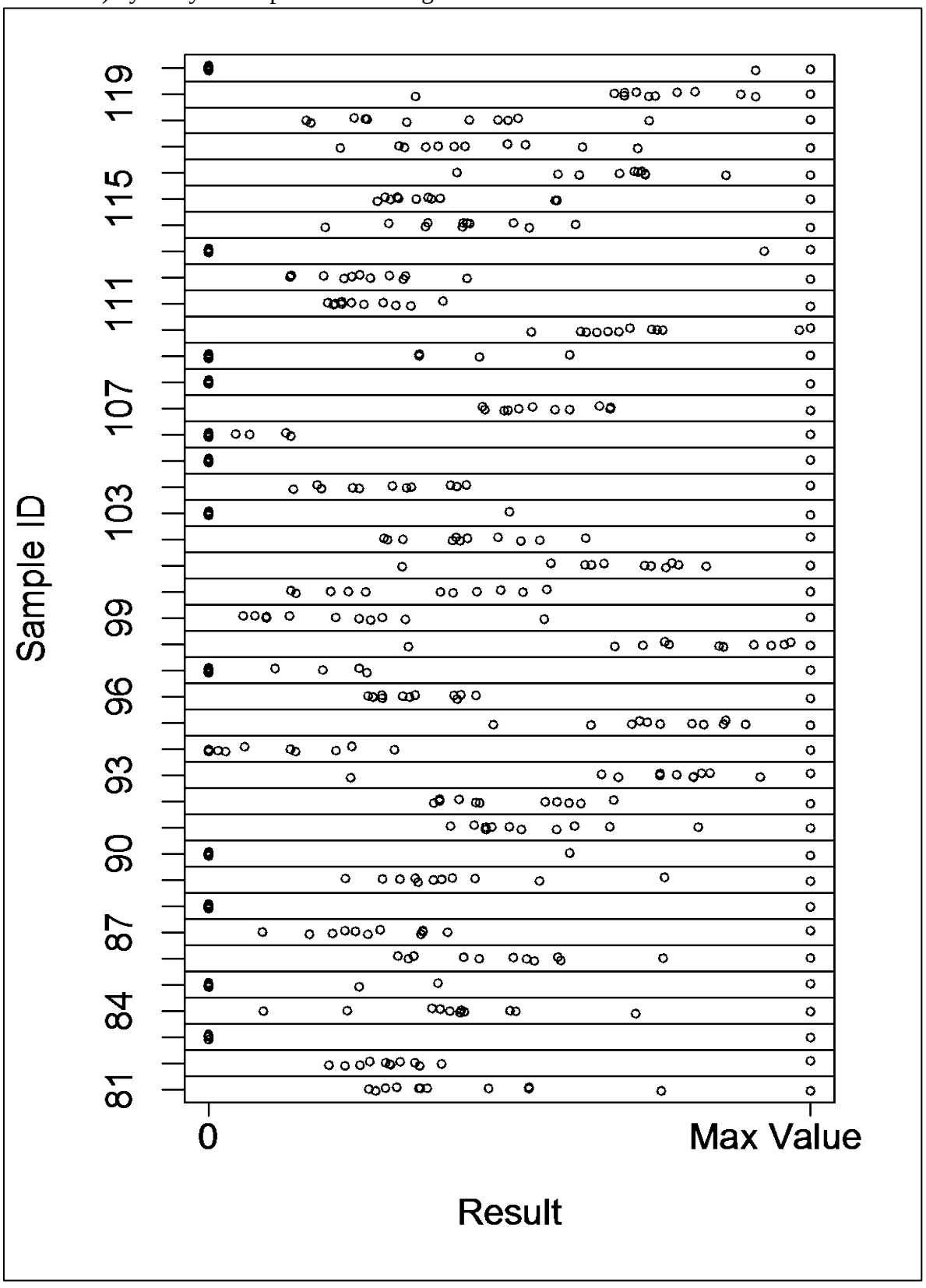


Figure 6d. Strip chart of analyte concentration values (scaled to have the same maximum) by analyte Sample ID number given in Table 2.

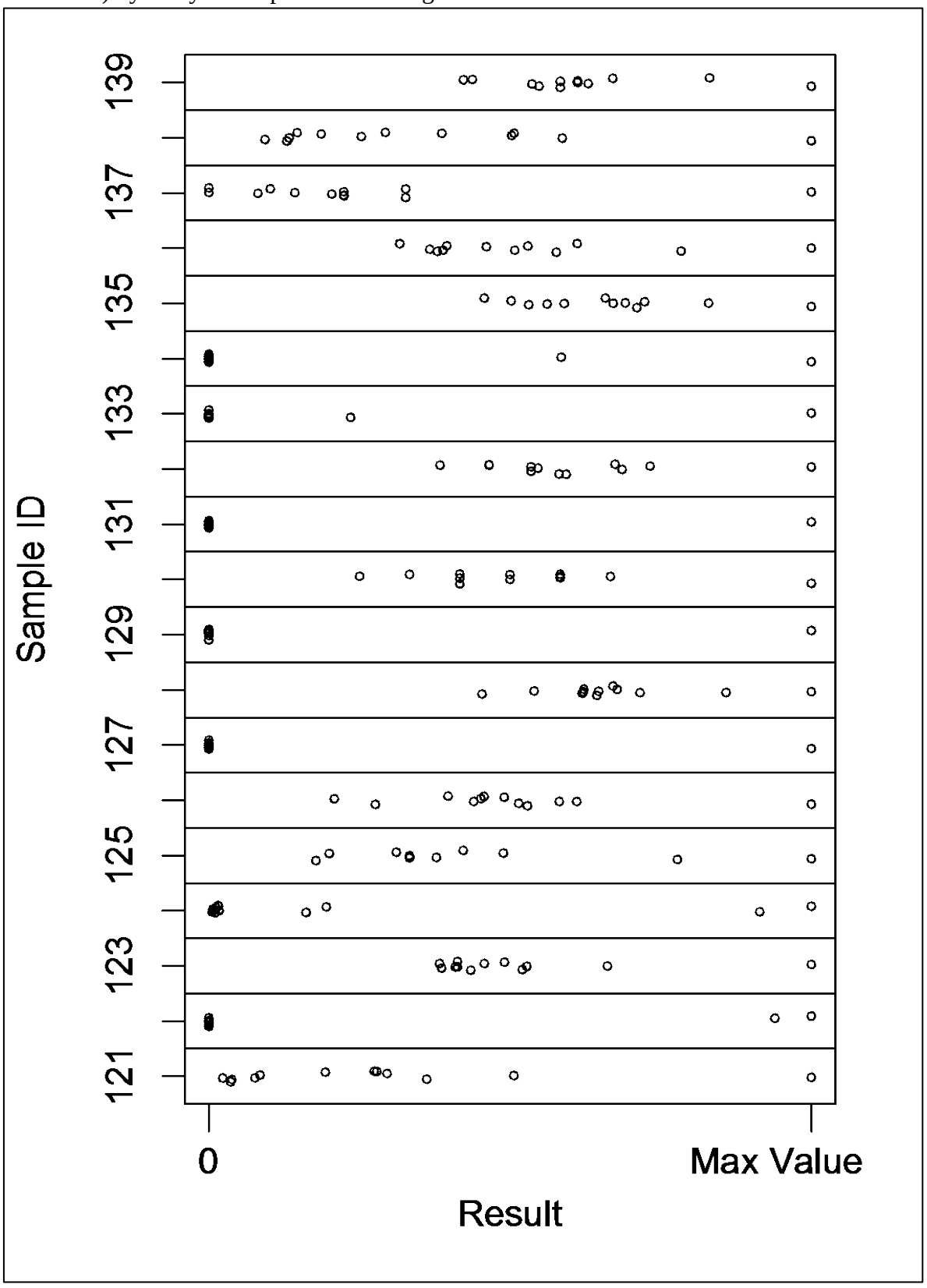
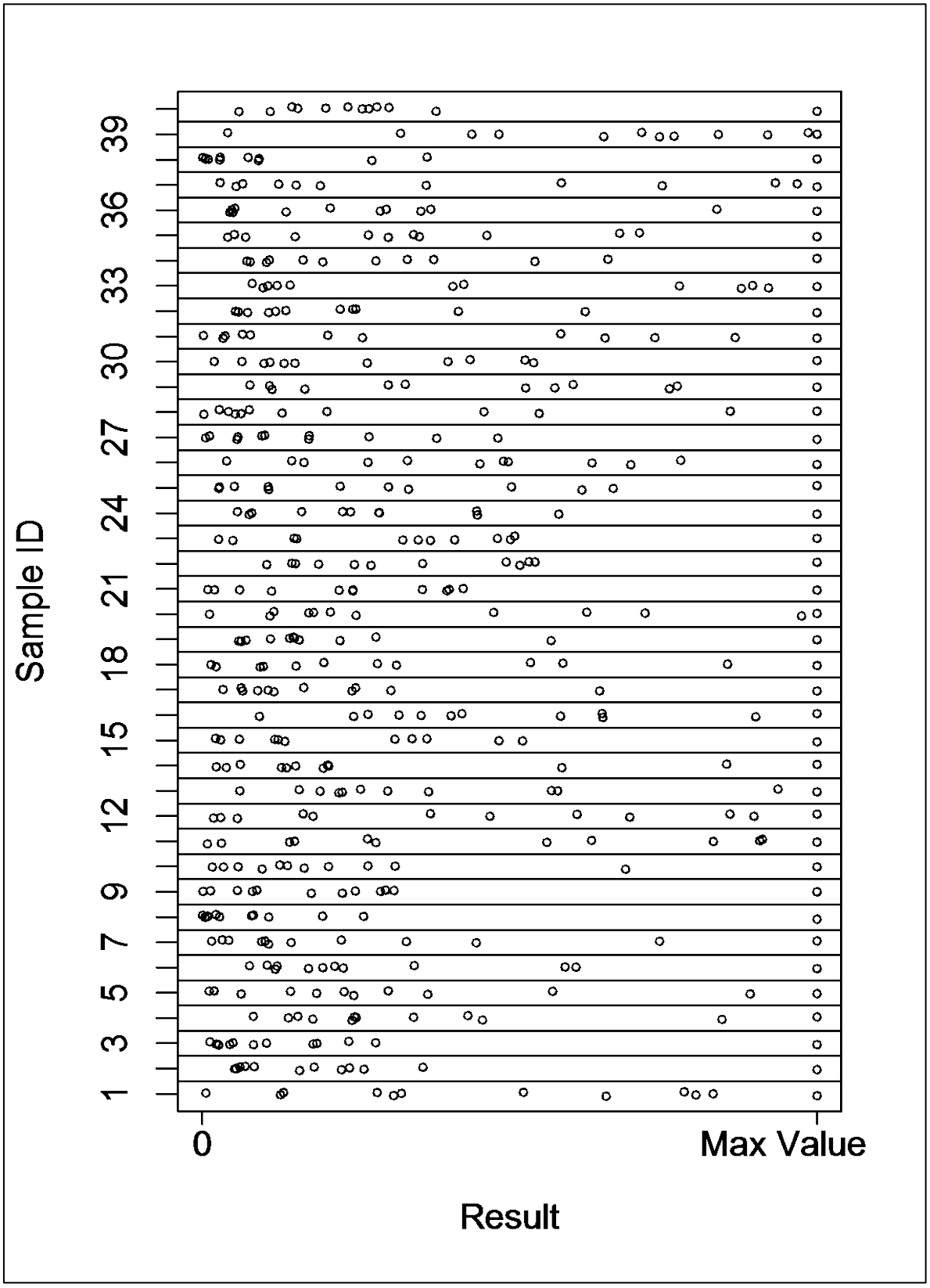


Figure 7. Strip chart of samples of size n=12 generated from the Gamma(1, 0.2) distribution (scaled to have the same maximum) for comparison with the strip charts for the study data.



References

Daniel, David (March 24, 2014), *Statistical Development of Soil Background Concentrations*, report prepared for Pamela Egan, Navarro Research and Engineering, Inc.

Hahn, Gerald J and Meeker, William Q. (1991), *Statistical Intervals – A Guide for Practitioners*, John Wiley & Sons, Inc., New York.

Helsel, Dennis R. (2012), *Statistics for Censored Environmental Data Using Minitab and R*, Second Edition, John Wiley & Sons, Hoboken, New Jersey.

Kutner, M. H., Nachtsheim, C. J., and Neter, J. (2004), *Applied Linear Regression Models*, Fourth Edition, McGraw-Hill/Irwin.

NASA White Sands Test Facility Soil Background Study Investigation Work Plan, May 2012, Revised September 2012.

New Mexico Environmental Department (NMED, November 2009), *National Aeronautics and Space Administration White Sands Test Facility (NASA WSTF) Permit Number NM8800019434*.

Ofungwu, Joseph (2014), *Statistical Applications for Environmental Analysis and Risk Assessment*, John Wiley & Sons, Hoboken, New Jersey.

R Core Team (2014), *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/

Singh, Anita; Armbya1, Narain; and Singh, Ashok K. (May 2010), *ProUCL Version 4.1.00 Technical Guide (Draft) – Statistical Software for Environmental Applications for Data Sets with and without Nondetect Observations*.

Singh, Anita and Singh, Ashok K. (September 2013), *ProUCL Version 5.0.00 Technical Guide (Draft) – Statistical Software for Environmental Applications for Data Sets with and without Nondetect Observations*.

U.S. Environmental Protection Agency (March 2009), *Statistical Analysis of Groundwater Monitoring Data at RCRA Facilities – Unified Guidance*, EPA 530/R-09-007, Office of Resource Conservation and Recovery.

Appendix A

R code for estimating gamma distribution parameters when ND's are present – a modification of method *fitdistr() in the MASS package.*

```
# This function is a modification of the fitdistr() method.  It attempts
# to solve for the MLE parameters of the specified distribution, densfun,
# given the sample data, x.  However, any values of x equal to zero are
# assumed to be non-detects and it initially substitutes half of the
# detection limit values for the non-detects values.  Thereafter, after
# every iteration of the solving process, the current values of the
# parameter estimates are used to get expected values for the specified
# distribution and the non-detects are replaced by the corresponding
# expected value estimates.

fitdistr.nds <- function (x, detect.limits, densfun, start, ...)
{
    n.total <- length(x)
    ord <- order(x)
    x <- x[ord]
    detect.limits <- detect.limits[ord]
    dl2 <- 0.5 * detect.limits
    wh.nds <- which(x == 0)
    wh.detects <- which(x != 0)
    # x.detects <- sort(x[x != 0])
    n.detects <- length(wh.detects)
    n.nds <- n.total - n.detects
    k.plus.1 <- n.total - n.detects + 1
    p <- (1:n.total - 0.5) / n.total
    p.detects <- p[wh.detects]
    p.nds <- p[wh.nds]
    x[wh.nds] <- dl2[wh.nds]


    myfn <- function(parm, ...) -sum(log(dens(parm, ...)))
    mylogfn <- function(parm, ...) -sum(dens(parm, ..., log = TRUE))
    mydt <- function(x, m, s, df, log) dt((x - m)/s, df, log = TRUE) -
        log(s)
    Call <- match.call(expand.dots = TRUE)
    Call$detect.limits <- NULL
    if (missing(start))
        start <- NULL
    dots <- names(list(...))
    dots <- dots[!is.element(dots, c("upper", "lower"))]
    if (missing(x) || length(x) == 0L || mode(x) != "numeric")
        stop("'x' must be a non-empty numeric vector")
    if (any(!is.finite(x)))
        stop("'x' contains missing or infinite values")
    if (missing(densfun) || !(is.function(densfun) || is.character(densfun)))
        stop("'densfun' must be supplied as a function or name")
    control <- list()
    n <- length(x)
    if (is.character(densfun)) {
        distname <- tolower(densfun)
        densfun <- switch(distname, beta = dbeta, cauchy = dcauchy,
            `chi-squared` = dchisq, exponential = dexp, f = df,
            gamma = dgamma, geometric = dgeom, `log-normal` = dlnorm,
            lognormal = dlnorm, logistic = dlogis, `negative binomial` = dnbinom,
            normal = dnorm, poisson = dpois, t = mydt, weibull = dweibull,
            NULL)
        quantfun <- switch(distname, beta = qbeta, cauchy = qcauchy,
            `chi-squared` = qchisq, exponential = qexp, f = qf,
```

```
    gamma = qgamma, geometric = qgeom, `log-normal` = qlnorm,
    lognormal = qlnorm, logistic = qlogis, `negative binomial` = qnbinom,
    normal = qnorm, poisson = qpois, t = qt, weibull = qweibull,
    NULL)
if (is.null(densfun))
    stop("unsupported distribution")
if (distname %in% c("lognormal", "log-normal")) {
    if (!is.null(start))
        stop(gettextf("supplying pars for the %s distribution is not supported",
          "log-Normal"), domain = NA)
    if (any(x <= 0))
        stop("need positive values to fit a log-Normal")
    lx <- log(x)
    sd0 <- sqrt((n - 1)/n) * sd(lx)
    mx <- mean(lx)
    estimate <- c(mx, sd0)
    sds <- c(sd0/sqrt(n), sd0/sqrt(2 * n))
    names(estimate) <- names(sds) <- c("meanlog", "sdlog")
    vc <- matrix(c(sds[1]^2, 0, 0, sds[2]^2), ncol = 2,
        dimnames = list(names(sds), names(sds)))
    names(estimate) <- names(sds) <- c("meanlog", "sdlog")
    return(structure(list(estimate = estimate, sd = sds,
        vcov = vc, n = n, convergence = -1, loglik = sum(dlnorm(x, mx,
          sd0, log = TRUE))), class = "fitdistr.nds"))
}
if (distname == "normal") {
    if (!is.null(start))
        stop(gettextf("supplying pars for the %s distribution is not supported",
          "Normal"), domain = NA)
    sd0 <- sqrt((n - 1)/n) * sd(x)
    mx <- mean(x)
    estimate <- c(mx, sd0)
    sds <- c(sd0/sqrt(n), sd0/sqrt(2 * n))
    names(estimate) <- names(sds) <- c("mean", "sd")
    vc <- matrix(c(sds[1]^2, 0, 0, sds[2]^2), ncol = 2,
        dimnames = list(names(sds), names(sds)))
    return(structure(list(estimate = estimate, sd = sds,
        vcov = vc, n = n, convergence = -1, loglik = sum(dnorm(x, mx, sd0,
          log = TRUE))), class = "fitdistr.nds"))
}
if (distname == "poisson") {
    if (!is.null(start))
        stop(gettextf("supplying pars for the %s distribution is not supported",
          "Poisson"), domain = NA)
    estimate <- mean(x)
    sds <- sqrt(estimate/n)
    names(estimate) <- names(sds) <- "lambda"
    vc <- matrix(sds^2, ncol = 1, nrow = 1, dimnames = list("lambda",
        "lambda"))
    return(structure(list(estimate = estimate, sd = sds,
        vcov = vc, n = n, convergence = -1, loglik = sum(dpois(x, estimate,
          log = TRUE))), class = "fitdistr"))
}
if (distname == "exponential") {
    if (any(x < 0))
        stop("Exponential values must be >= 0")
    if (!is.null(start))
        stop(gettextf("supplying pars for the %s distribution is not supported",
          "exponential"), domain = NA)
    estimate <- 1/mean(x)
    sds <- estimate/sqrt(n)
    vc <- matrix(sds^2, ncol = 1, nrow = 1, dimnames = list("rate",
        "rate"))
    names(estimate) <- names(sds) <- "rate"
    return(structure(list(estimate = estimate, sd = sds,
```

```
            vcov = vc, n = n, convergence = -1, loglik = sum(dexp(x, estimate,
              log = TRUE))), class = "fitdistr"))
    }
    if (distname == "geometric") {
        if (!is.null(start))
            stop(gettextf("supplying pars for the %s distribution is not supported",
              "geometric"), domain = NA)
        estimate <- 1/(1 + mean(x))
        sds <- estimate * sqrt((1 - estimate)/n)
        vc <- matrix(sds^2, ncol = 1, nrow = 1, dimnames = list("prob",
            "prob"))
        names(estimate) <- names(sds) <- "prob"
        return(structure(list(estimate = estimate, sd = sds,
            vcov = vc, n = n, convergence = -1, loglik = sum(dexp(x, estimate,
              log = TRUE))), class = "fitdistr"))
    }
    if (distname == "weibull" && is.null(start)) {
        if (any(x <= 0))
            stop("Weibull values must be > 0")
        lx <- log(x)
        m <- mean(lx)
        v <- var(lx)
        shape <- 1.2/sqrt(v)
        scale <- exp(m + 0.572/shape)
        start <- list(shape = shape, scale = scale)
        start <- start[!is.element(names(start), dots)]
    }
    if (distname == "gamma" && is.null(start)) {
        if (any(x < 0))
            stop("gamma values must be >= 0")
        m <- mean(x)
        v <- var(x)
        start <- list(shape = m^2/v, rate = m/v)
        start <- start[!is.element(names(start), dots)]
        control <- list(parscale = c(1, start$rate))
    }
    if (distname == "negative binomial" && is.null(start)) {
        m <- mean(x)
        v <- var(x)
        size <- if (v > m)
            m^2/(v - m)
        else 100
        start <- list(size = size, mu = m)
        start <- start[!is.element(names(start), dots)]
    }
    if (is.element(distname, c("cauchy", "logistic")) &&
        is.null(start)) {
        start <- list(location = median(x), scale = IQR(x)/2)
        start <- start[!is.element(names(start), dots)]
    }
    if (distname == "t" && is.null(start)) {
        start <- list(m = median(x), s = IQR(x)/2, df = 10)
        start <- start[!is.element(names(start), dots)]
    }
}
if (is.null(start) || !is.list(start))
    stop("'start' must be a named list")
nm <- names(start)
f <- formals(densfun)
args <- names(f)
m <- match(nm, args)
if (any(is.na(m)))
    stop("'start' specifies names which are not arguments to 'densfun'")
formals(densfun) <- c(f[c(1, m)], f[-c(1, m)])
```

```
    dens <- function(parm, x, ...) {

            # x[wh.nds] <- quantfun(p.nds, parm[1], parm[2])
            # densfun(x, parm[1], parm[2], ...)

            x[wh.nds] <- quantfun(p.nds, parm, ...)
            densfun(x, parm, ...)
        }

    if ((l <- length(nm)) > 1L)
        body(dens) <- parse( text = paste("{",
            paste("x[wh.nds] <- quantfun(p.nds,", paste("parm[",
                    1L:l, "]", collapse = ", "), ")"),
            "\n",
            paste("densfun(x,", paste("parm[",
                    1L:l, "]", collapse = ", "), ", ...)"), "}"))

    Call[[1L]] <- quote(stats::optim)
    Call$densfun <- Call$start <- NULL
    Call$x <- x
    Call$par <- start
    Call$fn <- if ("log" %in% args)
        mylogfn
    else myfn
    #     if ("log" %in% args) {
    #         Call$fn <- mylogfn
    #   print("log")
    #     } else {
    #   Call$fn <- myfn
    #   print("myfn")
    # }
    Call$hessian <- TRUE
    if (length(control))
        Call$control <- control
    if (is.null(Call$method)) {
        if (any(c("lower", "upper") %in% names(Call)))
            Call$method <- "L-BFGS-B"
        else if (length(start) > 1L)
            Call$method <- "BFGS"
        else Call$method <- "Nelder-Mead"
    }
    res <- eval.parent(Call)
    if (res$convergence > 0L)
        # stop("optimization failed")
        cat("optimization failed")
    vc <- solve(res$hessian)
    sds <- sqrt(diag(vc))
    return( structure(list(estimate = res$par, sd = sds, vcov = vc, loglik = -res$value,
        n = n, convergence = res$convergence), class = "fitdistr.nds") )
}
```

Appendix B

R code for calculating nonparametric approximate UTL's.

```
# ----------------
# This function performs a bootstrap on the given sample data, x, and
# then creates a function that will give probabilities from the KM ECDF.
# The KM ECDF is then used to obtain the empirical CDF percentile
# specified by the argument, percentile.  It iterates this process the
# number of times specified by num.samples, and returns that many
# bootstrapped percentile estiamtes, sorted in ascending order.  A 95%
# upper tolerance limit would be given by z[0.95*num.samples], where
# z is the vector of ordered bootstrap percentiles returned from this
# function.
#
# This technique may be impeded by the fact that ECDF is a stepwise function
# and can only produce percentiles that are in the original data set.  This
# could likely be improved by using a kernel density estimate calcualted from
# the bootstrap sample instead, and similarly determining the specified
# percentile, iterating num.sample times and obtaining a UTL in the same
# manner.

bootstrap.km.percent <- function(x, detect.limits, num.samples, percentile) {

   n.total <- length(x)
   indices <- 1:n.total

   # Bootstrap the specified percentile the specified number of times.
   perc.vec <- NULL
   i <- 0
   while(i < num.samples) {

      # Get a bootstrap sample and obtain the KM ECDF for it.
      bs.indices <- sample(indices, n.total, replace=TRUE)
      y <- x[bs.indices]
      # Ensure at least 5 detects
      if( sum(y != 0) < 5 ) next
      km.cdf <- km.distr(y, detect.limits[bs.indices])

      # Get the specified percentile from the ECDF.
      z <- sort(unique(y))

      # tryCatch({
         p <- km.cdf(z)
      # }, error = function(error.condition) {
      #   ERRORS <<- append(ERRORS, paste0("area=", cur.area, ", depth=", cur.depth,
      #                                    ", analyte=", a, ", distr=expon"))
      #   message(paste("area=", cur.area, ", depth=", cur.depth,
      #                 ", analyte=", a, "\n", sep=""))
      #   print(bs.indices)
      #   message(paste("i=", i, "\n", sep=""))
      #   message(paste("z=", z, "\n", "y=", y, "\n", sep=""))
      #   stop()
      #   NULL
      # })

      perc.vec <- c(perc.vec, z[which(p >= percentile)[1]])
      i <- i + 1
   }

   # Return all of the bootstrapped percentiles.
   return(sort(perc.vec))
```

```
}



# ----------------
# This function returns a function which will give an estimate of
# the cumulative probability of a specified X value based on the
# Kaplan-Meier emperical probability distribution.  It takes into
# account non-detects in the probability calculations.  Data values
# equal to zero are assumed to be non-detects and are replaced by
# their corresponding detection limits, which may differ for different
# observations.  These calculations are as per the formulations in
# "ProUCL Version 4.1.00 Technical Guide (Draft)", May-2010,
# Document ID:  EPA/600/R-07/041, pages 109-110.

km.distr <- function(x, detect.limits) {

   # Preliminary, basic calculations.
   n.total <- length(x)
   tf.nds <- (x == 0)
   x[tf.nds] <- detect.limits[tf.nds]

   # Order the data and corresponding variables.
   ord <- order(x, decreasing=FALSE)
   x <- x[ord]
   tf.nds <- tf.nds[ord]
   tf.detects <- (! tf.nds)
   detect.limits <- detect.limits[ord]

   # Get info on the detections and ND's.
   wh.nds <- which(tf.nds)
   wh.detects <- which(tf.detects)
   n.detects <- length(wh.detects)
   n.nds <- n.total - n.detects
   k.plus.1 <- n.total - n.detects + 1

   # Values used in the KM formula.
   x.prime <- unique(x[tf.detects])
   n.prime <- length(x.prime)
   mj <- table(x[tf.detects])
   nj <- cumsum(mj)
   x1.prime <- x.prime[1]
   xn.prime <- x.prime[n.prime]

   # Return the function that calculates KM ECDF.
   return(
      function(z) {
         p <- NULL
         for(y in z) {

            # Calculate the cdf probability depending on the value of deviates.
            p <- c(p,
               if(y >= xn.prime) {
                  1
               } else if(y >= x1.prime) {
                  wh.as.big <- which(x.prime > y)
                  prod( (nj[wh.as.big] - mj[wh.as.big]) / nj[wh.as.big] )
               } else if(y >= x[1]) {
                  wh.as.big <- which(x.prime > x1.prime)
                  prod( (nj[wh.as.big] - mj[wh.as.big]) / nj[wh.as.big] )
               } else {
                  0
               }
            )
         }
```

```
            return(p)
        }
    )
}
```

Appendix C

R code for a nonparametric UTL based upon bootstrapping the Chebyshev UPL calculation as given in Equation 5-2 of the ProUCL Version 4.1.00 Technical Guide.

```
# ----------------
# This function returns a function which will give an estimate of
# the cumulative probability of a specified X value based on the
# Kaplan-Meier emperical probability distribution.  It takes into
# account non-detects in the probability calculations.  Data values
# equal to zero are assumed to be non-detects and are replaced by
# their corresponding detection limits, which may differ for different
# observations.  These calculations are as per the formulations in
# "ProUCL Version 4.1.00 Technical Guide (Draft)", May-2010,
# Document ID:  EPA/600/R-07/041, pages 109-110.

km.distr <- function(x, detect.limits) {

   # Preliminary, basic calculations.
   n.total <- length(x)
   tf.nds <- (x == 0)
   x[tf.nds] <- detect.limits[tf.nds]

   # Order the data and corresponding variables.
   ord <- order(x, decreasing=FALSE)
   x <- x[ord]
   tf.nds <- tf.nds[ord]
   tf.detects <- (! tf.nds)
   detect.limits <- detect.limits[ord]

   # Get info on the detections and ND's.
   wh.nds <- which(tf.nds)
   wh.detects <- which(tf.detects)
   n.detects <- length(wh.detects)
   n.nds <- n.total - n.detects
   k.plus.1 <- n.total - n.detects + 1

   # Values used in the KM formula.
   x.prime <- unique(x[tf.detects])
   n.prime <- length(x.prime)
   mj <- table(x[tf.detects])
   nj <- cumsum(mj)
   x1.prime <- x.prime[1]
   xn.prime <- x.prime[n.prime]

   # Return the function that calculates KM ECDF.
   return(
      function(z) {
         p <- NULL
         for(y in z) {

            # Calculate the cdf probability depending on the value of deviates.
            p <- c(p,
               if(y >= xn.prime) {
                  1
               } else if(y >= x1.prime) {
                  wh.as.big <- which(x.prime > y)
                  prod( (nj[wh.as.big] - mj[wh.as.big]) / nj[wh.as.big] )
               } else if(y >= x[1]) {
                  wh.as.big <- which(x.prime > x1.prime)
                  prod( (nj[wh.as.big] - mj[wh.as.big]) / nj[wh.as.big] )
               } else {
                  0
```

```
                }
            )
        }
        return(p)
    }
  )
}


# ----------------
# This function calculates the mu-hat and se-hat(mu-hat) estimators based on the KM
# distribution (as per the ProUCL Technical Guide (Version 4.1, from the EPA).
# Returns a vector containing mu-hat and se-hat.

km.mean.stats <- function(x, detect.limits) {

   # Get the KM cdf.
   km.cdf <- km.distr(x, detect.limits)

   # Preliminary, basic calculations.
   n.total <- length(x)
   tf.nds <- (x == 0)
   x[tf.nds] <- detect.limits[tf.nds]

   # Order the data and corresponding variables.
   ord <- order(x, decreasing=FALSE)
   x <- x[ord]
   tf.nds <- tf.nds[ord]
   tf.detects <- (! tf.nds)
   detect.limits <- detect.limits[ord]

   # Get info on the detections and ND's.
   wh.nds <- which(tf.nds)
   wh.detects <- which(tf.detects)
   n.detects <- length(wh.detects)
   n.nds <- n.total - n.detects

   # Get the unique values of x, x', and include x_0=0.
   x.prime <- unique(x[tf.detects])
   n.prime <- length(x.prime)
   mj <- as.vector(table(x[tf.detects]))
   nj <- cumsum(mj)


   # Loop through all values of xp (except x_0).
   se2.hat <- 0
   a_i <- 0

   # First iteration for mu-hat.
   mu.hat <-x.prime[1] * (km.cdf(x.prime[1]) - km.cdf(0))

   for(i in if(n.prime >= 2) 2:n.prime else NULL) {

      # Calculate mu-hat.
      mu.hat <- mu.hat  +   x.prime[i] * (km.cdf(x.prime[i]) - km.cdf(x.prime[i-1]))

      # Calculate a_i.
      a_i <- a_i  +   (x.prime[i] - x.prime[i-1]) * km.cdf(x.prime[i-1])

      # Calculate sigma.sq-hat.
      se2.hat <- se2.hat + a_i*a_i * mj[i] / ( nj[i] * (nj[i] - mj[i]) )
   }

   se2.hat <- se2.hat * (n.total - n.nds) / (n.total - n.nds - 1)
```

```
    x[tf.nds] <- x[tf.nds] * 0.5
    sse <- 0
    for(i in 1:n.total) {
        sse <- sse + (x[i] - mu.hat)^2
    }

    sd2.hat <- sse/(n.total - 1)

    return( list(mu.hat=mu.hat, se.hat=sqrt(se2.hat), sd.hat=sqrt(sd2.hat)) )
}



# ----------------
# This function returns a function which will give an estimate of
# the cumulative probability of a specified X value based on the
# Kaplan-Meier emperical probability distribution.  It takes into
# account non-detects in the probability calculations.  Data values
# equal to zero are assumed to be non-detects and are replaced by
# their corresponding detection limits, which may differ for different
# observations.  These calculations are as per the formulations in
# "ProUCL Version 4.1.00 Technical Guide (Draft)", May-2010,
# Document ID:  EPA/600/R-07/041, pages 109-110.

km.distr <- function(x, detect.limits) {

    # Preliminary, basic calculations.
    n.total <- length(x)
    tf.nds <- (x == 0)
    x[tf.nds] <- detect.limits[tf.nds]

    # Order the data and corresponding variables.
    ord <- order(x, decreasing=FALSE)
    x <- x[ord]
    tf.nds <- tf.nds[ord]
    tf.detects <- (! tf.nds)
    detect.limits <- detect.limits[ord]

    # Get info on the detections and ND's.
    wh.nds <- which(tf.nds)
    wh.detects <- which(tf.detects)
    n.detects <- length(wh.detects)
    n.nds <- n.total - n.detects
    k.plus.1 <- n.total - n.detects + 1

    # Values used in the KM formula.
    x.prime <- unique(x[tf.detects])
    n.prime <- length(x.prime)
    mj <- table(x[tf.detects])
    nj <- cumsum(mj)
    x1.prime <- x.prime[1]
    xn.prime <- x.prime[n.prime]

    # Return the function that calculates KM ECDF.
    return(
        function(z) {
            p <- NULL
            for(y in z) {

                # Calculate the cdf probability depending on the value of deviates.
                p <- c(p,
                    if(y >= xn.prime) {
                        1
                    } else if(y >= x1.prime) {
                        wh.as.big <- which(x.prime > y)
```

```
                prod( (nj[wh.as.big] - mj[wh.as.big]) / nj[wh.as.big] )
            } else if(y >= x[1]) {
                wh.as.big <- which(x.prime > x1.prime)
                prod( (nj[wh.as.big] - mj[wh.as.big]) / nj[wh.as.big] )
            } else {
                0
            }
        )
    }
    return(p)
  }
 )
}


# ----------------
# This function calculates the mu-hat and se-hat(mu-hat) estimators based on the KM
# distribution (as per the ProUCL Technical Guide (Version 4.1, from the EPA).
# Returns a vector containing mu-hat and se-hat.

km.mean.stats <- function(x, detect.limits) {

   # Get the KM cdf.
   km.cdf <- km.distr(x, detect.limits)

   # Preliminary, basic calculations.
   n.total <- length(x)
   tf.nds <- (x == 0)
   x[tf.nds] <- detect.limits[tf.nds]

   # Order the data and corresponding variables.
   ord <- order(x, decreasing=FALSE)
   x <- x[ord]
   tf.nds <- tf.nds[ord]
   tf.detects <- (! tf.nds)
   detect.limits <- detect.limits[ord]

   # Get info on the detections and ND's.
   wh.nds <- which(tf.nds)
   wh.detects <- which(tf.detects)
   n.detects <- length(wh.detects)
   n.nds <- n.total - n.detects

   # Get the unique values of x, x', and include x_0=0.
   x.prime <- unique(x[tf.detects])
   n.prime <- length(x.prime)
   mj <- as.vector(table(x[tf.detects]))
   nj <- cumsum(mj)


   # Loop through all values of xp (except x_0).
   se2.hat <- 0
   a_i <- 0

   # First iteration for mu-hat.
   mu.hat <-x.prime[1] * (km.cdf(x.prime[1]) - km.cdf(0))

   for(i in if(n.prime >= 2) 2:n.prime else NULL) {

      # Calculate mu-hat.
      mu.hat <- mu.hat  +   x.prime[i] * (km.cdf(x.prime[i]) - km.cdf(x.prime[i-1]))

      # Calculate a_i.
      a_i <- a_i  +   (x.prime[i] - x.prime[i-1]) * km.cdf(x.prime[i-1])
```

```
        # Calculate sigma.sq-hat.
        se2.hat <- se2.hat + a_i*a_i * mj[i] / ( nj[i] * (nj[i] - mj[i]) )
    }

    se2.hat <- se2.hat * (n.total - n.nds) / (n.total - n.nds - 1)

    x[tf.nds] <- x[tf.nds] * 0.5
    sse <- 0
    for(i in 1:n.total) {
        sse <- sse + (x[i] - mu.hat)^2
    }

    sd2.hat <- sse/(n.total - 1)

    return( list(mu.hat=mu.hat, se.hat=sqrt(se2.hat), sd.hat=sqrt(sd2.hat)) )
}



# ----------------
# This function returns a Chebyshev upper prediction limit given estimates
# of the pop mean and pop sd.  These calculations are as per the formulations in
# "ProUCL Version 4.1.00 Technical Guide (Draft)", May-2010,
# Document ID:  EPA/600/R-07/041, page 129.

chebyshev.upl <- function(mean, sd, se, confidence) {
    return(
        mean +
            sqrt(
                (confidence / (1 - confidence) ) *
                (sd*sd + se*se)
            )
    )
}



# ----------------
# This function performs a bootstrap on the given sample data, x, and
# then creates a function that will give probabilities from the
# Chebyshev UPL.  This method utilizes the KM ECDF to get the mean, sd,
# and SE for the Chebyshev UPL calculations.  It iterates this process the
# number of times specified by num.samples, and returns that many
# bootstrapped percentile estimates, sorted in ascending order.  A 95%
# upper tolerance limit would be given by z[0.95*num.samples], where
# z is the vector of ordered bootstrap percentiles returned from this
# function.
#
# This technique may be impeded by the fact that ECDF is a stepwise function
# and can only produce percentiles that are in the original data set.  This
# could likely be improved by using a kernel density estimate calcualted from
# the bootstrap sample instead, and similarly determining the specified
# percentile, iterating num.sample times and obtaining a UTL in the same
# manner.

bootstrap.km.chebyshev.percent <-
        function(x, detect.limits, num.samples, coverage, conf) {

    # Some observed values are equal to their DL, so scale back the DL
    # to prevent treating them equal.
    detect.limits <- 0.999*detect.limits

    n.total <- length(x)
    indices <- 1:n.total
```

```
    # Bootstrap the specified percentile the specified number of times.
    # coverage.list <- as.list(coverage)
    # names(coverage.list) <- as.character(coverage.list)
    # upls <- data.frame(coverage.list)
    # upls <- upls[-1, ]
    upls <- matrix(nrow=0, ncol=length(coverage))

    i <- 0
    while(i < num.samples) {

        # Get a bootstrap sample and obtain the KM ECDF for it.
        bs.indices <- sample(indices, n.total, replace=TRUE)
        y <- x[bs.indices]
        # Ensure at least 5 detects
        if( sum(y != 0) < 5 ) next
        # km.cdf <- km.distr(y, detect.limits[bs.indices])

        # Get the specified percentile from the ECDF.
        # z <- sort(unique(y))

        # Get the KM stats.
        km.mean.stats <- km.mean.stats(y, detect.limits[bs.indices])
        upl <- chebyshev.upl(km.mean.stats$mu.hat, km.mean.stats$sd.hat,
                             km.mean.stats$se.hat, confidence=coverage)

        upls <- rbind(upls, upl)

        i <- i + 1
    }

    upl.vector <- NULL
    index <- ceiling(conf*num.samples)
    for(i in 1:ncol(upls)) {
        upl.vector <- c(upl.vector, (sort(upls[, i]))[index])
    }

    # Return all of the bootstrapped percentiles.
    return(upl.vector)
}
```