# Statool: A Tool for Distribution Envelope Determination (DEnv), an Interval-Based Algorithm for Arithmetic on Random Variables

Daniel Berleant, Lizhi Xie, and Jianzhong Zhang
Department of Electrical and Computer Engineering
Iowa State University, Ames, Iowa 50011, USA
berleant@iastate.edu

## Abstract

We present Statool, a software tool for obtaining bounds on the distributions of sums, products, and various other functions of random variables where the dependency relationship of the random variables need not be specified. Statool implements the DEnv algorithm, which we have described previously [4] but not implemented. Our earlier tool addressed only the much more elementary case of independent random variables [3]. An existing tool, RiskCalc [13], also addresses the case of unknown dependency using a different algorithm [33] based on copulas [23], while descriptions and implementations of still other algorithms for similar problems will be reported soon [17] as the area proceeds through a phase of rapid development.

## 1 Introduction

The problem of determining derived distributions, the distributions of random variables whose samples are a function of samples of other random variables, has received considerable attention. Springer's monograph [30] is fairly comprehensive up to its time of publication. Much of the subsequent work has focused on copulas [23], which have motivated a number of conferences [1][11][12][27]. Much of the existing work addresses analytical techniques. A shortcoming of the analytical approach is its tendency to produce results applying to specific classes of distributions, such as normal, lognormal, etc. Numerical methods form an alternative approach that tends to be applicable to a wider class of distributions. Monte Carlo is the classical numerical method, but has some serious shortcomings [14], such as difficulties in safely handling problems in which dependency relationships among random variables are unknown, or in which distributions are not fully specified. Non-Monte Carlo numerical methods rely on discretization of distributions followed by computation on the discretized forms.

Algorithms for non-Monte Carlo, numerical computation of derived distributions have been known since at least as early as 1968 [18]. Early algorithms [10][18][19][22] assume that distributions whose samples are summed, multiplied, etc., to yield the derived distribution of interest are independent. Furthermore they rely on discretizations that approximate, as discretizations often do, implying that results are not validated. Discretizations that enclose rather than approximate can lead to validated results, and also can lead to the ability to compute a random variable derived from two others, one described with a distribution function and the other described by an interval [2]. Algorithms cited in this paragraph are based on numerical convolution.

Algorithms for computing distributions of derived random variables face an additional challenge when the dependency relationship of the input random variables is unknown, since a relatively straightforward application of numerical convolution is no longer sufficient. This situation is important because often information about the precise dependency relationship of two random variables is unknown, and while in practice independence is often assumed for the sake of tractability, obviously it is better to avoid making unjustified assumptions. When the dependency is unknown, it is not generally possible to describe the derived random variable with a single distribution, since different plausible dependency relationships typically result in different derived distributions. However, the set of plausible derived distributions can be bounded. Williamson and Downs describe a copula-based approach called Probabilistic Arithmetic [33] which was subsequently

implemented in RiskCalc [13]. A non-copula-based approach, Distribution Envelope Determination (DEnv), has also been described [4]. An implementation of DEnv, which has not heretofore been published, is the focus of this paper. We describe Statool, a tool implementing DEnv and containing no code from our previous tool [3].

Regan et al. [26] show that DEnv and Probabilistic Arithmetic are equivalent in important ways, and the results produced by both methods on some standard problems [24] are consistent [5][15]. However, the non-copula approach of DEnv is easier to understand for many people, and will likely remain so until copulas become a standard part of the course curricula experienced by individuals subsequently needing an understanding of the algorithms. Recently other approaches have been presented by Fetz and Oberguggenberger [16], Lodwick [21], Red-Horse [25], and Tonon [32].

Assuming independence on the one hand, and assuming nothing about dependency on the other, are extremes of a continuum. Intermediate situations involve partial information about the dependency relationship between the input random variables contributing to a derived random variable. Currently Statool supports the need to use partial information about dependency by allowing a value or range for the Pearson correlation, the most common kind of correlation, between two input random variables to be specified [6]. Incorporating into algorithms various other useful information about dependency that may be available or that application domain experts are likely to feel is reasonable to assume about some problems is a promising area of investigation. We believe much opportunity remains for advances in this area.

# 2 The Tool

Statool has been applied to problems in electric power [7][29], value at risk [29], activity networks [9], and reliability [8]. This section explains the new capabilities of Statool. Functionality that reimplements similar functionality in our previously described tool [3] will only be mentioned as needed for clarity.

Figure 1 shows the Statool main screen. As in [3] the major portion of the screen shows three panes. The top two, arbitrarily labeled **X** and **Y**, describe two random variables used as inputs to a derived random variable, whose description appears in the bottom pane, labeled **Z**. All three variables are shown using left and right envelopes that bound their cumulative distributions. Note that for the input variables **X** and **Y**, the southeast corners of the left envelope touch northwest corners of the right envelope in order to express the effects of discretization reliably by bounding rather than approximation (Figure 2).

## 2.1 Specifying the discretization

The internal representation of a random variable is in terms of intervals and probabilities associated with them. This flexible representation is easily graphed with envelopes bounding the space through which the random variable's cumulative distribution function (CDF) may pass (Figure 2). It is also easily graphed as a histogram (Figure 3) when, as in the case of PDFs (probability density functions) that have been discretized, the intervals collectively partition the support of the random variable. However when intervals overlap, the bars of the histogram would overlap as well, constituting a graphic representation more general than the usual histogram concept.

Histograms are user-friendly in important ways, since they can clearly show regions of maximum density, and shape properties like skewedness and bimodality. A serious deficiency of histograms is their inability to express envelopes that do not touch at corners (Figure 1, 3rd pane). However, this is often not a problem for input distributions. Thus Statool uses histograms as the basis of a graphical input distribution editor (click on a histogram, click "edit," then left click above the top of a bar to make it higher, below the top to lower it, or right click above or below to widen or narrow it). Alternatively, files specifying a set of intervals and their associated probabilities may be edited in a text editor and saved as plain text in a file with the suffix "`.idf`" (for "intermediate distribution format"). Figure 4 shows an example.

## 2.2 Deriving Distributions from More than Two Inputs

Computing envelopes around a distribution derived from three random variables in Statool requires the user to compute an intermediate result from two of them, then move the intermediate result from the result pane into the top pane (by either saving it to a file and then reading the file in or by clicking "switch" in

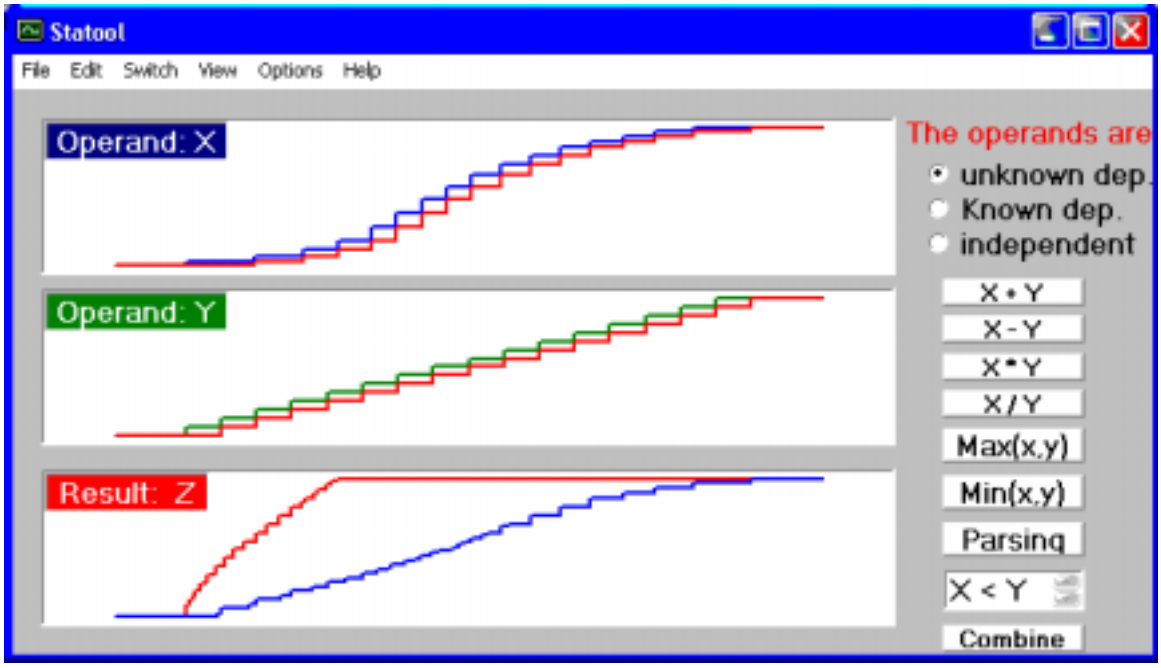


Figure 1: Statool main screen, showing the result (bottom panel) of a multiplication operation on operands in the top and middle panels, when the dependency relationship between the operands is unknown.

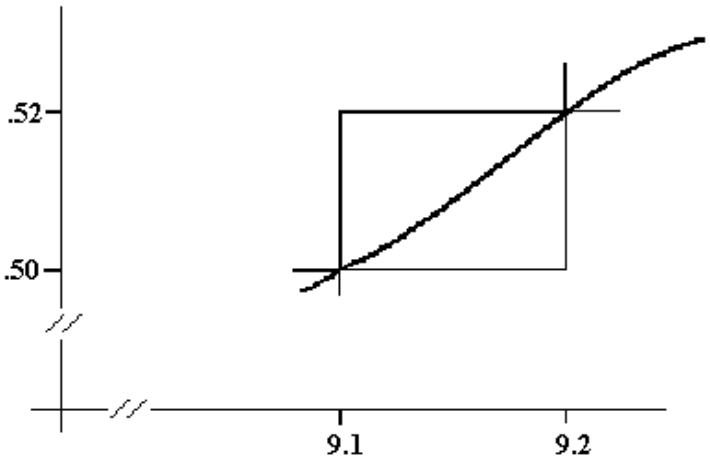


Figure 2: A portion of a cumulative distribution function and its enclosing discretization. The discretization specifies that a cumulation of 0.02 occurs between horizontal axis values 9.1 and 9.2. In other words, $p([9.1, 9.2]) = 0.02$. The envelopes, a staircase-shaped left envelope above the curve and another (the right envelope) below it, do not specify how the 0.02 mass is distributed within $[9.1, 9.2]$, unlike the curve itself. The full discretization partitions the *support* (the region of the horizontal axis over which the height of the curve is $> 0$ and $< 1$) into intervals and associates a probability with each interval.

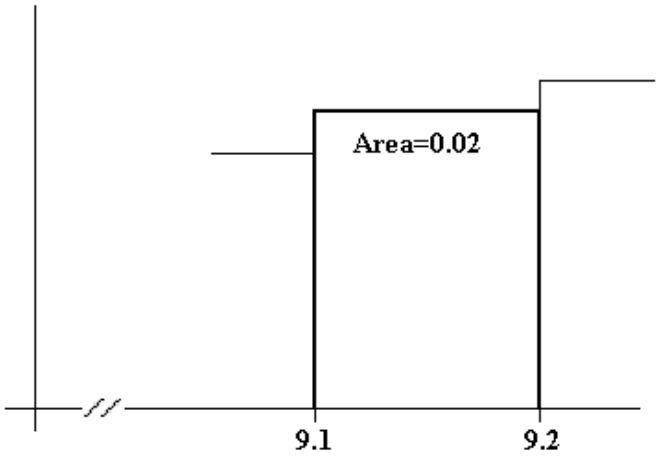


Figure 3: Portion of a histogram expressing the same information as in Figure 2. The flat top of the bars is a graphical convention only and is not intended to imply anything about the distribution of the probability mass (represented as the area) of a bar over its interval on the horizontal axis.

```
4,0
3,5,.2
5.2,6,.3
7,7.000001,.3
5.8,9,.2
```

Figure 4: An `.idf` file which may be saved and read in, or created and edited, either graphically in Statool or as a plain text file with a text editor. The numeral `4` in the first line states that there are 4 intervals in the specification (implying 4 more lines in the file). The `0` in the first line is reserved for future use. The remaining 4 lines contain three comma-separated numbers each. The first states an interval low bound, the second the interval high bound, and the third the probability mass associated with the interval. This example contains a gap between the first and second intervals, an overlap between the second and fourth, and a narrow third interval which is inside the bounds of the fourth. (True impulses are specified using a text editor and a `.smp` ("SaMPle") format file.)

the top menu bar), then place the 3rd operand in the middle pane, and then invoke the computation that gives the overall result. Some expressions in three variables need to be transformed into an algebraically equivalent form that allows this. For example, (**B***A+**B***C) can be restated as **B***(**A**+**C**), allowing **A**+**C** to be an intermediate result which is then multiplied with **B**. If **A**, **B**, and **C** are independent, or if their dependency relationships are all unknown, this works. Certain cases of partially known dependency can also be handled. For example, in computing **C***(**A**+**B**), if **A** and **B** have a known correlation, but they are either both independent of **C** or the dependency relationship between each of them and **C** is unknown, one can compute **A**+**B** to give an intermediate result which is multiplied with **C**. Other cases of partially known dependency cannot be so easily addressed. For example if the correlation between **B** and **C** is known and **A** is independent of **B**, then **C***(**A**+**B**) might be computed by computing (**A***C+**B***C), which in turn might be addressed by computing **D**=**A***C, **E**=**B***C, and finally **D**+**E** with no assumption about the dependency between **D** and **E**. However the problem structure does imply information about the dependency between **D** and **E** although Statool does not use this knowledge, so the envelopes obtained may be too widely separated, enclosing but weaker than the results that are potentially achievable.

Other expressions in three variables are even less tractable. **A***B+**B***C+**C***A could be computed by computing each term, then adding two of them, then adding that intermediate result to the third term. However, the additions of terms would be done in Statool without assuming any dependency relationships among the terms, despite their shared variables, because the correlations among the terms are not easily obtainable and the only type of partial dependency information Statool can currently use is correlation. The end result would be enclosing envelopes that are more widely separated than they would be if Statool could use the dependency information available. In principle, a solution to these problems is to handle expressions in three variables directly, rather than decomposing them into sequences of computations on two variables. This would add a third dimension to the joint distribution tableaus used by the DEnv algorithm (see Appendix), a generalization of the algorithm that we have not implemented. Taking this further, to handle arbitrary functions of $n$ distributions would require working with an $n$-dimensional joint distribution tableau.

Fortunately many functions of multiple distributions can be evaluated without loss of information by combining two of them, then combining that intermediate result with another to get a next-level intermediate result, and continuing that way until all the distributions are accounted for. Using an intermediate result as the input to a follow-on computation requires converting the intermediate result, which consists of left and right envelopes, into a list of intervals and associated probabilities (the intermediate distribution format described earlier). This is done by tiling the space between the envelopes with horizontal stripes. The bottom and top edges of the stripes are positioned so that each stripe is a rectangle, for which one side is on a vertical segment of the left envelope and the opposite side is on a vertical segment of the right envelope, defining an interval. The bottom-to-top height defines a probability associated with that interval. A figure and full description of this appears in [26].

## 2.3 Operations Used in Deriving Distributions

For distributions **X**, **Y** and **Z**, we write **Z**=**X**+**Y** to mean **Z** is the distribution of samples obtained by summing samples of **X** and **Y**. Statool addresses the operations $\{+, -, *, /\}$ on independent inputs, as did our previous tool [3], but in addition, also addresses these operations on input distributions of unknown dependency or with values or intervals given for their correlation. Statool also handles some new operations, described next.

**Z** = $max$(**X**, **Y**).
If $x$ is a sample of distribution **X** and $y$ is a sample of **Y**, the corresponding sample of **Z** is $z = max(x, y)$. This can be useful for problems such as determining the time to complete a project consisting of two concurrent tasks. (The completion time for the project is the later of the two individual completion times.) Since Statool's discretization process partitions the support of each input distribution into a set of intervals, the definition for $max$ given above must be modified to the case of intervals. This is done by defining $max(X, Y)$ for intervals $X$ and $Y$ to be $[max(\underline{X}, \underline{Y}), max(\overline{X}, \overline{Y})]$. This makes sense for the concurrent task completion time problem and similar problems. Figure 5 shows examples of distributions **Z** derived using **Z** = $max$(**X**, **Y**).

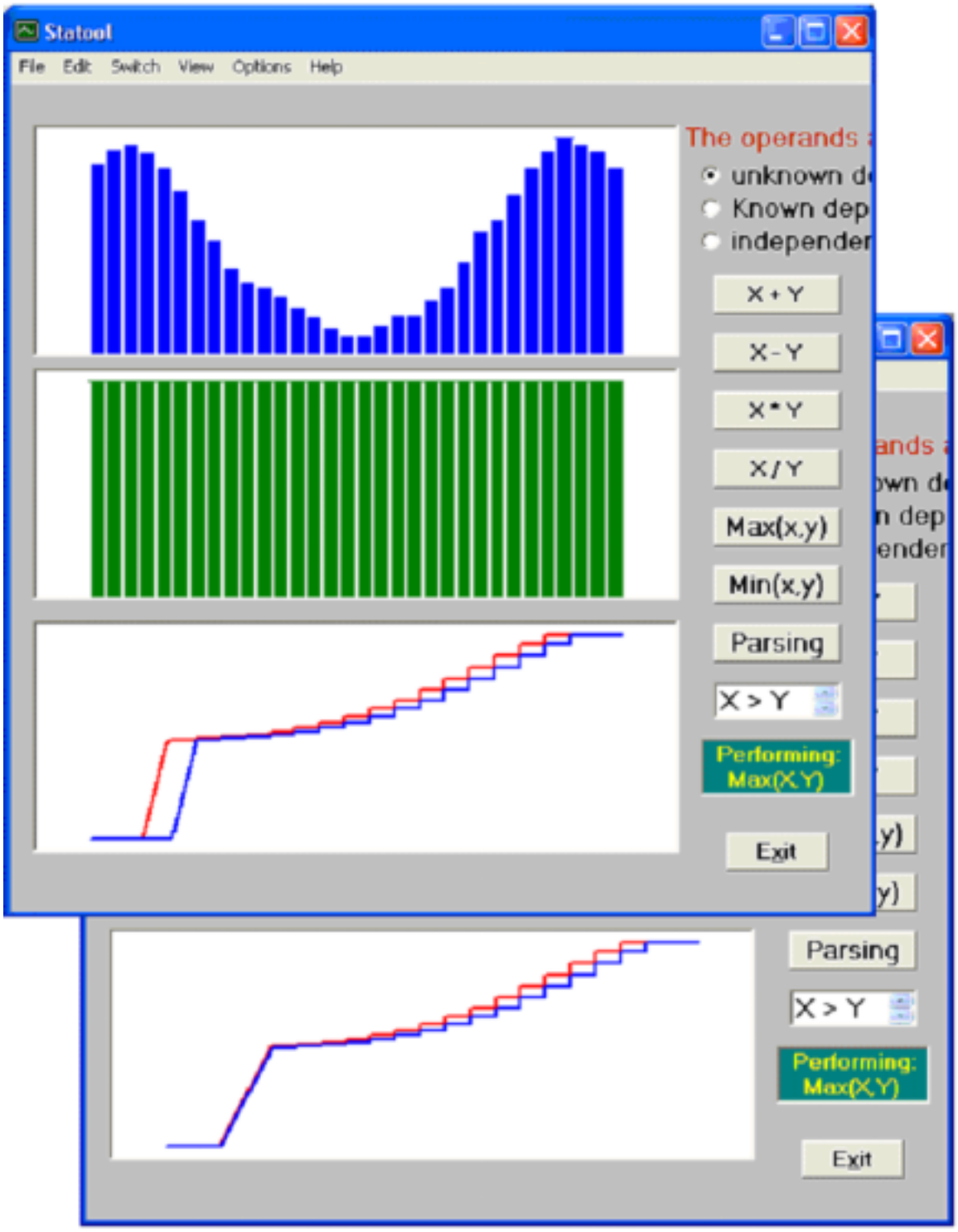


Figure 5: The *max* operation is invoked on the distributions in the 1st pane at top, **X**, and the 2nd pane, **Y**. The result when their dependency is unknown is shown in the third pane. Assuming independence results in envelopes that are closer together (fourth pane). The support for **X** is [6,9] and for **Y** is [7.4,7.6], so the support for the results of the *max* operation is [7.4, 9], with **Y** dominating the results over [7.4,7.6] (hence the relatively straight appearance over that range) and **X** dominating over (7.6,9].

$\mathbf{Z} = min(\mathbf{X}, \mathbf{Y})$.
If $x$ is a sample of $\mathbf{X}$ and $y$ is a sample of $\mathbf{Y}$, the corresponding sample of $\mathbf{Z}$ is $z = min(x, y)$. This can be useful for problems such as determining the time to failure of the first component to fail in a two-component system. This failure time is the earlier of the two individual failure times. For Statool, the definition for $min$ must be intervalized. Thus, we define $min(X, Y)$ for intervals $X$ and $Y$ to be $[min(\underline{X}, \underline{Y}), min(\overline{X}, \overline{Y})]$. This makes sense for the two-component failure problem and similar problems. Figure 6 shows examples of distributions $\mathbf{Z}$ derived using $\mathbf{Z} = min(\mathbf{X}, \mathbf{Y})$.

$\mathbf{Z} = (\mathbf{X} < \mathbf{Y})$.
This is defined such that if $x$ is a sample of $\mathbf{X}$ and $y$ is a sample of $\mathbf{Y}$, the corresponding sample of $\mathbf{Z}$ is 1 if $x < y$ and 0 otherwise. This can be useful for problems such as determining the probability that a given one of two events will occur first. For Statool, this definition must be intervalized. Similarly to Kazakov [20] we define $X < Y$ for intervals $X$ and $Y$ to be

$$\left\{ \begin{array}{lll} 0 & if & \underline{X} \geq \overline{Y} \\ 1 & if & \overline{X} < \underline{Y} \\ [0,1] & otherwise & \end{array} \right\}.$$

An alternative definition would assign the set {0,1} instead of the interval [0,1].

In Statool, distributions $\mathbf{X}$ and $\mathbf{Y}$ are discretized as sets of intervals and their associated probabilities. Some intervals in the discretization of $\mathbf{X}$ might be less than some in the discretization of $\mathbf{Y}$, others might be greater, and still others might overlap. Minimizing the sum of the probabilities (see Appendix) of cases where intervals in $\mathbf{X}$ are less than intervals in $\mathbf{Y}$ gives a low bound for $p(x < y)$, and maximizing the sum of the probabilities of cases where intervals in $\mathbf{X}$ are not $\geq$ intervals in $\mathbf{Y}$ gives a high bound for $p(x < y)$. (The definition of $\geq$ follows.)

$\mathbf{Z} = (\mathbf{X} \leq \mathbf{Y})$, $\mathbf{Z} = (\mathbf{X} > \mathbf{Y})$, and $\mathbf{Z} = (\mathbf{X} \geq \mathbf{Y})$.
Definitions for these appear in the following chart.

| $\mathbf{Z} = (\mathbf{X} \leq \mathbf{Y})$ | $\mathbf{Z} = (\mathbf{X} > \mathbf{Y})$ | $\mathbf{Z} = (\mathbf{X} \geq \mathbf{Y})$ |
|---|---|---|
| Define $X \leq Y$ for intervals as | Define $X > Y$ for intervals as | Define $X \geq Y$ for intervals as |
| $\left\{ \begin{array}{lll} 0 & if & \underline{X} > \overline{Y} \\ 1 & if & \overline{X} \leq \underline{Y} \\ [0,1] & otherwise. & \end{array} \right\}$ | $\left\{ \begin{array}{lll} 0 & if & \overline{X} \leq \underline{Y} \\ 1 & if & \underline{X} > \overline{Y} \\ [0,1] & otherwise. & \end{array} \right\}$ | $\left\{ \begin{array}{lll} 0 & if & \overline{X} < \underline{Y} \\ 1 & if & \underline{X} \geq \overline{Y} \\ [0,1] & otherwise. & \end{array} \right\}$ |

$\mathbf{Z} = f(\mathbf{X}, \mathbf{Y})$.
A function that combines $\mathbf{X}$ and $\mathbf{Y}$ by invoking a supported arithmetic operation one time may be invoked easily in Statool as described earlier. Expressions that involve more than one operation can sometimes be evaluated in more than one step, using the result of one step as an intermediate result to serve as the input to the next step. This approach suffers from potential problems similar to those described in Section 2.2. For example, consider the expression $\mathbf{X}(\mathbf{X} + \mathbf{Y})$. Let $\mathbf{W} = \mathbf{X} + \mathbf{Y}$ be the intermediate result. The next step is to calculate $\mathbf{Z} = \mathbf{X} * \mathbf{W}$. However, if $\mathbf{X}$ and $\mathbf{Y}$ are independent, then $\mathbf{X}$ and $\mathbf{W}$ are not independent, and their precise dependency relationship cannot currently be expressed in Statool. Statool can compute $\mathbf{Z} = \mathbf{X} * \mathbf{W}$ with no assumption at all about the dependency relationship between $\mathbf{X}$ and $\mathbf{W}$, but this ignores the fact that there is a partial dependency present, and so the computation of $\mathbf{Z}$ is likely to be weaker than desired (that is, the envelopes for $\mathbf{Z}$ are more widely separated than they would be if the dependency was taken into account). Similar arguments apply for other dependency relationships between $\mathbf{X}$ and $\mathbf{Y}$ besides independence, and for the case where the dependency relationship is unknown.

Since operations on random variables in Statool are computed by discretizing distributions as sets of intervals and performing the operation on those intervals (see Appendix), the solution in this case is to compute, for intervals $X_i$ in the discretization of $\mathbf{X}$ and $Y_j$ in the discretization of $\mathbf{Y}$, the intervals $Z_{ij} = X_i(X_i + Y_j)$ in one step, using an interval method that produces tight enclosures for each $Z_{ij}$. Fortunately this is trivial for expressions that are monotonic over the box defined by the interval-valued arguments. Statool allows an expression in two variables $x$ and $y$ to be typed in (after clicking the "Parsing" button shown in Figures 1, 5 & 6).

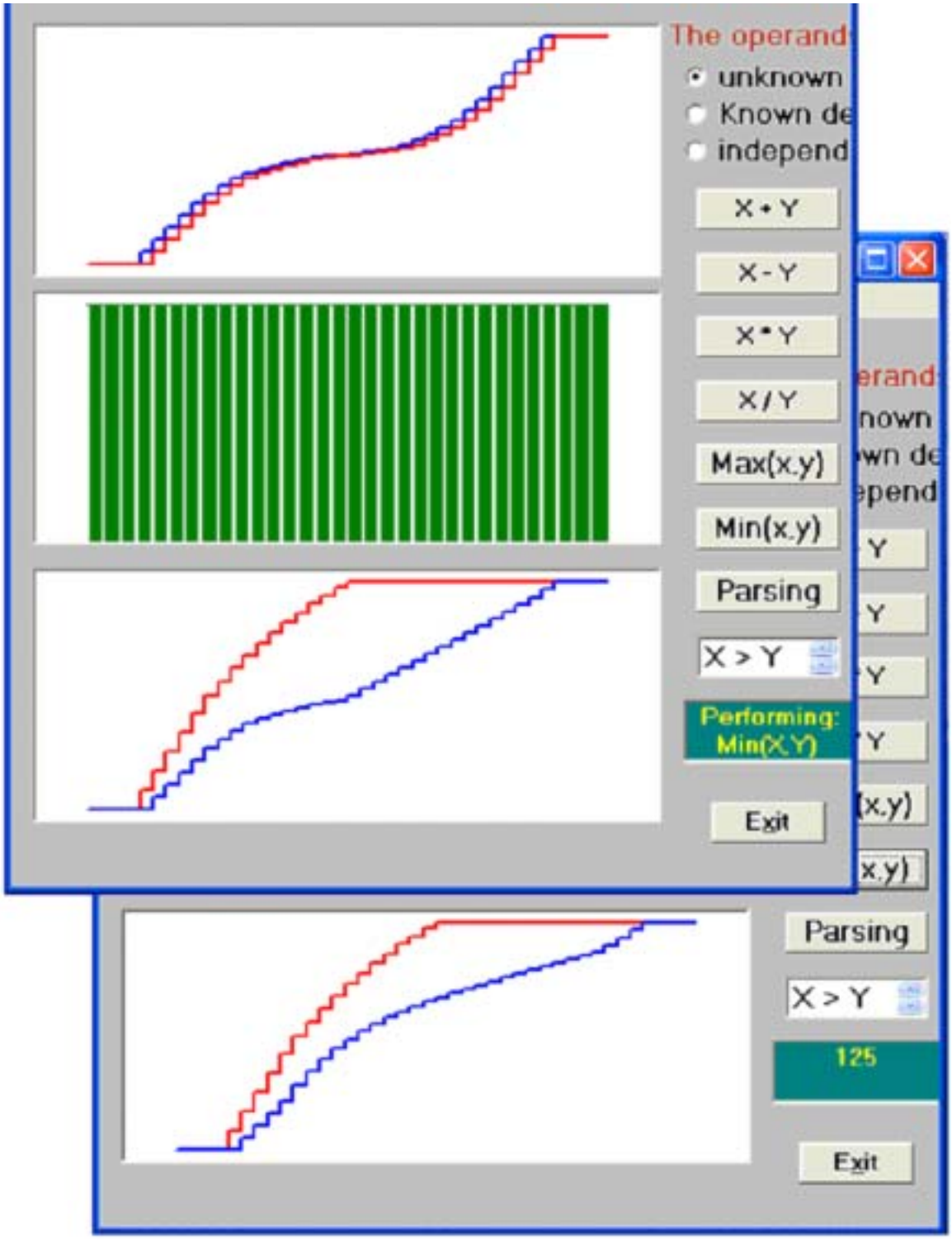


Figure 6: The *min* operation is invoked on the distributions shown in envelope form in the **X** pane (top), and histogram form in the **Y** pane (2nd from top). The support for **X** is [6,9] as in the Figure for *max*, but unlike there the support for **Y** here is the same as for **X**, [6,9]. The result when the dependency between **X** and **Y** is unknown is shown in the 3rd pane. Specifying that **X** and **Y** have a correlation in $[-1,-0.97]$ results in envelopes that are closer together (4th pane).

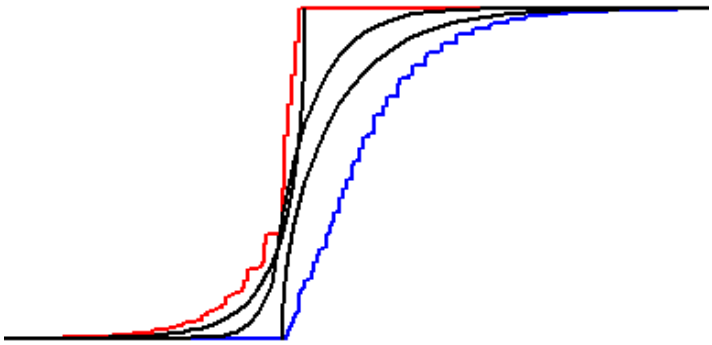

Figure 7: Envelopes produced by Statool enclose three distributions generated by Monte Carlo simulation under three conditions: independence and correlations of 1 and −1. The inputs were normal curves with mean and variance of 1, tail-trimmed to fit in [−3,−5] and with the trimmed mass evenly distributed over the interval. The enclosing envelopes bound the space of possible result distributions for any dependency relationship of the inputs. (Note that the MC-generated curves are not bounds because other dependency relationships on the inputs can lead to result curves that, for example, travel to the right of the rightmost of the three. The DEnv algorithm produces envelopes that are both rigorous and optimal in the sense of [4], which ignores machine arithmetic error, and weakens results to account for input curve discretization but only as much as necessary.)

## 2.4 The Software: Verification, Performance, and Platform

Statool's results have been checked against results obtained elsewhere (Section 2.4.1) and its speed has been measured for problems with various characteristics (Section 2.4.2). Statool is written using C for the algorithmic core and Visual Basic for the user interface, and due to the use of Visual Basic runs only on Windows systems. Statool may be downloaded free for non-commercial use [31].

### 2.4.1 Verification

Bugs are an ever-present danger in complex software. To help check the implementation of DEnv in Statool we ran it on a number of benchmark problems [24] that were the focus of a recent workshop [28]. Results produced by Statool verified those produced by RiskCalc [13] and vice versa in those cases where the results of these systems could be directly compared.

A separate check was done using Monte Carlo simulation to compute the product of two normal distributions that straddle zero asymmetrically (i.e. with a non-zero mean). Results were generated for three dependency assumptions. The three result distributions fit within the envelopes for this problem generated by Statool, further verifying the implementation (Figure 7).

### 2.4.2 Performance

*Independent operands.* This is the simplest case. The algorithm involves numerical convolution. Since the convolution process requires applying the requested arithmetic operation to the Cartesian product of the intervals in the discretization of distribution **X** and the intervals in the discretization of distribution **Y** to the get the joint distribution tableau (see Appendix), which is a discretized form of the joint distribution, the solution times are dominated by the fineness of the discretizations. If both distributions are discretized using histograms of $n$ bars each, the joint distribution tableau will have $n^2$ intervals. In the worst case no intervals share end points, so each interval contributes one value for which the height of the left envelope is computed and one for which the height of the right envelope is computed. Each of the up to $2n^2$ height computations causes each of the $n^2$ joint distribution tableau entries to be examined to see if its mass contributes to the height, so the time complexity is $O(n^4)$.

Empirical results for a PC running Windows with a 750 MHz Pentium III CPU for $n = 64$, Statool's current maximun $n$, using input distributions that were uniform over [1,10], showed that * and / took about 1

sec., and the other supported basic operations ($+, -, min, max, >, <, \geq$, and $\leq$) about $\frac{3}{4}$ sec. For expressions composed of more than one operation, the empirical running time was 11 sec. plus 1 sec. for each operation (evaluating $\mathbf{X} * \mathbf{Y}$ took 12 sec., $\mathbf{X} * \mathbf{Y} + \mathbf{X} * \mathbf{Y}$ took 14 sec.,..., $\mathbf{X} * \mathbf{Y} + \mathbf{X} * \mathbf{Y} + \mathbf{X} * \mathbf{Y} + \mathbf{X} * \mathbf{Y} + \mathbf{X} * \mathbf{Y}$ took 20 sec., and summing 10 of those terms took 29.5 sec.)

*Operands of unknown dependency.* This condition requires that the probability mass associated with cells in the joint distribution tableau be moved around among cells consistently with the marginal cells in such a way as to minimize or maximize the summed probabilities of specific subsets of the cells to obtain corresponding points on the right or left envelopes (see Appendix). A subset is defined by the value of the derived random variable whose envelope heights are desired, by whether the linear programming is to minimize (for a point on the right envelope) or maximize (for a point on the left envelope), and by the arithmetic operation. Thus each point computed for each envelope requires all of the processing of the independent case, plus the solution to a linear programming problem with approximately $n^2$ variables. For the case of unknown dependency the structure of the linear programming problems is simpler than in the general case and allows the transportation simplex algorithm to be used, which is faster than the simplex algorithm.

Using the same inputs as used for measuring run times for the independent case, the unknown dependency case exhibited an empirical time complexity of $O(n^6)$, where $n$ is the number of intervals in the discretization of each distribution or, if the discretizations contain different numbers of intervals $n_1$ and $n_2$, $n$ is the geometric mean $\sqrt{n_1 \cdot n_2}$. On a PC running Windows with a 750 MHz Pentium III CPU, solution times for sums of two operands are about 2 seconds for $n$ near 16, in the 1-2 minute range for $n$ near 32, and 1-2 hours for $n$ near 64. Statool currently has an upper limit on $n$ of 64 for distributions to be used as operands. This may be increased if applications are found that require it. Times are longer than predicted for small $n$ of about 16 or less. This is typical because algorithms often have complexities with lower order terms that are significant for smaller problems but are not accounted for in the big-O complexity expression, which is intended to capture complexity for large problems.

*Correlated operands.* When information about the correlation between input distributions is available, additional constraints are implied that the linear programming (LP) calls can use [6]. These new constraints result in LP problems that do not have the form required by the relatively efficient transportation simplex algorithm, so they must be solved by another linear programming algorithm. In this case Statool uses the simplex algorithm, a classical general purpose LP algorithm which, though not as fast as Karmarkar's method in principle, is simpler.

An empirical investigation of the computational complexity of the simplex algorithm calls that Statool makes when correlation constraints are present was undertaken. The solution times for the different LP problems occurring during computation of different points on an envelope can vary greatly, so the mean solution time over all the LP problems solved during the computation of a pair of envelopes was determined.

These mean solution times were recorded three times for each of the $+, -, *, /, max$, and $min$ operations. Of these three, one was for correlation=0, one was for correlation=0.98, and one was for correlation=$-0.98$. Six operations and three correlation conditions yield 18 mean LP solution times, for which the geometric mean was computed. This geometric mean figure was calculated for different values of $n$. Results were consistent with a time complexity of approximately $O(n^6)$ for the LP calls. Since the number of LP calls increases with problem size at up to $O(n^2)$, when correlation is used Statool has an overall time complexity of up to $O(n^6 n^2) = O(n^8)$. Typical overall envelope computation times on a PC running Windows with a 1000 MHz Pentium III CPU are on the order of days for multiplication when $n = 64$, hours for addition when $n = 64$, minutes for multiplication when $n = 32$, about a minute for addition when $n = 32$, and seconds for $n = 16$. Solution time often varies significantly for different correlation values in problems that are otherwise the same. Both the big-O complexity and empirical solution times could be significantly improved for larger problems if some of the up to $n^2$ points on each envelope that need to be computed for best possible results are in fact not computed, implying a tradeoff between speed and envelope quality.

# 3 Conclusion

A downloadable tool for arithmetic on random variables, Statool, has been described, and several applications mentioned. Future advances in the theory and implementation will be driven by the needs of application

problems. Such needs may include extending Statool to non-monotonic functions, using forms of partial information about dependency other than Pearson correlation, extending the data structures to handle discretizations of more than 64 intervals, and improving execution time for larger problems.

## Acknowledgements

The authors are grateful to Gerald B. Sheblé for motivating advances in Statool through extensive discussions on the needs of applications in the electric power industry, including power systems economics. Rujun Hu assisted in this, with Mei-Peng Cheong currently continuing work on the applications aspect of the project. This research was supported in part by funding from the Power Systems Engineering Research Center (PSERC) to G. Sheblé, D. Berleant, and R. Thomas.

## Appendix: the DEnv Algorithm

The DEnv (Distribution Envelope Determination) algorithm was discussed formally elsewhere [4]. It is reviewed here, for convenience informally but emphasizing the key intuitions.

DEnv finds envelopes between which the CDF of a derived distribution must be. By *derived distribution* is meant the distribution of a random variable whose sample values are a function of sample values of other random variables, termed *inputs.* Since different dependency relationships between inputs usually imply different derived distributions, if the dependency relationship is not fully specified then the derived CDF is typically not fully specified. In such cases, DEnv computes envelopes around the derived CDF. DEnv may be described in three major steps.

*Step 1: construct a joint distribution tableau and identify the constraints.* Each input is discretized by representing its PDF (probability density function) with intervals, each assigned an associated probability mass. These discretized inputs form the marginals of a *joint distribution tableau* (Figure 8, top).

Although the interior cell probability masses are not determined, they are constrained by *marginal constraints:* the probabilities of the interior cells in each row or column must sum to the probability of the row or column's marginal cell (Figure 8, bottom).

*Step 2: find bounds on the cumulative probability of the derived distribution at some value.* Each interior cell of a joint distribution tableau states a range over which its associated probability mass is distributed. Call the distribution of the mass of a single interior cell its *mini-distribution.*

The tableau does not specify the shape of a mini-distribution, only that it has a specific mass distributed somehow over a specific interval. We cannot assume more than that if we wish to obtain a solution that makes no assumptions about the dependency relationship of the marginals and no assumptions about the marginals themselves beyond what is stated by their descriptions in the tableau. Therefore even the most extreme mini-distributions must be considered, in particular a spike at the low or high bound of an interior cell interval. The cumulative probability of the derived distribution will rise fastest (giving its left envelope) when all mini-distributions are spikes at the low bounds of their intervals, and will rise slowest (giving its right envelope) when all mini-distributions are spikes at the high bounds of their intervals.

Once the subset of the interior cells contributing to the maximum or minimum cumulation at a given value of the composite random variable's distribution is identified (see Figure 9), the subset's collective probability mass must be maximized or minimized consistently with the marginal constraints. This may be done with linear programming, or even by careful inspection for checking purposes. Figure 9 gives an example based on Figure 8.

*Step 3: construct the envelopes from results of calls to Step 2.* Performing Step 2 for a given horizontal-axis value of the derived distribution gives maximum and minimum values of the cumulation at that value, which then become points on the left and right envelopes respectively. The envelopes may be constructed by multiple executions of Step 2, each for a different value of the derived distribution. The set of values chosen may be the low bounds of interior cell intervals to get the left envelope and the high bounds to get the right envelope, as Statool does, if the narrowest possible envelopes are desired (Figure 10). Each envelope may

| **[1,3]** $\boldsymbol{p = 0.2}$ | **(3,4]** $\boldsymbol{p = 0.3}$ | **(4,5]** $\boldsymbol{p = 0.3}$ | **(5,7]** $\boldsymbol{p = 0.2}$ | $\leftarrow \boldsymbol{f}$ <br> $\boldsymbol{h = f * g}$ $\quad \boldsymbol{g}$ <br> $\swarrow$ $\quad \downarrow$ |
|---|---|---|---|---|
| [2,15] <br> $p_{11} =$ | (6,20] <br> $p_{12} =$ | (8,25] <br> $p_{13} =$ | (10,35] <br> $p_{14} =$ | **[2,5]** <br> $\boldsymbol{p = 0.4}$ |
| (5,21] <br> $p_{11} =$ | (15,28] <br> $p_{12} =$ | (20,35] <br> $p_{13} =$ | (25,49] <br> $p_{14} =$ | **(5,7]** <br> $\boldsymbol{p = 0.3}$ |
| (7,24] <br> $p_{31} =$ | (21,32] <br> $p_{32} =$ | (28,40] <br> $p_{33} =$ | (35,56] <br> $p_{34} =$ | **(7,8]** <br> $\boldsymbol{p = 0.2}$ |
| (8,27] <br> $p_{31} =$ | (24,36] <br> $p_{32} =$ | (32,45] <br> $p_{33} =$ | (40,63] <br> $p_{34} =$ | **(8,9]** <br> $\boldsymbol{p = 0.1}$ |

Row constraints:

$p_{11} + p_{12} + p_{13} + p_{14} = 0.4 \quad p_{21} + p_{22} + p_{23} + p_{24} = 0.3$

$p_{31} + p_{32} + p_{33} + p_{34} = 0.2 \quad p_{41} + p_{42} + p_{43} + p_{44} = 0.1$

Column constraints:

$p_{11} + p_{21} + p_{31} + p_{41} = 0.2 \quad p_{12} + p_{22} + p_{32} + p_{42} = 0.3$

$p_{13} + p_{23} + p_{33} + p_{43} = 0.3 \quad p_{14} + p_{24} + p_{34} + p_{44} = 0.2$

Correlation constraints:

(present if correlation is a problem input, and described in detail elsewhere [6])

Figure 8: A joint distribution tableau (top) and the constraints it implies (bottom). Two PDFs used as inputs and forming the marginals $\boldsymbol{f}$ and $\boldsymbol{g}$ are each discretized into 4 intervals with associated probability masses (bolded). If the dependency relationship between $\boldsymbol{f}$ and $\boldsymbol{g}$ is insufficiently specified, the joint distribution is not known, so the derived distribution $\boldsymbol{h = f * g}$ described by the interior cells (not bolded) has underdetermined probabilities $p_{ij}$.

| **[1,3]** $\boldsymbol{p = 0.2}$ | **(3,4]** $\boldsymbol{p = 0.3}$ | **(4,5]** $\boldsymbol{p = 0.3}$ | **(5,7]** $\boldsymbol{p = 0.2}$ | $\leftarrow \boldsymbol{f}$ $\quad \boldsymbol{g}$ <br> $\downarrow$ |
|---|---|---|---|---|
| **[2,15]** <br> $p_{11} = 0$ | **(6,20]** <br> $p_{12} = 0.3$ | **(8,25]** <br> $p_{13} = 0.1$ | **(10,35]** <br> $p_{14} = 0$ | **[2,5]** <br> $\boldsymbol{p = 0.4}$ |
| **(5,21]** <br> $p_{21} = 0$ | (15,28] <br> $p_{22} = 0$ | (20,35] <br> $p_{23} = 0.2$ | (25,49] <br> $p_{24} = 0.1$ | **(5,7]** <br> $\boldsymbol{p = 0.3}$ |
| **(7,24]** <br> $p_{31} = 0.2$ | (21,32] <br> $p_{32} = 0$ | (28,40] <br> $p_{33} = 0$ | (35,56] <br> $p_{34} = 0$ | **(7,8]** <br> $\boldsymbol{p = 0.2}$ |
| **(8,27]** <br> $p_{41} = 0$ | (24,36] <br> $p_{42} = 0$ | (32,45] <br> $p_{43} = 0$ | (405,63] <br> $p_{44} = 0.1$ | **(8,9]** <br> $\boldsymbol{p = 0.1}$ |

Figure 9: A joint distribution tableau and the interior cells (**larger bolded** font) whose intervals imply that their probabilities contribute to the maximum possible cumulation of derived distribution $\boldsymbol{h = f * g}$ at a value of 11, that is, to the maximum possible value of $p(h \leq 11)$ where $h$ is a sample of $\boldsymbol{h}$. An interior cell is bolded if its interval low bound is $\leq 11$, because if the mini-distributions of those cells were spikes at their interval low bounds, then their probabilities would contribute to the cumulation at 11. To compute the maximum cumulation at 11, the probabilities associated with the shaded cells must be maximized consistently with the marginal (row and column) constraints listed in Figure 8. The interior cell probabilities shown here are a solution to this maximization problem, and since their summed mass is 0.6, that is the maximum cumulation at 11. The minimum cumulation is 0, since if all mini-distributions were spikes at their interval high bounds, then none of the interior cells would contribute to the cumulation at 11 since every interior cell interval has a high bound above 11.

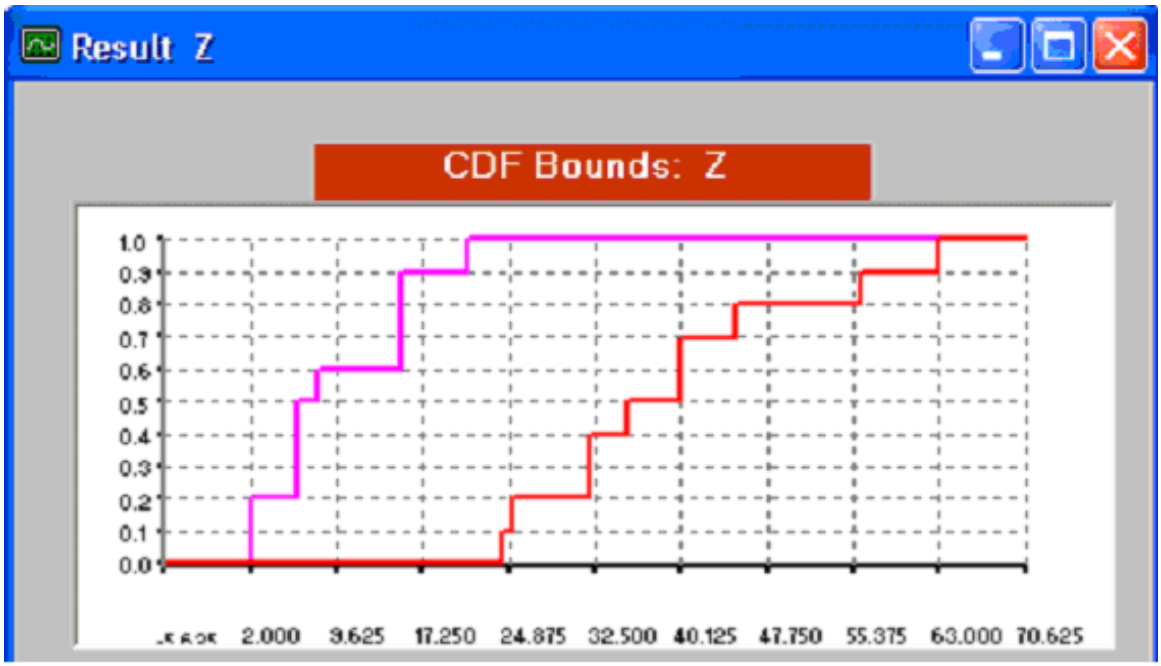


Figure 10: Envelopes around the derived distribution $\boldsymbol{h}$ described in the interior cells of the joint distribution tableau of Figure 8, under the condition of no assumption about the dependency relationship between the marginals $\boldsymbol{f}$ and $\boldsymbol{g}$.

be safely interpolated between two computed points by two line segments at right angles, with the angle pointing northwest for the left envelope and southeast for the right envelope. Simply drawing a straight line between two computed points will result in smoother curves that may work well for many, most, or perhaps nearly all potential applications but do not enclose the space of all cumulations mathematically consistent with the joint distribution tableau. Finer discretizations will typically provide envelopes that are closer to those resulting from drawing a straight line. Figure 10 shows the envelopes for the example.

## References


1. Beněs, V. and J. Štěpán, eds., *Distributions with Given Marginals and Moment Problems,* Kluwer Academic Publishers, Dordrecht, 1997.

2. Berleant, D., "Automatically verified reasoning w ith both intervals and probability density functions," *Interval Computations* (1993 No. 2), pp. 48-70.

3. Berleant, D. and H. Cheng, "A software tool for automatically verified operations on intervals and probability distributions," *Reliable Computing* **4** (1) (1998), pp. 71-82.

4. Berleant, D. and C. Goodman-Strauss, "Bounding the results of arithmetic operations on random variables of unknown dependency using intervals," *Reliable Computing* **4** (2) (1998), pp. 147-165.

5. Berleant, D. and J. Zhang, "$(a+b)^a$: cumulative credibility and the distribution envelope determination (DEnv) algorithm," poster presented at Sandia (2002 [28]), http://www.sandia.gov/epistemic/Papers/berleant.pdf.

6. Berleant, D. and J. Zhang, "Using correlation to improve envelopes around derived distributions," submitted, http://class.ee.iastate.edu/berleant/home/me/cv/papers/correlationpaper.ps.

7. Berleant, D., J. Zhang, R. Hu, and G. Sheblé, "Economic dispatch: applying the interval-based distribution envelope algorithm to an electric power problem," SIAM Workshop on Validated Computing 2002, Extended Abstracts, Toronto, May 23-25, pp. 32-35, http://www.cs.utep.edu/interval-comp/interval.02/berla.pdf.

8. Berleant, D., J. Zhang, and G. Sheblé, "On bounding the times to failure of two-component systems," submitted, http://class.ee.iastate.edu/berleant/home/me/cv/papers/2components.htm.

9. Berleant, D., J. Zhang, and G. Sheblé, "On completion times of networks of concurrent and sequential tasks," under revision, http://class.ee.iastate.edu/berleant/home/me/cv/papers/onCompletionTime.htm.

10. Colombo, A.G. and R.J. Jaarsma, "A powerful numerical method to combine random variables," *IEEE Transactions on Reliability* **R-29** (2) (1980), pp. 126-129.

11. Cuadras, C.M., http://www.bio.ub.es/estad/personal/cuadras/marg.htm, Web site for *Distributions with Given Marginals and Statistical Modeling,* Barcelona, July 17-20, 2000.

12. Dall'Aglio, G., S. Kotz, and G. Salinetti, eds., *Advances in Probability Distributions with Given Marginals,* Kluwer Academic Publishers, Dordrecht, 1991.

13. Ferson, S., *RAMAS Risk Calc 4.0: Risk Assessment with Uncertain Numbers,* Lewis Press, Boca Raton, Florida, 2002.

14. Ferson, S., "What Monte Carlo methods cannot do," *Journal of Human and Ecological Risk Assessment* 2 (4) (1996), pp. 990-1007.

15. Ferson, S. and J. Hajagos, "Don't open that envelope: solutions to the Sandia problems using probability boxes," poster presented at Sandia (2002 [28]), www.sandia.gov/epistemic/Papers/ferson.pdf.

16. Fetz, T. and M. Oberguggenberger, "Solution to challenge problem 1 in the framework of sets of probability measures," poster and manuscript presented at Sandia (2002 [28]), {fetz, Michael}@mat1.uibk.ac.at.

17. Helton, J., ed., special issue of *Reliability Engineering and System Safety* (forthcoming), synopsis at www.sandia.gov/epistemic/eup_workshop1.htm#papers.

18. Ingram, G.E., E.L. Welker, and C.R. Herrmann, "Designing for reliability based on probabilistic modeling using remote access computer systems," in *Proc. 7th Reliability and Maintainability Conference,* American Society of Mechanical Engineers, 1968, pp. 492-500.

19. Kaplan, S., "On the method of discrete probability distributions in risk and reliability calculations, applications to seismic risk assessment," *Risk Analysis* **1** (3) (1981), pp. 189-196.

20. Kazakov, D.A., Interval arithmetic for ADA, version 1.0, http://www.dmitry-kazakov.de/ada/intervals.htm.

21. Lodwick, W., algorithm presented at Validated Computing 2002, weldon.lodwick@cudenver.edu.

22. Moore, R.E., "Risk analysis without Monte Carlo methods," *Freiburger Intervall-Berichte,* 84/1, 1984, pp. 1-48.

23. Nelsen, R.B., *An Introduction to Copulas,* Lecture Notes in Statistics, Vol. 139, Springer-Verlag, 1999.

24. Oberkampf, W.L., J.C. Helton, C.A. Joslyn, S.F. Wojtkiewics, and S. Ferson, "Challenge problems: uncertainty in system response given uncertain parameters," *Reliability Engineering and System Safety,* forthcoming.

25. Red-Horse, J., "A probabilistic approach to UQ using approximate information," presented at Sandia (2002 [28]), jrredho@sandia.gov.

26. Regan, H., S. Ferson, and D. Berleant, "Equivalence of five methods for bounding uncertainty," accepted pending revision, http://class.ee.iastate.edu/berleant/home/me/cv/papers/pBoundsEquivalence.htm.

27. Rüschendorf, L., R. Schweizer, and M.D. Taylor, eds., *Distributions with Fixed Marginals and Related Topics,* Institute of Mathematical Statistics, Hayward, California, 1996.

28. Sandia National Laboratory, Epistemic Uncertainty Workshop, August 6-7, 2002, Albuquerque, www.sandia.gov/epistemic/eup_workshop1.htm.

29. Sheblé, G. and D. Berleant, "Bounding the composite value at risk for energy service company operation with DEnv, an interval-based algorithm," *SIAM Workshop on Validated Computing 2002, Extended Abstracts,* Toronto, May 23-25, pp. 166-171, http://www.cs.utep.edu/interval-comp/interval.02/sheb.pdf.

30. Springer, D., *The Algebra of Random Variables,* John Wiley and Sons, New York, 1979.

31. Statool software, http://class.ee.iastate.edu/berleant/home/Research/Pdfs/versions/statool/distribution/index.htm.

32. Tonon, F., "Using random set theory to solve challenge problem B," poster presented at Sandia (2002 [28], www.sandia.gov/epistemic/Papers/tonon.pdf.

33. Williamson, R.C. and T. Downs, "Probabilistic arithmetic i: numerical methods for calculating convolutions and dependency bounds," International Journal of Approximate Reasoning 4 (1990), pp. 89-158.