WP 94-05

November 1994

FILE COPY

Working Paper

Department of Agricultural, Resource, and Managerial Economics
Cornell University, Ithaca, New York 14853-7801 USA

Implementing the Convolutions Approach:

A Companion to "Measuring the Difference (X-Y) of Simulated Distributions: A Convolutions Approach"

by

Gregory L. Poe, Michael P. Welsh, and Eric K. Lossin

It is the policy of Cornell University actively to support equality of educational and employment opportunity. No person shall be denied admission to any educational program or activity or be denied employment on the basis of any legally prohibited discrimination involving, but not limited to, such factors as race, color, creed, religion, national or ethnic origin, sex, age or handicap. The University is committed to the maintenance of affirmative action programs which will assure the continuation of such equality of opportunity.

Implementing the Convolutions Approach: A Companion to "Measuring the Difference (X-Y) of Simulated Distributions: A Convolutions Approach"

Gregory L. Poe, Michael P. Welsh, and Eric K. Lossin*

Working Paper 94-05
Department of Agricultural, Resource, and Managerial Economics
Cornell University
(Revised: November 1994)

Abstract: This paper provides a companion piece to the Department of Agricultural, Resource, and Managerial Economics Working Paper 93-03 (revised 5-94) entitled "Measuring the Difference (X-Y) in Simulated Distributions: Application of the Convolutions Approach"and reprinted under the same title in the November issue of the *American Journal of Agricultural Economics* 76(4):, pp 904-915. Whereas WP-93-03 focused on the justification for the convolutions approach and the application of this approach to dichotomous choice contingent valuation, this paper provides a more intuitive explanation of the convolutions approach and a step- by-step discussion of how to implement this approach. Importantly, a step-by-step discussion of the GAUSS program for implementing a convolution is provided. Electronic copies of the GAUSS convolutions program via Internet are available by written request to the authors.

* The authors are respectively: Assistant Professor, Department of Agricultural, Resource, and Managerial Economics, Cornell University; Senior Project Manager, HBRS, Inc.; and former Research Assistant, Department of Agricultural Economics, University of Wisconsin -- Madison. Since the original printing of this paper, Lossin has changed his surname to Severance-Lossin. Funding for this paper was provided by the College of Agriculture and Life Sciences, Cornell University and the College of Agricultural and Life Sciences, University of Wisconsin -- Madison.

Implementing the Convolutions Approach:
A Companion to "Measuring the Difference (X-Y) of Simulated Distributions: A Convolutions Approach"

In a recent paper, Poe, Severance-Lossin, and Welsh[1] suggested that an empirical convolutions approach would offer an appropriate statistical method for determining if the difference (X-Y) in simulated or bootstrap distributions is significant. In that paper, we argued that there is a need for such a statistical test in welfare economics and other applied economics research, and that past methods of assessing the statistical difference of two distributions were inappropriate and statistically biased in many instances. An alternative approach, based on the convolution formula from statistics and mathematics, was proposed and implemented for dichotomous-choice contingent valuation (DC-CVM).

Several colleagues and an anonymous journal reviewer pointed out that the Poe, Severance-Lossin, and Welsh paper provided only a sketch of how to implement the procedure, and that it needed a companion "step-by-step recipe book" explaining the implementation of the empirical convolution formula that would be accessible for a "well-qualified research assistant". This Working Paper is intended to provide such a "how to" guide. The paper is organized into two sections. The first section briefly reviews the theoretical foundation of the empirical convolutions approach and provides a demonstration using a simple set of distributions. The second section discusses the implementation of the GAUSS program for convolutions, again with a simple demonstration. The careful reader should be able to adapt this program to other matrix based languages (e.g,. SAS-IML). We would welcome any comments on or improvements in our program.

It is important to stress that advanced concepts are discussed, and that the accessibility of this "how to" manual will be enhanced by a prior knowledge of polynomial multiplication and the GAUSS language.

---

[1] Poe, Gregory L., Eric K. Severance-Lossin, and Michael P. Welsh, 1994. "Measuring the Difference (X-Y) of Simulated Distributions: A Convolutions Approach", *American Journal of Agricultural Economics*, 76(4):904-915 (November).

I: Background on the Convolutions Approach

Let X and Y be independent random variables (with respective probability density functions $f_X(x)$ and $f_Y(y)$) and define the difference V = X-Y to be a random variable. The probability of the event V=v is defined as the union of all the possible combinations of x and y which result in a difference of v. For continuous functions this relation is given explicitly as

$$f_V(v) = \int_{-\infty}^{\infty} f_Y(x-v) f_X(x)\,dx = \int_{-\infty}^{\infty} f_X(v+y) f_Y(y)\,dy \qquad (1)$$

which is a variant of the convolution formula. Using only the far right hand side of Equation (1), the corresponding cumulative distribution function $F_V(v^\circ)$ of the difference of X and Y is

$$F_V(v^\circ) = \int_{-\infty}^{v^\circ} f_V(v)\,dv = \int_{-\infty}^{v^\circ}\int_{-\infty}^{\infty} f_X(v+y) f_Y(y)\,dy\,dv \qquad (2)$$

For empirical applications with discrete observations, the dimensions of Equation (2) can be reduced substantially. To make the problem tractable, finite width windows (Δ) must be imposed upon the continuum of values associated with X and Y. The range of the convoluted values can also be bounded by excluding the regions of the X, Y, X - Y domain in which no simulated values appear. Letting min(.) and max(.) denote minimum and maximum values, the approximate cumulative empirical distribution function associated with Equation (2) is given by

$$\hat{F}_{\hat{V}}(v^\circ) = \sum_{\min(\hat{X}-\hat{Y})}^{v^\circ} \sum_{\min(\hat{Y})}^{\max(\hat{Y})} \hat{f}_{\hat{X}}(\hat{v}+\hat{y}) \hat{f}_{\hat{Y}}(\hat{y}) \Delta y \Delta v \qquad (3)$$

where the '^' indicates that the distributions or values are a discrete approximation of a true underlying distribution or value. Equation (3) can be directly applied to the information provided from the simulated distributions to test the null hypothesis H_o: $X - Y = 0$. Adopting a 'percentile approach', the lower bound and upper bound of the 1-α confidence intervals are respectively defined as

$$\hat{L}_{1-\alpha}(\hat{V}) = \hat{F}_{\hat{V}}^{-1}(\alpha/2) \qquad \hat{U}_{1-\alpha}(\hat{V}) = \hat{F}_{\hat{V}}^{-1}(1-\alpha/2) \qquad (4)$$

and, $[\hat{L}_{1-\alpha}(\hat{V}), \hat{U}_{1-\alpha}(\hat{V})]$ is the approximate (1-α) central confidence interval for V. The null hypothesis that the difference X - Y equals zero is accepted at the α level of significance if the approximate (1-α) confidence interval of the convolution includes zero and rejected otherwise.

Using the convolutions approach, the approximate two-sided significance of the difference between distributions is determined by $2*\hat{F}_{\hat{V}}^{-1}(0)$ if $\hat{F}_{\hat{V}}^{-1}(0) \leq 0.50$, and $2*(1-\hat{F}_{\hat{V}}^{-1}(0))$ otherwise. If, for some *a priori* reason, a one-sided alternative to H_o is justified, the significance of the difference is given by $\hat{F}_{\hat{V}}^{-1}(0)$.

The complex mathematical notation belies the simplicity of this approach. Suppose that we are interested in evaluating the difference between the two simple distributions presented in Table 1.

Table 1: Hypothetical Distributions

Range	0	1	2	3	4	5
$f_X(.)$	0.00	0.00	0.10	0.40	0.40	0.10
$f_Y(.)$	0.05	0.30	0.60	0.05	0.00	0.00

The probability density function ($f_V(.)$), cumulative distribution function ($F_V(.)$), and the calculations required to generate a convolution of these distributions are demonstrated in Table 2 using only the "relevant" values that lie within the bounds set by min(Y), max(Y), and min(X-Y). Evaluating $F_V(0)$ indicates that the two distributions are different at the 17 (=2*0.085) percent level.

Table 2: Demonstration of Convolution for Simple Distributions

			$F_V(-2)$	$= 0.000$
$f_V(-1) =$	$f_x(2)f_y(3)$	$= 0.005$	$F_V(-1)$	$= 0.005$
	$(.1)(.05)$			
$f_V(0) =$	$f_x(2)f_y(2) + f_x(3)f_y(3)$	$= 0.080$	$F_V(0)$	$= 0.085$
	$(.1)(.6) + (.4)(.05)$			
$f_V(1) =$	$f_x(2)f_y(1) + f_x(3)f_y(2) + f_x(4)f_y(3)$	$= 0.290$	$F_V(1)$	$= 0.375$
	$(.1)(.3) + (.4)(.6) + (.4)(.05)$			
$f_V(2) =$	$f_x(2)f_y(0) + f_x(3)f_y(1) + f_x(4)f_y(2) + f_x(5)f_y(3)$	$= 0.370$	$F_V(2)$	$= 0.745$
	$(.1)(.05) + (.4)(.3) + (.4)(.6) + (.1)(.05)$			
$f_V(3) =$	$f_x(3)f_y(0) + f_x(4)f_y(1) + f_x(5)f_y(2)$	$= 0.200$	$F_V(3)$	$= 0.945$
	$(.4)(.05) + (.4)(.3) + (.1)(.6)$			
$f_V(4) =$	$f_x(4)f_y(0) + f_x(5)f_y(1)$	$= 0.050$	$F_V(4)$	$= 0.995$
	$(.4)(.05) + (.1)(.3)$			
$f_V(5) =$	$f_x(5)f_y(0)$	$= 0.005$	$F_V(5)$	$= 1.000$
	$(.1)(.05)$			

This method of computing convolutions is based on a polynomial multiplication approach that mirrors the computational algorithm used in the analysis (GAUSS) and is appropriate for independent samples. An alternative graphical approach may be more intuitive for some. Given the independence assumption, the joint distribution of X and Y is given by $f(X,Y)= f_X(X)f_Y(Y)$. For the two distributions in Table 2, the joint distribution is given as:

Figure 1: An Alternative Demonstration of the Convolutions Approach

F(X,Y)

Y \ X	0	1	2	3	4	5
5	0	0	0	0	0	0
4	0	0	0	0	0	0
3	0	0	0.005	0.020	0.020	0.005
2	0	0	0.060	0.240	0.240	0.060
1	0	0	0.030	0.120	0.120	0.030
0	0	0	0.005	0.020	0.020	0.005

The probability of the difference $f_v(V^o = X_i - Y_j)$ is given by the sum of probabilities that extend at a 45° angle from V^o. For example, $f_v(2)$ is given by the sum of the shaded cells in the joint distribution: i.e., $f_v(2) = f(2,0) + f(3,1) + f(4,2) + f(5,3) = 0.005 + 0.120 + 0.240 + 0.005 = 0.370$. Note that this is the exact same value as given in Table 2.

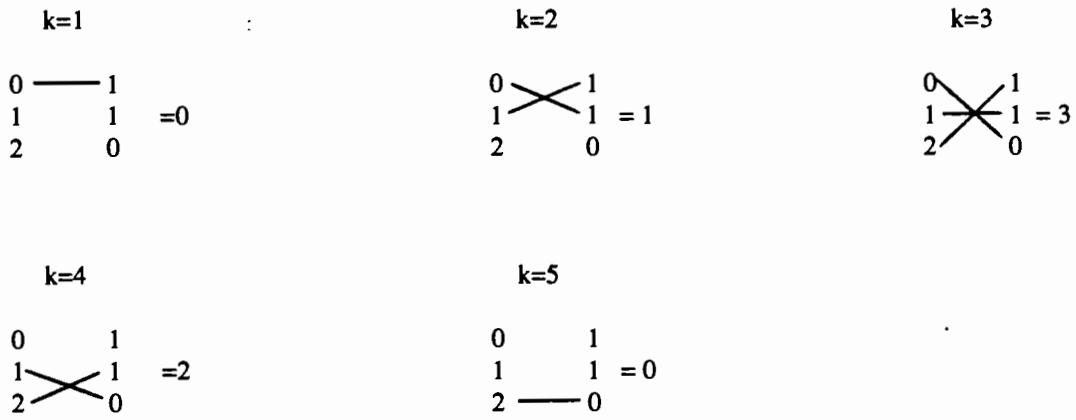
Importantly, this joint distribution provides insights as to how the convolutions approach would be modified when the independence assumption is not satisfied. In DC-CVM studies, for example, the independence assumption is typically not appropriate for questionnaires that ask individual respondents to value multiple programs. Given a correlation in responses, the X and Y responses must be paired. In essence, the bootstrap would provide an empirical approximation of the distribution V by computing $X_i - Y_j$ $\forall i=j$. Conceptually, a similar complete arithmetic approach using all possible i,j combinations would be possible for evaluating the difference of two independent distributions. However, for sample sizes typically used in simulating distributions, the convolutions approach described above offers a computational advantage over the i*j calculations required under the complete arithmetic approach. Alternatively, as a proxy to computing the exact distribution of the difference via the convolution or the complete arithmetic approach, the researcher might capture the flavor of the convolutions approach by various sampling schemes of the difference $X_i - Y_j$.

II: Programming the Empirical Convolutions Formula

II.1. Conceptual Framework:

The convolutions program used in Poe, Severance-Lossin, and Welsh centers on the convolution (CONV) routine in GAUSS. Essentially, when two vectors are used, this program is the same as a polynomial multiplication or the termwise multiplication of a power series. For instance if x={0,1,2} and y={1,1,0} are both 3X1 vectors, then the polynomial multiplication of X and Y is the 5X1 (kX1) vector {0,1,3,2,0} representing the sum of possible termwise products corresponding to the sum of the powers equalling k+1. That is, if i is the location in the first vector and j is the location in the second vector, then the polynomial multiplication would collect all the powers for which k=i+j-1. The complete set of possible combinations of the X,Y vectors is denoted by the connecting lines in Figure 2. For example, if k=2, the associated polynomial multiplication is X(1,1)*Y(2,1) + X(2,1)*Y(1,1) = 0*1 + 1*1 = 1.

Figure 2: Example of the Calculations in the Polynomial Multiplication Used in the GAUSS CONV Routine for Two Vectors



The TRICK is to get this routine to act like the subtractive convolution described in the background section. One approach is to apply the following four steps. First, the lower bounds of the convolution are determined. For the values in Table 1 and Table 2, the lower bound of the convolution (v=-1) is the sum of minimum value with a non-zero probability in the X vector (i.e. x=2) minus the maximum non-zero value in the Y vector (i.e,. y=3). Second, the vectors are organized so that Y=-Y (i.e.,

the Y vector is rotated around the origin). Following this manipulation, the distributions in Table 1 become:

Table 3: Creating a Subtractive Version of the Values in Table 1

Range	-5	-4	-3	-2	-1	0	1	2	3	4	5
$f_x(.)$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.40	0.40	0.10
$f_{-y}(.)$	0.00	0.00	0.05	0.60	0.30	0.05	0.00	0.00	0.00	0.00	0.00

The polynomial multiplication discussed at the beginning of this section is performed, which for the 9X1 vectors (depicted here as 1X9) in Table 3 results in a 21X1 vector v={0, 0, 0, 0, 0, 0, 0, 0, 0, 0.005, 0.080, 0.290, 0.370, 0.200, 0.050, 0.005, 0, 0, 0, 0, 0}. The final critical step in this process is to wade through the superfluous values and to link the probability vector resulting from the polynomial multiplication to its associated values derived if the convolution were performed manually as in Table 2 or in Figure 1. This is accomplished by calculating the minimum possible convoluted value (as above), the maximum possible convoluted value, and the range of incremental values in between. In the current example, the minimum value equals -1, the maximum value equals 5, and the increment equals 1. Using this information to link the convoluted values with their probabilities results in the following:

Table 4: The Convoluted Values and Their Corresponding Probabilities

Convoluted Value (v)	-1	0	1	2	3	4	5
$f_v(v)$	0.005	0.080	0.290	0.370	0.200	0.050	0.005
$F_v(V)$	0.005	0.085	0.375	0.745	0.945	0.995	1.000

Again, the values in the vector correspond to the previous results of the example.

II.2. Steps in the GAUSS Program:

The GAUSS program (see Appendix) for performing a convolution of two bootstrapped or simulated vectors essentially follows the four-step process outlined above. While the CONV/polynomial

multiplication outlined previously is the foundation of the program, the real challenge (and the majority of the program) is to set up the matrices for convolution, and then to line up the convoluted probabilities with their associated values. In addition, attempts to generalize the program and to make it interactive through prompts have added some bulk to what is essentially a very straightforward programming problem.

Some of the steps will be self-explanatory (at least for those who are familiar with GAUSS notation). More complex computations are further highlighted by line identification numbers. The remainder of this section describes the actual program. Interactive steps for conducting a trial convolution on two GAUSS data sets AD11235.fmt and AD22231.fmt are provided, with actual entries/responses to prompts being denoted by << >> followed by a carriage return (cr).

Step 1 - Input Matrices: The convolutions program inputs two GAUSS (*.fmt) matrices with dimensions i*c and j*c. These matrices are simply columns of raw data obtained from a simulation exercise such as bootstrapping, Monte Carlo, or Krinsky and Robb. The i and j values correspond to the number of simulated points in each bootstrapped or simulated distribution (usually 1,000 to 10,000 points). In this program i does not have to equal j, but we have set it up so that the number of columns (c) must match. For DC-CVM, these columns might correspond to the median, mean, truncated mean, etc. The default assumptions can be readily changed by altering the GAUSS commands and adding some additional programming: i.e., inputs can be changed from GAUSS to ASCII format, and the number of columns can be varied. In the first set of prompts, the user is asked to specify the desired matrices (in this example AD11235.fmt and AD22231.fmt) and the number of simulated points in each column.

Step 1: Entering the Matrices

```
*************************PROGRAM: convol-s. ****************************
*** This program takes selected columns from two simulated matrices ***
*** and convolutes them in a subtractive manner as described in the ***
*** A CONVOLUTIONS APPROACH TO MEASURING THE DIFFERENCE (X-Y) OF    ***
*** SIMULATED DISTRIBUTIONS:  APPLICATION TO DICHOTOMOUS CHOICE    ***
*** CONTINGENT VALUATION'  Gregory L. Poe, Eric K. Severance-Lossin,***
*** and Michael P. Welsh (Cornell Agricultural Econ Working Paper  ***
*** (93-03) reprinted under the same title in the November 1994    ***
*** issue of the American Journal of Agricultural Economics, 76(4): ***
*** 1994, pp 904-915.It assumes that you have two matrices w equal ***
*** number of columns, where the corresponding columns in each mat- ***
*** rix represent simulations for comparison.                      ***
***                                                               ***
*** NOTE: It is assumed that each matrix is in .fmt format and that ***
*** the output file is c:convol-s.out.  The user can change these  ***
*** default assumptions in the GAUSS code.                         ***
*** NOTE: This program was initially designed for positive distri- ***
*** butions only, but has been generalized so that it should work  ***
*** on any pair of distributions on the real line.                 ***
***************************************************************************

Name of first xxx.fmt input matrix (e.g. a:\boot1.fmt)
   <<ad11235.fmt>> cr

Rows in first input matrix
   <<1000.0000>>  cr

Name of second xxx.fmt input matrix (e.g. a:\boot2.fmt)
   <<ad22231.fmt>> cr

Rows in second input matrix
```

Next, the user will be prompted to enter the columns to be compared.

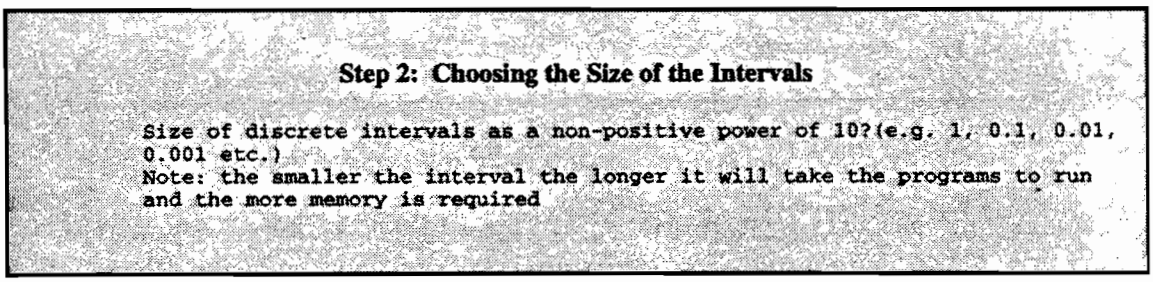
Step 1 (Continued): Selecting the Columns to be Compared

```
Column of matrices to be compared?
      <<2>> cr
```

Step 2 - Determination of Interval (Finite Windows) Size: This section prompts a selection of window size (prec), which we usually set as a power of 10. A reminder is provided that smaller windows will take longer time and memory. With the virtual memory option in recent GAUSS versions running under windows, the memory constraint is no longer as binding, and is typically not of concern.

Selecting or determining the size of the window is based on experience and "feel", much like determining the range and intervals to be considered in the graphing of bar charts. You want enough precision to be meaningful but not too precise so as to be overwhelming. Over time, each of us has developed our own *ad hoc* approach to selecting the size of windows. The common theme in our approaches is that we typically follow an iterative process of reducing window size. That is, we start with relatively large windows, and examine to see if that suffices for our purposes. If not, we then ratchet down to a smaller window size, and again re-examine the output. Footnote 3, in the Poe, Severance-Lossin, and Welsh paper outlines one technical approach to choosing window sizes.

Importantly, window size may vary with objectives of the research and the shapes of the distributions being considered. While our suggestion of developing individual rules of thumb may seem arbitrary, our experience suggests that window size selection is something that is quickly learned with some experience in applying the program. One user commented that "you just know" when the windows are too large for a particular application. In the example, 0.01 is used as the interval size, but the reader should experiment with alternative window sizes.[2]



[2] Using window sizes of 1, 0.1, 0.01, and 0.001 , the convoluted values are respectively 0.579, 0.087, 0.073, and 0.071. Associated times for running each on a GATEWAY2000 4DX2-66V (486) machine were 0.00 seconds, 0.16 seconds, 19.88 seconds, and 52.98 minutes.

Step 3 - Specification of Confidence Intervals: This section allows the user to choose the two sided central confidence intervals ($1-\alpha$) by specifying a significance level (α). Also, a title for the convolution is prompted. This will provide appropriate headings for the output.

Step 3: Specification of Confidence Intervals and Title

```
Please enter two sided signif. level for conf. int. (CI) between 1 and 20
(e.g. 5 corresponds to a 5 percent level of significance and a 95 percent
CI)

      << 5 >> cr

Please enter title

      << ad11235 - ad22231 >>  cr
```

Step 4 - Organize Bounds for the Convolution: This section sets up the range of values and calculated the probabilities associated withTables 1-4. It prints a descriptive output of the range of values needed.

4a - Reorder Matrices to Conform to a Large - Small Ordering: This step is merely one of convenience that sets the vector with the higher values as the X vector in the X-Y framework. The simple correction used here is not absolutely foolproof, and there is a second correction later in the program to help keep things straight.

4b - Create a Pseudo Data Set by Rounding to the Upper Bound of the Corresponding Interval

4.b.1 and 4.b.2: Rounds up to the upper value in a finite window. This value is used for determining the bounds and the range of the convoluted values.

4.b.3 through 4.b.6: Determines the maximum and minimum values for the convolution. Noting that the Y value is still in its positive form.

4.b.7 through 4.b.9: Calculates the lowest possible value, the highest possible value, and the range of the convolution.

4c - Print Out Bounds and Precision of the Convolution: The output at this point will look like:

```
************************************************************
                    ad11235 - ad22231
************************************************************
Convolution evaluated is ad11235.fmt minus ad22231.fmt

Minimum Value of Distribution 1          27.7800
Minimum Value of Distribution 2          26.9800
Maximum Value of Distribution 1          30.4100
Maximum Value of Distribution 2          29.4300

Level of precision of the convolution     0.0100
Lowest Possible Convoluted Value         -1.6500
Highest Possible Convoluted Value         3.4300
Difference                                5.0800
```

The maximum and minimum values define the range of the distribution. The finite window size is provided for reference.

Step 5 - Set Up Matrices for Convolution:

5.1: Determines the maximum absolute value found in the two matrices.

5.2: Uses the maximum to create the dimensions of the problem, which equals 2*MAXCVAL + 1. In other words it is equivalent to an incremental vector centered on zero that extends symmetrically to the maximum value in both the positive and negative directions.

5.3: Creates an incremental vector corresponding to the range determined by combining 5.1 and 5.2.

5.4 and 5.5: Creates frequencies for each increment or finite window that has been defined.

5.6: Rotates the second distribution around zero, which now corresponds to $f_{-y}(.)$ in the example.

5.7: Merges the distributions. This corresponds to the values in Table 3.

5.8: Identifies the lowest point in which a non-zero probability is observed. In Table 3, this would be -3.

5.9: Eliminates the unneeded series of zeros below the value determined in 5.8. This truncation is not necessary for the convolution. However, in cases in which the maximum values of the

distributions are far apart, it may eliminate many unnecessary calculations. In the example, this truncation would make Table 3 begin at -3. Note that the minimum probability value corresponds to a probability associated with a bootstrap distribution of 10,000 points or less. If a larger bootstrap is used, the sensitivity values in the indices used here and later in the program will have to be altered.

Step 6 - Conduct and Report the Convolution: Given all the data manipulation thus far, this is the heart of the program.

6.1: Performs the convolution/polynomial multiplication. The trailing zeros in the GAUSS CONV routine are pointers indicating that the entire vectors/distributions are being compared in the convolution.

Step 6a - Place Upper and Lower Limits on the Convolution:

6.a.1: Establishes a minimum cut-off point for the lower bound of the convolution based on the earliest observed non-zero probability.

6.a.2: Convdon2 is the truncated matrix in which all zero probabilities observed preceding the first non-zero observation in the probability vector from the convolution are truncated.

6.a.3: Identifies the upper bound of the probability distribution where the cumulative probabilities equal 1.

6.a.4: Creates a distribution corresponding to the values in Table 4.

Step 6b - Identify and Report Two-Sided Value of Convolution at Zero

6.b.1. Provides a skip if the convolution does not include zero. This is needed because the indexing in the following steps would not work if the truncated vector does not include zero.

6.b.2. Identifies the index (zv) for which the value equals zero.

6.b.3. Using the index above, finds the corresponding probability and multiplies by two to provide a two-sided confidence interval.

Step 6c - Identify and Report Upper and Lower Confidence Bounds This section takes the provided confidence bounds, translates them into values, and reports the values. In addition, the section provides a print out of whether or not the value includes zero.

The outputs from steps 6b and 6c would look like:

```
alpha (significance)  0.07272600

lower bound 95 percent CI  -0.09000000
upper bound 95 percent CI   1.97000000

Designated 95 Confidence Interval Includes Zero
```

Step 7 - Saving Options for Graphing, Etc: This section allows the user to save the input and output vectors to ASCII format for graphing or other purposes. Self explanatory.

II.3. Interpreting the Output

The interpretation of this output is that the 95 percent confidence interval ranges from -0.09 to 1.97 and thus includes zero. As such, the null hypothesis that X-Y=0 cannot be rejected at the 5 percent level of significance. The actual level of significance is about 7.3 percent in this case.

Appendix

```
/*PROGRAM convol-s.prg*/

#lineson;
cls;
format /m1 /rd 10,5;
output reset;
output file = c:conv--s.out;  @can change output@
output on;

*************************PROGRAM: conv--s.  ****************************;
**** This program takes selected columns from two simulated matrices ****;
**** and convolutes them in a subtractive manner as described in the ****;
**** A CONVOLUTIONS APPROACH TO MEASURING THE DIFFERENCES OF         ****;
**** SIMULATED DISTRIBUTIONS:  APPLICATION TO DICHOTOMOUS CHOICE     ****;
**** CONTINGENT VALUATION' by Gregory L. Poe, Eric K. Severance-     ****;
**** Lossin, and Michael P. Welsh (Cornell AE Working Paper          ****;
**** (93-03)reprinted under the same title in the November 1994      ****;
**** issue of the American Journal of Agricultural Economics 76(4)   ****;
**** 904-915.It assumes that you have two matrices with equal        ****;
**** number of columns, where the corresponding columns in each mat- ****;
**** rix represent simulations for comparison.                       ****;
****                                                                 ****;
**** NOTE: It is assumed that each matrix is in .fmt format and that ****;
**** the output file is a:convol-s.out.  The user can change these   ****;
**** default assumptions in the GAUSS code.                          ****;
*************************************************************************;

print;
beggar:
closeall f1; closeall f2;
format /m1 /rd 10,4;

/*************************************************************************/
/********                          STEP 1                          *******/
/*************************************************************************/
/* This section allows the user to input matrices and select vectors  */
/*************************************************************************/

"Name of first xxx.fmt input matrix (e.g. a:\\boot1.fmt)"; name1=cons;
      open f1 = ^name1;
      "    " name1;
      print;

"Rows in first input matrix"; nr1 = con(1,1);
      if nr1==0; goto bye; endif;
      value11 = readr(f1,nr1);
      nr1;
      print;

"Name of second xxx.fmt input matrix (e.g. a:\\boot2.fmt)"; name2=cons;
      open f2 = ^name2;
      "     " name2;
      print;

"Rows in second input matrix"; nr2 = con(1,1);
      if nr2==0; goto bye; endif;
      value22 = readr(f2,nr2);
      nr2;
      print;

   if cols(value11) ne cols(value22);
     "Number of Cols does not match. Re-enter as prompted or CTRL C to escape";
      print;
      clear value11, value22;
      goto beggar;
    endif;

"Column of matrices to be compared?"; nc=con(1,1);
      if nc==0; goto bye; endif;
      nc;
      print;
```

```
/*************************************************************************/
/********                      STEP 2                             *****/
/*************************************************************************/
/********          Determination of Intervals                     *****/
/*************************************************************************/

inter:
"Size of discrete intervals as a non-positive power of 10?";
"(e.g. 1, 0.1, 0.01, 0.001 etc.)";

@auth. note: you may use other 0 to 1 ints but we generally have not found a
  need to do so @

"Note: the smaller the interval the longer it will take the programs to"
"      run and the more memory is required";

    prec=con(1,1); iprec=1/prec;
    if prec LE 0 or prec GT 1;
      "Precision out of bounds.  Re-enter as prompted or CTRL C to escape";
      print;
      goto inter;
    endif;

/*************************************************************************/
/********                      STEP 3                             *****/
/*************************************************************************/
/********        Specification of Confidence Intervals            *****/
/*************************************************************************/

"Please enter two sided signif. level for conf. int. (CI) between 1 and 20";
"(e.g. 5 corresponds to a 5 percent level of significance and a 95 percent CI)";
    sigl=con(1,1);
    sigperc=sigl/100;
    print;

"Please enter title            ";;title=cons;
cls;
print;

/*************************************************************************/
/********                      STEP 4                             *****/
/*************************************************************************/
/**************organize bounds for the convolution********************/
/*************************************************************************/

/*************************************************************************/
/*  Step 4a: reorder matrices to conform to a large - small ordering  */
/*************************************************************************/
if maxc(value11) ge maxc(value22);
   vh=value11;
   vl=value22;
   vhname=name1;
   vlname=name2;
  endif;

if maxc(value11) lt maxc(value22);
   vh=value22;
   vl=value11;
   vhname=name2;
   vlname=name1;
  endif;

/*************************************************************************/
/*Step 4b:Create a psuedo data set by rounding up to upperbound of int*/
/*************************************************************************/

4.b.1  ceilvh = (ceil(vh[.,nc]*iprec))/iprec;
4.b.2  ceilvl = (ceil(vl[.,nc]*iprec))/iprec;

4.b.3  mincvh =  minc(ceilvh);
4.b.4  mincvl =  minc(ceilvl);
4.b.5  maxcvh =  maxc(ceilvh);
```

```
4.b.6   maxcvl =  maxc(ceilvl);

4.b.7   lowcval  = mincvh - maxcvl;
4.b.8   higcval = maxcvh - mincvl;
4.b.9   diffc    = higcval - lowcval;

        /*************************************************************************/
        /*Step 4c: Print out bounds and precision of the convolution***********/
        /*************************************************************************/

        ************************************************************************;
        *                    " title;
        ************************************************************************;

        "Convolution evaluated is " vhname " minus " vlname;
        print;
        "Minimum Value of Distribution 1       ";; mincvh;
        "Minimum Value of Distribution 2       ";; mincvl;
        "Maximum Value of Distribution 1       ";; maxcvh;
        "Maximum Value of Distribution 2       ";; maxcvl;
        print;
        "Level of precision of the convolution ";; prec;
        "Lowest Possible Convoluted Value      ";; lowcval;
        "Highest Possible Convoluted Value     ";; higcval;
        "Difference                            ";; diffc;
        print;

        /*************************************************************************/
        /********                          STEP 5                          *****/
        /*************************************************************************/
        /*************Set up Matrices for Convolutions   ***********************/
        /*************************************************************************/

        rh = rows(vh);
        rl = rows(vl);

5.1     maxcval=ceil(maxc(abs(vh[.,nc])|abs(vl[.,nc])));
5.2     dimen=(2*maxcval/prec)+1;
5.3     conval=-maxcval - prec + (prec*cumsumc(ones(dimen,1)));

        format /m1 /rd 12,8;
5.4     dist1 = (counts(vh[.,nc],conval))/rh;
5.5     dist2 = (counts(vl[.,nc],conval))/rl;
5.6        rdist2=rev(dist2);

5.7     fulldist = conval~dist1~rdist2;
5.8     minind=minc(indnv(1,cumsumc(dist1).>0.000001)|
5.9             indnv(1,cumsumc(rdist2).>0.000001));
        dist = fulldist[minind:rows(fulldist),.];

        /*************************************************************************/
        /********                          STEP 6                          *****/
        /*************************************************************************/
        /****************conduct and report convolution***************************/
        /*************************************************************************/


6.1     convdone=conv(dist[.,2],dist[.,3],0,0);

        /*************************************************************************/
        /*Step 6a: Place upper and lower limits on the convolution ************/
        /*************************************************************************/

6.a.1   cutoff1 = indnv(1,(convdone.>0.00000000001));
             print;
6.a.2   convdon2= convdone[cutoff1:rows(convdone),1];
        rw=rows(convdon2);

        outval=lowcval - prec + (prec*cumsumc(ones(rw,1)));

        format /m1 /rd 12,8;
        sumdist=cumsumc(convdon2);
6.a.3   cutoff2 = indnv(1,(sumdist.>0.9999999));
```

```
6.a.4   outval=outval~convdon2~sumdist;
        outval=outval[1:cutoff2,.];
        ov1=outval[.,1];
        ov3=outval[.,3];

        @ if you want to see the convouluted matrix across the screen,
        then delete comments here @    .

        /*
        "OUTVAL:";
        outval; print;
        */


        /*************************************************************************/
        /*Step 6b: Identify and report two sided value of convolution at zero */
        /*         Note the skip pattern if the convolution does not incl zero*/
        /*************************************************************************/

6.b.1   if lowcval ge 0;
           "Convolution Does Not Include Zero"; print;
            goto CICALC;
        endif;

6.b.2   zv=indnv(1,(ov1.<0.000001 .and ov1.>-0.0000001).==1.0000000);
          if outval[zv,3] le 0.50;
6.b.3     zval=2*outval[zv,3];
          endif;
        @in case the ordering by maxvals did not mean order@
          if outval[zv,3] gt 0.50;
            zv=2*(1-outval[zv,3]);


           endif;

        "alpha (two sided significance)";; zval;print;

        /*************************************************************************/
        /*Step 6c: Identify and report upper and lower confidence bounds      */
        /*         Report if designated confidence intervals include zero     */
        /*************************************************************************/
        CICALC:
        lb=indnv(1,minc(abs(ov3-(sigperc/2))).==abs(ov3-(sigperc/2)));
           lbval=outval[lb,1];
        ub=indnv(1,minc(abs(ov3-(1-sigperc/2))).==abs(ov3-(1-sigperc/2)));
           ubval=outval[ub,1];

        "lower bound" 1-sigperc " percent CI";; lbval;
        "upper bound" 1-sigperc " percent CI";; ubval;
        print;

        if (lb lt 0) and (ub gt 0);
           "Designated " 1-sigperc "Confidence Interval Includes Zero";
           else;
           "Designated " 1-sigperc "Confidence Interval Does Not Include Zero";
          endif;
```

```
/*************************************************************************/
/********                        STEP 7                           *******/
/*************************************************************************/
/****************Saving Options for Graphing Etc.*************************/
/*************************************************************************/

"Do you want this to be outputed to an ascii file? yes=1 no=0"; asc=con(1,1);
  if asc==0; goto bye; endif;
     print;
     "Name of saved ascii file (e.g. wtpwta.asc)"; name3 = cons;
     print;
     output file = ^name3 reset;
     screen off;
     print outval;
     output off;
     screen on;

"Do you want to output the input files to an ascii file? yes=1 no=0";
     asc2=con(1,1);
     if asc2==0; goto bye; endif;
     print;
     "Name of saved ascii file (e.g. dis1dis2.asc)"; name4=cons;
     print;
     output file = ^name4 reset;
     screen off;
     print dist;
     output off;
     screen on;

bye:
"Do you wish to do another? 1=yes 0=no"; try=con(1,1);
if try==1; goto beggar; endif;
```

OTHER A.R.M.E. WORKING PAPERS

No. 93-08	Exchange Rate Reform and Its Effects on Ecuador's Traditional Agricultural Export Sector	Xavier Bejarano David R. Lee Duty Greene
No. 93-09	Processed Sweet Potato: Responding to Kenya's Urban Food Needs	Njeri Gakonyo
No. 93-10	Estimating Consumer Energy Demand Using International Data: Theoretical and Policy Implications	Dale S. Rothman Jong Ho Hong Timothy D. Mount
No. 93-11	Monitored Retrievable Storage of Spent Nuclear Fuel in Indian Country: Liability, Sovereignty, and Socioeconomics	Jon D. Erickson Duane Chapman Ronald E. Johnny
No. 93-12	Urban Influences on Farmland Use in New York State	Thomas A. Hirschl Nelson L. Bills
No. 93-13	An Empirical Evaluation of Incremental Conditional Damages and Benefits	Gregory L. Poe Richard Bishop
No. 94-01	Income and Dietary Change: International Comparisons Using Purchasing-Power-Parity Conversions	Thomas T. Poleman Lillian Thomas
No. 94-02	Meeting Clean Air Standards for Acid Rain and Urban Ozone: Implications for Electric Utilities in New York State	Martha Czerwinski Brian Minsker Timothy Mount
No. 94-03	Economic Evaluation of Residential Solar Electricity in Developing Countries	Duane Chapman
No. 94-04	Fishing in Stochastic Waters	Gunter Schamell John M. Conrad