

# 2Rsim

## FILE OPERATIONS

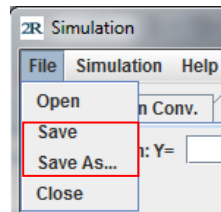
2R Sim is capable of reading and writing **.2r** files, which contain the complete description of a specific simulation model:

1. The model's equation.
2. All the variables and their corresponding probability distribution where appropriate, along with the pertinent parameters.
3. The correlation matrix.

These files are the means for 2R Sim users to save their work, as well as the medium of distribution of models that could be of interest to other 2R Sim users.

## SAVING MODELS

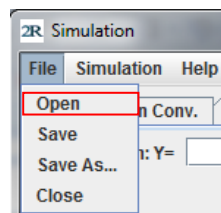
Even if a model isn't yet complete, a user can decide to save its information for later use. In order to do this, the user must navigate through the **File** menu and select the **Save** or **Save As...** option:



The difference between **Save** and **Save As...** is that, while **Save** will only ask for the file's name and destination once and will then overwrite that same file on any subsequent uses, **Save As...** will ask for the file's name and destination every time it is invoked. Thus, **Save As...** is to be used whenever a user wants to save modifications made to a file without modifying the base file.

## OPENING MODELS

In order to load the information contained inside a **.2r** file, the user must navigate through the **File** menu and select the **Open** option:

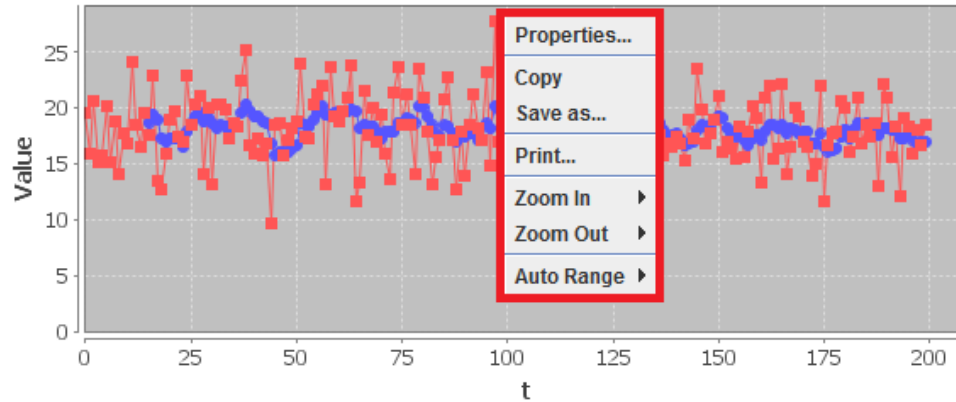


If no errors occur, the loaded model is shown in the **Main** tab (equation, variables, and correlation matrix).

GRAPHS

All of the graphs generated in 2R Soft provide a wide array of options in the form of a context menu. **The context menu appears when you right-click over a graph:**

**S1 Series**



PROPERTIES PANE

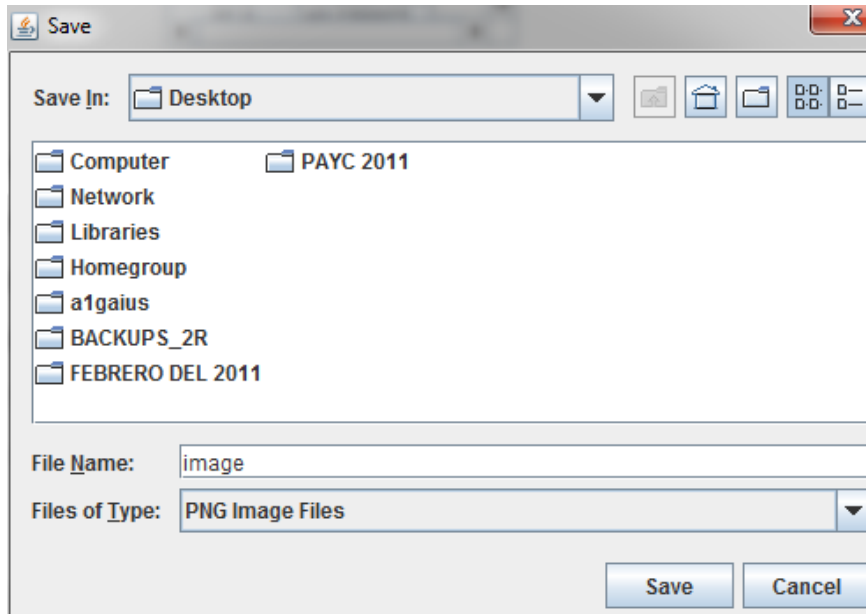
If you select the **Properties...** option, a properties pane appears. The properties pane lets you change the graph title, axis names, axis ranges, and font size.

---

## COPY AND SAVE AS...

If you select **COPY**, the graph is copied to the system clipboard, so you can **PASTE** it anywhere else (Microsoft Word, Microsoft PowerPoint, etc).

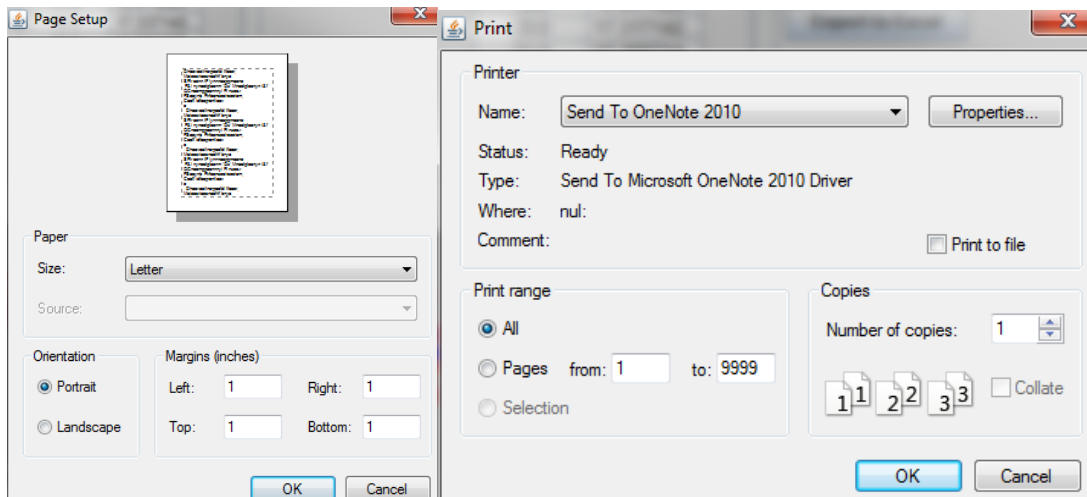
Meanwhile, if **Save As...** is selected, 2R Soft will save the graph as a **PNG** image file in your hard disk after selecting the desired output folder and file name:



---

## PRINT

The **Print** option does just that: it sends the graph to the printer of your choice (local or networked):



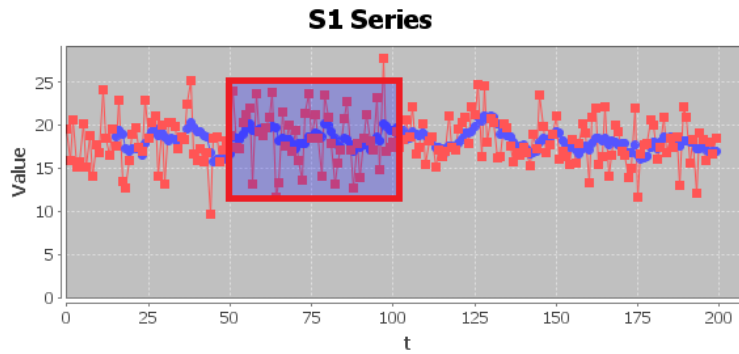
---

## SCALE OPTIONS

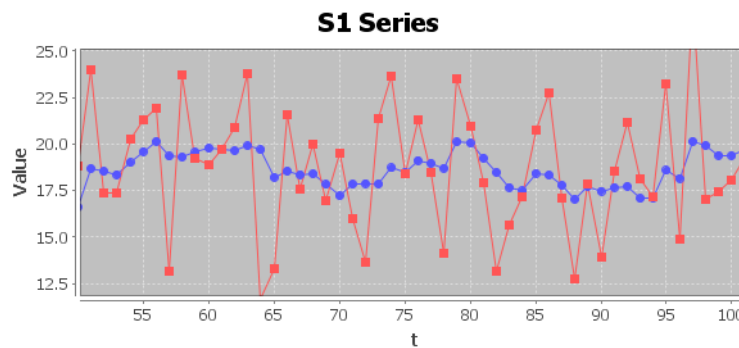
The **Auto Range**, **Zoom In**, and **Zoom Out** options are a quick way to inspect the graph. If a very specific range is needed for an axis, we highly recommend the [Properties Pane](#).

## MANUAL ZOOM IN

For user convenience, all **2R Soft graphs support manual zoom in by regions**. If you're interested in a specific region, **hold your left-click and drag the mouse** to generate a highlighted box around that region:



The end result:



## EQUATION EDITOR

User-entered equations are common in 2R Soft. This section of the document explains the use of the equation editor.

## FUNCTIONS AND OPERATIONS

Equations can contain the following functions and operations:

Function/Op.	Description	Usage (A and B are declared variables or numeric values)
+	Addition.	A+B Example: 4+7=11
-	Subtraction.	A-B Example: 4-7=-3
*	Multiplication.	A*B Example: 4*7=28
/	Division.	A/B Example: 6/4=1.5
^	Returns the value of the first operand raised to the power of the second operand. (Microsystems)	A^B Example: 5^3=125 Example: 5^(-3)=1/125
%	Modulo operation. Divides the value of one expression by the value of another, and returns the remainder. (MSDN)	A%B Example: 7%3=1 Example: 67%10=7
cos	Returns the trigonometric cosine of an angle. (Microsystems)	cos(A), where A is in radians Example: cos(3.14) ≈ 1.0

<b>sin</b>	Returns the trigonometric sine of an angle. (Microsystems)	sin(A), where A is in radians Example: sin(3.14) ≈ 0.0
<b>tan</b>	Returns the trigonometric tangent of an angle. (Microsystems)	tan(A), where A is in radians Example: tan(3.14) ≈ 0.0
<b>acos</b>	Returns the arc cosine of an angle, in the range of 0.0 through pi. (Microsystems)	acos(A), where A is the value whose arc cosine is to be returned. Example: acos(1)=0.0
<b>asin</b>	Returns the arc sine of an angle, in the range of -pi/2 through pi/2 (Microsystems)	asin(A), where A is the value whose arc sine is to be returned. Example: asin(0)=0.0
<b>atan</b>	Returns the arc tangent of an angle, in the range of -pi/2 through pi/2. (Microsystems)	atan(A), where A is the value whose arc tangent is to be returned. Example: atan(0)=0.0
<b>sqrt</b>	Returns the correctly rounded positive square root of a positive real number. (Microsystems)	sqrt(A), where A is a real positive number. Example: sqrt(9)=3
<b>sqr</b>	Returns the value of the argument squared (to the power of 2).	sqr(A) Example: sqr(-4)=16 Example: sqr(2)=4
<b>ln</b>	Returns the natural logarithm (base e) of a real value. (Microsystems)	ln(A), where A is a positive real number greater than zero. Example: ln(1)=0 Example: ln(e^2)=2
<b>min</b>	Returns the smaller of two real values. That is, the result is the value closer to negative infinity. (Microsystems)	min(A,B) Example: min(4,9)=4 Example: min(-4,-11)=-11
<b>max</b>	Returns the greater of two real values. That is, the result is the argument closer to positive infinity. If the arguments have the same value, the result is that same value. (Microsystems)	max(A,B) Example: max(4,9)=9 Example: max(-4,-11)=-4
<b>ceil</b>	Returns the smallest (closest to negative infinity) real value that is not less than the argument and is equal to a mathematical integer. (Microsystems)	ceil(A) Example: ceil(9.23)=10 Example: ceil(-1.25)=-1
<b>floor</b>	Returns the largest (closest to positive infinity) value that is not greater than the argument and is equal to a mathematical integer (Microsystems)	floor(A) Example: floor(9.23)=9 Example: floor(-1.25)=-2
<b>abs</b>	Returns the absolute value of a real value. If the argument is not negative, the argument is returned. If the argument is negative, the negation of the argument is returned. (Microsystems)	abs(A) Example: abs(4.15)=4.15 Example: abs(-4.3)=4.3 Example: abs(0)=0
<b>neg</b>	Changes the sign of the value received as an argument (negative to positive or positive to negative).	neg(A) Example: neg(-1)=1 Example: neg(1)=-1 Example: neg(0)=0
<b>rnd</b>	Returns a pseudo-random real value between 0.0 (included) and the value received as an argument (excluded).	rnd(A) Example: rnd(50)=0,1.45,2.78,49.9 Example: rnd(-20)=0,-1.45,-18.392,-19.61
<b>exp</b>	Returns Euler's number e raised to the power of a real value. (Microsystems)	exp(A) Example: exp(0)=1 Example: exp(1)=e Example: exp(-2)=1/(e^2)
<b>log</b>	Returns the logarithm (base 10) of a real value. (Microsystems)	log(A), where A is a positive real number greater than zero. Example: log(1)=0 Example: log(10^2)=2

## PROBABILITY DISTRIBUTION TYPES

When declaring a variable with a known distribution, the user can select one of many types of probability distributions.

Distribution	Description	Parameters
<b>Beta</b>	<p>The <i>beta</i> distribution has shape parameters <math>\alpha &gt; 0</math> and <math>\beta &gt; 0</math> over the interval <math>(a, b)</math>, where <math>a &lt; b</math>.</p> <p>It has density:  <math>f(x) = (x - a)^{\alpha-1} (b - x)^{\beta-1} / [B(\alpha, \beta)(b - a)^{\alpha+\beta-1}]</math>                      for <math>a &lt; x &lt; b</math>, and 0 elsewhere.</p> <p>It has the following distribution function:  <math>F(x) = I_{a, \theta}(x) = \int_a^x (\xi - a)^{\alpha-1} (b - \xi)^{\beta-1} / [B(\alpha, \beta)(b - a)^{\alpha+\beta-1}] d\xi</math>, for <math>a &lt; x &lt; b</math></p> <p>(Simard)</p>	<p>Alpha – shape parameter, <math>\alpha &gt; 0</math>                      Beta – shape parameter, <math>\beta &gt; 0</math>                      a – lower bound of the interval                      b – upper bound of the interval, <math>b &gt; a</math></p>
<b>Binomial</b>	<p>The binomial distribution with parameters <math>n</math> and <math>p</math>, where <math>n</math> is a positive integer and <math>0 \leq p \leq 1</math>. Its mass function is given by:  <math>p(x) = nCr(n, x)p^x(1 - p)^{n-x} = n!/[x!(n - x)!] p^x(1 - p)^{n-x}</math> for <math>x = 0, 1, 2, \dots, n</math>,</p> <p>and its distribution function is:  <math>F(x) = \sum_{j=0}^x nCr(n, j) p^j(1 - p)^{n-j}</math> for <math>x = 0, 1, 2, \dots, n</math>,                      where <math>nCr(n, x)</math> is the number of possible combinations of <math>x</math> elements chosen among a set of <math>n</math> elements.</p> <p>(Simard)</p>	<p><math>p</math> – probability of success on each trial (<math>0 \leq p \leq 1</math>)  <math>n</math> – number of trials (integer), <math>n &gt; 0</math></p>
<b>Chi Square</b>	<p>The <i>chi-square</i> distribution with <math>n</math> degrees of freedom, where <math>n</math> is a positive integer. Its density is:  <math>f(x) = x^{(n/2)-1} e^{-x/2} / (2^{n/2} \Gamma(n/2))</math>, for <math>x &gt; 0</math>                      where <math>\Gamma(x)</math> is the gamma function. The <i>chi-square</i> distribution is a special case of the <i>gamma</i> distribution with shape parameter <math>n/2</math> and scale parameter <math>1/2</math>.</p> <p>(Simard)</p>	<p><math>n</math> – degrees of freedom (integer), <math>n &gt; 0</math></p>
<b>Deterministic</b>	<p>Distribution that represents a constant value, <i>val</i>. Consequently:  <math>f(x) = \begin{cases} 1 &amp; \text{if } x = \text{val} \\ 0 &amp; \text{if } x \neq \text{val} \end{cases}</math>, <math>F(x) = \begin{cases} 1 &amp; \text{if } x \geq \text{val} \\ 0 &amp; \text{if } x &lt; \text{val} \end{cases}</math></p>	<p>Value – any real number</p>
<b>Discrete Uniform</b>	<p>The <i>discrete uniform</i> distribution over the integers in the range <math>[i, j]</math>. Its mass function is given by:  <math>p(x) = 1/(j - i + 1)</math> for <math>x = i, i + 1, \dots, j</math>                      and 0 elsewhere.</p> <p>The distribution function is:  <math>F(x) = (\text{floor}(x) - i + 1) / (j - i + 1)</math> for <math>i \leq x \leq j</math>                      and its inverse is:  <math>F^{-1}(u) = i + (j - i + 1)u</math> for <math>0 \leq u \leq 1</math>.</p> <p>(Simard)</p>	<p>Min. – lower bound (integer)                      Max. – upper bound (integer)                      (Max. &gt; Min.)</p>
<b>Exponential</b>	<p>The <i>exponential</i> distribution with mean <math>1/\lambda</math> where <math>\lambda &gt; 0</math>. Its density is:  <math>f(x) = \lambda e^{-\lambda x}</math> for <math>x \geq 0</math>,                      its distribution function is:  <math>F(x) = 1 - e^{-\lambda x}</math>, for <math>x \geq 0</math>,                      and its inverse distribution function is:  <math>F^{-1}(u) = -\ln(1 - u)/\lambda</math>, for <math>0 &lt; u &lt; 1</math></p> <p>(Simard)</p>	<p>Lambda – rate parameter, <math>\lambda &gt; 0</math></p>
<b>F-Distribution</b>	<p>The Fisher F distribution with <math>n</math> and <math>m</math> degrees of freedom, where <math>n</math> and <math>m</math> are positive integers. Its density is:  <math>f(x) = \Gamma((n + m)/2) n^{n/2} m^{m/2} / [\Gamma(n/2) \Gamma(m/2)] x^{(n-2)/2} / (m + nx)^{(n+m)/2}</math>, for <math>x &gt; 0</math>.                      where <math>\Gamma(x)</math> is the gamma function</p> <p>(Simard)</p>	<p>D.O.F. 1 – the <math>n</math> degrees of freedom (integer), D.O.F. 1 &gt; 0                      D.O.F. 2 – the <math>m</math> degrees of freedom (integer), D.O.F. 2 &gt; 0</p>

<b>Gamma</b>	<p>The <i>gamma</i> distribution with shape parameter <math>\alpha &gt; 0</math> and scale parameter <math>\lambda &gt; 0</math>. The density is:  <math>f(x) = \lambda^\alpha x^{\alpha-1} e^{-\lambda x} / \Gamma(\alpha)</math>, for <math>x &gt; 0</math>,  where <math>\Gamma</math> is the gamma function, defined by:  <math>\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx</math>.</p> <p>In particular, <math>\Gamma(n) = (n - 1)!</math> when <math>n</math> is a positive integer.</p> <p>(Simard)</p>	<p>Alpha – shape parameter, alpha &gt; 0  Lambda – scale parameter, lambda &gt; 0</p>
<b>Geometric</b>	<p>The <i>geometric</i> distribution with parameter <math>p</math>, where <math>0 &lt; p &lt; 1</math>. Its mass function is:  <math>p(x) = p(1 - p)^x</math>, for <math>x = 0, 1, 2, \dots</math>  The distribution function is given by:  <math>F(x) = 1 - (1 - p)^{x+1}</math>, for <math>x = 0, 1, 2, \dots</math>  and its inverse is:  <math>F^{-1}(u) = \text{floor}(\ln(1 - u) / \ln(1 - p))</math>, for <math>0 \leq u &lt; 1</math></p> <p>(Simard)</p>	<p><math>p</math> – probability of success on each trial  (<math>0 &lt; p &lt; 1</math>)</p>
<b>Gumbel</b>	<p>The Gumbel distribution, with location parameter <math>\delta</math> and scale parameter <math>\theta \neq 0</math>. Using the notation <math>z = (x - \delta) / \theta</math>, it has density:  <math>f(x) = e^{-z} e^{-e^{-z}} /  \theta </math>, for <math>-\infty &lt; x &lt; \infty</math>.  and distribution function:  <math>F(x) = e^{-e^{-z}}</math>, for <math>\theta &gt; 0</math>  <math>F(x) = 1 - e^{-e^{-z}}</math>, for <math>\theta &lt; 0</math></p> <p>(Simard)</p>	<p>Beta – scale parameter, beta <math>\neq 0</math>  Delta – location parameter, any real number</p>
<b>Hypergeometric</b>	<p>The <i>hypergeometric</i> distribution with <math>k</math> elements chosen among <math>l</math>, <math>m</math> being of one type, and <math>l - m</math> of the other. The parameters <math>m</math>, <math>k</math> and <math>l</math> are positive integers where <math>1 \leq m \leq l</math> and <math>1 \leq k \leq l</math>. Its mass function is given by:</p> $p(x) = \frac{nCr(m, x)nCr(l - m, k - x)}{nCr(l, k)}$ <p>for <math>\max(0, k - l + m) \leq x \leq \min(k, m)</math>  where <math>nCr(n, x)</math> is the number of possible combinations of <math>x</math> elements chosen among a set of <math>n</math> elements.</p> <p>(Simard)</p>	<p><math>m</math> – number of elements of one type (integer), <math>m &gt; 0</math>  <math>l</math> – total elements (integer), <math>l &gt; 0</math>  <math>k</math> – number of elements chosen among <math>l</math> (integer), <math>k &gt; 0</math></p>
<b>Logistic</b>	<p>The <i>logistic</i> distribution. It has location parameter <math>\alpha</math> and scale parameter <math>\lambda &gt; 0</math>. The density is:  <math>f(x) = (\lambda e^{-\lambda(x-\alpha)}) / ((1 + e^{-\lambda(x-\alpha)})^2)</math> for <math>-\infty &lt; x &lt; \infty</math>.  and the distribution function is:  <math>F(x) = 1 / [1 + e^{-\lambda(x-\alpha)}]</math> for <math>-\infty &lt; x &lt; \infty</math>.</p> <p>For <math>\lambda = 1</math> and <math>\alpha = 0</math>, one can write:  <math>F(x) = (1 + \tanh(x/2)) / 2</math>.</p> <p>The inverse distribution function is given by:  <math>F^{-1}(u) = \ln(u / (1 - u)) / \lambda + \alpha</math> for <math>0 \leq u &lt; 1</math></p> <p>(Simard)</p>	<p>Alpha – location parameter, any real number  Lambda – scale parameter, lambda &gt; 0</p>
<b>Lognormal</b>	<p>The <i>lognormal</i> distribution. It has scale parameter <math>\mu</math> and shape parameter <math>\sigma &gt; 0</math>. The density is:  <math>f(x) = ((2\pi)^{-1/2} \sigma^{-1}) e^{-(\ln(x)-\mu)^2 / (2\sigma^2)}</math> for <math>x &gt; 0</math>, and 0 elsewhere.</p> <p>The distribution function is:  <math>F(x) = \Phi((\ln(x)-\mu) / \sigma)</math> for <math>x &gt; 0</math>,  where <math>\Phi</math> is the standard normal distribution function.</p> <p>Its inverse is given by:  <math>F^{-1}(u) = e^{\mu + \sigma \Phi^{-1}(u)}</math> for <math>0 \leq u &lt; 1</math></p> <p>If <math>\ln(Y)</math> has a <i>normal</i> distribution, then <math>Y</math> has a <i>lognormal</i> distribution with the same parameters.</p> <p>(Simard)</p>	<p>log mu – scale parameter, any real number  log sigma – shape parameter,  log sigma &gt; 0</p>

<b>Negative Binomial</b>	<p>The negative binomial distribution with real parameters <math>\gamma</math> and <math>p</math>, where <math>\gamma &gt; 0</math> and <math>0 &lt;= p &lt;= 1</math>. Its mass function is:  <math>p(x) = \Gamma(\gamma + x) / (x! \Gamma(\gamma)) p^\gamma (1 - p)^x</math>, for <math>x = 0, 1, 2, \dots</math>  where <math>\Gamma</math> is the gamma function.</p> <p>If <math>\gamma</math> is an integer, <math>p(x)</math> can be interpreted as the probability of having <math>x</math> failures before the <math>\gamma</math>-th success in a sequence of independent Bernoulli trials with probability of success <math>p</math>.</p>	<p>Gamma – number of failures until the experiment is stopped, Gamma <math>&gt; 0</math>  <math>p</math> – success probability in each experiment, <math>0 \leq p \leq 1</math></p>																		
(Simard)																				
<b>Normal</b>	<p>The normal distribution. It has mean <math>\mu</math> and variance <math>\sigma^2</math>. Its density function is:  <math>f(x) = e^{-((x-\mu)^2/(2\sigma^2))} / ((2\pi)^{1/2} \sigma)</math> for <math>-\infty &lt; x &lt; \infty</math>, where <math>\sigma &gt; 0</math>.</p> <p>When <math>\mu = 0</math> and <math>\sigma = 1</math>, we have the standard normal distribution, with corresponding distribution function:  <math>F(x) = \Phi(x) = \int_{-\infty}^x e^{-t^2/2} dt / (2\pi)^{1/2}</math> for <math>-\infty &lt; x &lt; \infty</math>.</p>	<p>Mean – self-explanatory, any real number  Standard Deviation – self-explanatory, Std. Dev. <math>&gt; 0</math></p>																		
(Simard)																				
<b>Pareto</b>	<p>The Pareto family, with shape parameter <math>\alpha &gt; 0</math> and location parameter <math>\beta &gt; 0</math>. The density for this type of Pareto distribution is:  <math>f(x) = \alpha \beta^\alpha / x^{\alpha+1}</math> for <math>x \geq \beta</math>, and 0 otherwise.</p> <p>The distribution function is:  <math>F(x) = 1 - (\beta/x)^\alpha</math> for <math>x \geq \beta</math>,</p> <p>and the inverse distribution function is:  <math>F^{-1}(u) = \beta(1 - u)^{-1/\alpha}</math> for <math>0 &lt;= u &lt; 1</math></p>	<p>Alpha – shape parameter, alpha <math>&gt; 0</math>  Beta – location parameter, beta <math>&gt; 0</math></p>																		
(Simard)																				
<b>Poisson</b>	<p>The Poisson distribution with mean <math>\lambda \geq 0</math>. The mass function is:  <math>p(x) = e^{-\lambda} \lambda^x / (x!)</math>, for <math>x = 0, 1, \dots</math>  and the distribution function is:  <math>F(x) = e^{-\lambda} \sum_{j=0}^x \lambda^j / (j!)</math>, for <math>x = 0, 1, \dots</math></p>	<p>Lambda – mean, lambda <math>\geq 0</math></p>																		
(Simard)																				
<b>Student's T</b>	<p>The Student-t distribution with <math>n</math> degrees of freedom, where <math>n</math> is a positive integer. Its density is:  <math>f(x) = [\Gamma((n+1)/2) / (\Gamma(n/2) (\pi n)^{1/2})] [1 + x^2/n]^{-(n+1)/2}</math> for <math>-\infty &lt; x &lt; \infty</math>, where <math>\Gamma(x)</math> is the gamma function</p>	<p>D.O.F – degrees of freedom (integer), D.O.F <math>&gt; 0</math></p>																		
(Simard)																				
<b>Triangular</b>	<p>The triangular distribution with domain <math>[a, b]</math> and mode (or shape parameter) <math>m</math>, where <math>a \leq m \leq b</math>. The density function is:</p> <table border="1" data-bbox="540 1230 984 1325"> <tbody> <tr> <td><math>f(x) = 2(x - a) / [(b - a)(m - a)]</math></td> <td>for <math>a \leq x \leq m</math>,</td> </tr> <tr> <td><math>f(x) = 2(b - x) / [(b - a)(b - m)]</math></td> <td>for <math>m \leq x \leq b</math>,</td> </tr> <tr> <td><math>f(x) = 0</math></td> <td>elsewhere,</td> </tr> </tbody> </table> <p>the distribution function is:</p> <table border="1" data-bbox="540 1377 984 1514"> <tbody> <tr> <td><math>F(x) = 0</math></td> <td>for <math>x &lt; a</math>,</td> </tr> <tr> <td><math>F(x) = (x - a)^2 / [(b - a)(m - a)]</math></td> <td>if <math>a \leq x \leq m</math>,</td> </tr> <tr> <td><math>F(x) = 1 - (b - x)^2 / [(b - a)(b - m)]</math></td> <td>if <math>m \leq x \leq b</math>,</td> </tr> <tr> <td><math>F(x) = 1</math></td> <td>for <math>x &gt; b</math>,</td> </tr> </tbody> </table> <p>and the inverse distribution function is given by:</p> <table border="1" data-bbox="480 1577 1044 1640"> <tbody> <tr> <td><math>F^{-1}(u) = a + ((b - a)(m - a)u)^{1/2}</math></td> <td>if <math>0 \leq u \leq (m - a)/(b - a)</math>,</td> </tr> <tr> <td><math>F^{-1}(u) = b - ((b - a)(b - m)(1 - u))^{1/2}</math></td> <td>if <math>(m - a)/(b - a) \leq u \leq 1</math></td> </tr> </tbody> </table>	$f(x) = 2(x - a) / [(b - a)(m - a)]$	for $a \leq x \leq m$ ,	$f(x) = 2(b - x) / [(b - a)(b - m)]$	for $m \leq x \leq b$ ,	$f(x) = 0$	elsewhere,	$F(x) = 0$	for $x < a$ ,	$F(x) = (x - a)^2 / [(b - a)(m - a)]$	if $a \leq x \leq m$ ,	$F(x) = 1 - (b - x)^2 / [(b - a)(b - m)]$	if $m \leq x \leq b$ ,	$F(x) = 1$	for $x > b$ ,	$F^{-1}(u) = a + ((b - a)(m - a)u)^{1/2}$	if $0 \leq u \leq (m - a)/(b - a)$ ,	$F^{-1}(u) = b - ((b - a)(b - m)(1 - u))^{1/2}$	if $(m - a)/(b - a) \leq u \leq 1$	<p><math>a</math> – lower bound of the domain, any real number  <math>b</math> – upper bound of the domain, any real number  mode – shape parameter, any real number</p> <p>"<math>a \leq mode \leq b</math>" must be satisfied</p>
$f(x) = 2(x - a) / [(b - a)(m - a)]$	for $a \leq x \leq m$ ,																			
$f(x) = 2(b - x) / [(b - a)(b - m)]$	for $m \leq x \leq b$ ,																			
$f(x) = 0$	elsewhere,																			
$F(x) = 0$	for $x < a$ ,																			
$F(x) = (x - a)^2 / [(b - a)(m - a)]$	if $a \leq x \leq m$ ,																			
$F(x) = 1 - (b - x)^2 / [(b - a)(b - m)]$	if $m \leq x \leq b$ ,																			
$F(x) = 1$	for $x > b$ ,																			
$F^{-1}(u) = a + ((b - a)(m - a)u)^{1/2}$	if $0 \leq u \leq (m - a)/(b - a)$ ,																			
$F^{-1}(u) = b - ((b - a)(b - m)(1 - u))^{1/2}$	if $(m - a)/(b - a) \leq u \leq 1$																			
(Simard)																				
<b>Uniform</b>	<p>The uniform distribution over the interval <math>[a, b]</math>. Its density is:  <math>f(x) = 1/(b - a)</math> for <math>a \leq x \leq b</math>, and 0 elsewhere.</p> <p>The distribution function is:  <math>F(x) = (x - a)/(b - a)</math> for <math>a \leq x \leq b</math></p> <p>and its inverse is:  <math>F^{-1}(u) = a + (b - a)u</math> for <math>0 \leq u \leq 1</math></p>	<p>Min. – lower bound  Max. – upper bound  (Max. <math>&gt;</math> Min.)</p>																		
(Simard)																				



## Weibull

The *Weibull* distribution with shape parameter  $\alpha > 0$ , location parameter  $\delta$ , and scale parameter  $\lambda > 0$ . The density function is:  
 $f(x) = \alpha\lambda(x - \delta)^{\alpha-1}e^{-(\lambda(x-\delta))^\alpha}$  for  $x > \delta$ .

the distribution function is:  
 $F(x) = 1 - e^{-(\lambda(x-\delta))^\alpha}$  for  $x > \delta$ ,

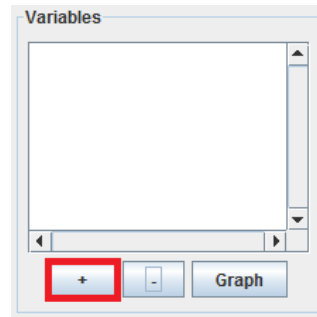
and the inverse distribution function is:  
 $F^{-1}(u) = (-\ln(1 - u))^{1/\alpha}/\lambda + \delta$  for  $0 \leq u < 1$

(Simard)

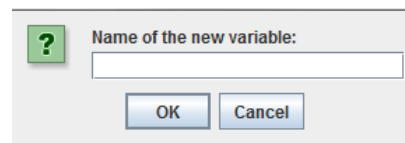
Alpha – shape parameter, alpha > 0  
Lambda – scale parameter, lambda > 0  
Delta – location parameter, any real number

## ADDING VARIABLES

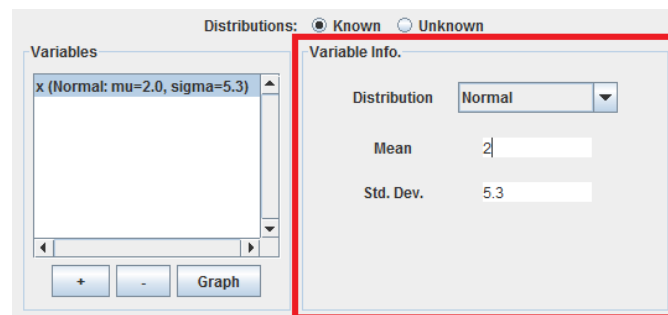
In order to add a variable to a model, the user must left-click over the **+** button that can be found in the **Main** tab:



The user will be asked to enter the new variable's name:

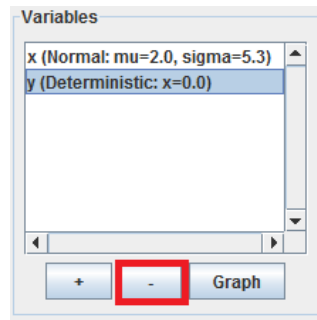


As long as there is no other previously declared variable with the same name as the one entered, the new variable will show up in the list of variables. Afterwards, the user must change the variable's information in the **Variable Info** pane, depending on the type of distribution that is to be assigned to the variable:



## REMOVING VARIABLES

Removing variables is straight-forward. The user just needs to select the variable that is to be removed from the simulation model in the **Variables** pane and then left-click over the – button:

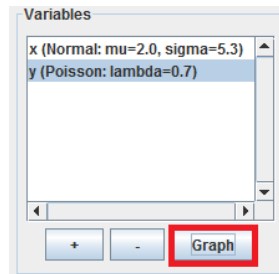


In the screenshot above, the variable **y** would disappear from the model after clicking the – button.

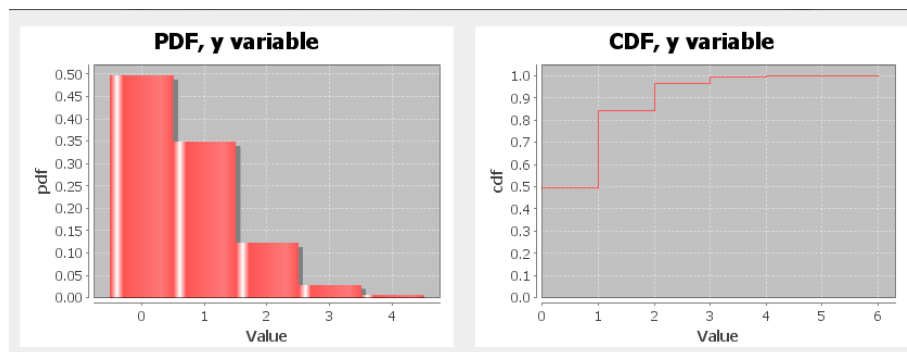
## GRAPHING A SINGLE VARIABLE

In order to verify that the parameters of the variables coincide with what the user expects, 2R Soft provides the option to visualize the PDF (probability density function) and CDF (cumulative density function) curves associated with each variable before running a model.

To view the behavior of a particular variable in graph form, the user must select such variable in the **Variables** pane and then left-click over the **Graph** button:

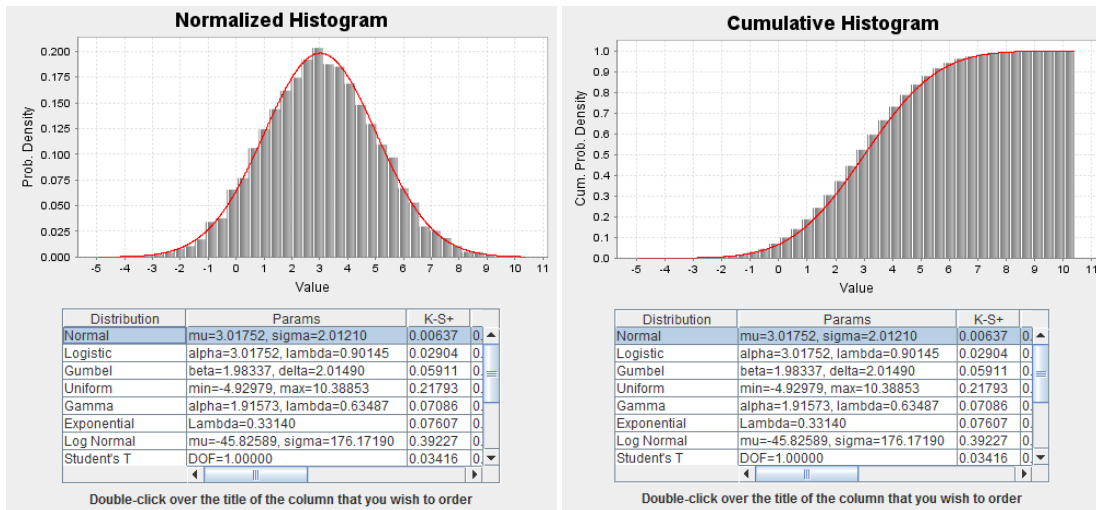


A new window will pop-up with the PDF and CDF curves:



## NORMALIZED AND CUMULATIVE HISTOGRAMS

Normalized and Cumulative histograms are found all throughout 2R Soft. These two types of graphs are of great importance, considering that they show the stochastic tendencies of a data set. With them, a goodness-of-fit table is displayed with various probability distributions and the best estimates for their parameters, along with different goodness-of-fit tests:



The distribution selected in the goodness-of-fit table is juxtaposed with the histograms (red curve). Refer to the [Probability Distribution Types](#) section for more information on the various distribution types supported by 2R Soft.

### GOODNESS-OF-FIT (GOF) STATISTICS

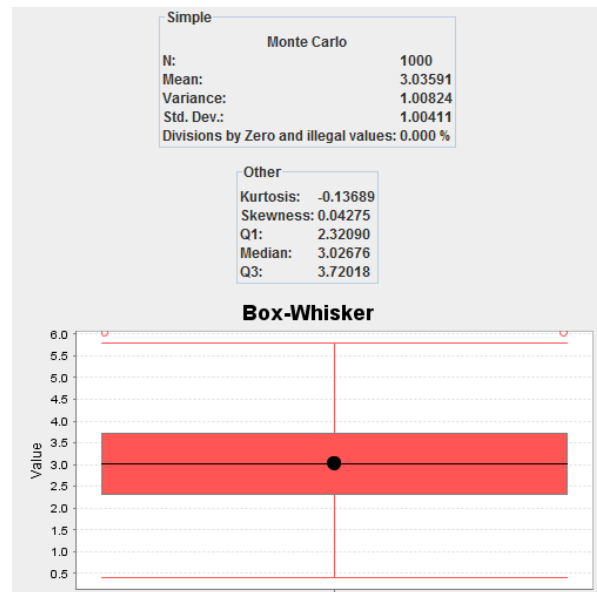
To decide whether to accept or reject a proposed probability distribution for the generated data, it is necessary to at least use one goodness-of-fit statistic as a criterion.

Col.	Test Type	Explanation	Critical Values																				
<b>A-D</b>	Anderson-Darling	<p>The Anderson-Darling test is defined as:</p> <p><math>H_0</math>: The data follow a specified distribution.</p> <p><math>H_a</math>: The data do not follow the specified distribution</p> <p>Test Statistic: The Anderson-Darling test statistic is defined as</p> $A^2 = -N - S$ <p>where</p> $S = \sum_{i=1}^N \frac{(2i-1)}{N} [\ln F(Y_i) + \ln (1 - F(Y_{N+1-i}))]$ <p><math>F</math> is the cumulative distribution function of the specified distribution. Note that the <math>Y_i</math> are the ordered data.</p> <p>The test is a one-sided test and the hypothesis that the distribution is of a specific form is rejected if the test statistic, <math>A</math>, is greater than the critical value.</p> <p>(SEMATECH2)</p>	<p>The critical values for the Anderson-Darling test are dependent on the specific distribution that is being tested. (SEMATECH2)</p> <p>For Normal and Lognormal distributions:</p> <table border="1" style="width: 100%; border-collapse: collapse; margin: 5px 0;"> <thead> <tr> <th>alpha</th> <th>0.1</th> <th>0.05</th> <th>0.025</th> <th>0.01</th> </tr> </thead> <tbody> <tr> <td><math>A^2_{crit}</math></td> <td>0.631</td> <td>0.752</td> <td>0.873</td> <td>1.035</td> </tr> </tbody> </table> <p>For Weibull and Gumbel distributions:</p> <table border="1" style="width: 100%; border-collapse: collapse; margin: 5px 0;"> <thead> <tr> <th>alpha</th> <th>0.1</th> <th>0.05</th> <th>0.025</th> <th>0.01</th> </tr> </thead> <tbody> <tr> <td><math>A^2_{crit}</math></td> <td>0.637</td> <td>0.757</td> <td>0.877</td> <td>1.038</td> </tr> </tbody> </table> <p>(Annis)</p>	alpha	0.1	0.05	0.025	0.01	$A^2_{crit}$	0.631	0.752	0.873	1.035	alpha	0.1	0.05	0.025	0.01	$A^2_{crit}$	0.637	0.757	0.877	1.038
alpha	0.1	0.05	0.025	0.01																			
$A^2_{crit}$	0.631	0.752	0.873	1.035																			
alpha	0.1	0.05	0.025	0.01																			
$A^2_{crit}$	0.637	0.757	0.877	1.038																			

K-S	Kolmogorov-Smirnov	<p>The Kolmogorov-Smirnov test is defined by:</p> <p><math>H_0</math>: The data follow a specified distribution</p> <p><math>H_a</math>: The data do not follow the specified distribution</p> <p>Test Statistic: The Kolmogorov-Smirnov test statistic is defined as</p> $D = \max_{1 \leq i \leq N} \left( F(Y_i) - \frac{i-1}{N}, \frac{i}{N} - F(Y_i) \right)$ <p>where <math>F</math> is the theoretical cumulative distribution of the distribution being tested which must be a continuous distribution (i.e., no discrete distributions such as the binomial or Poisson), and it must be fully specified.</p>	<p>An attractive feature of this test is that the distribution of the K-S test statistic itself does not depend on the underlying cumulative distribution function being tested. Therefore, the critical values are universal. (SEMATECH1)</p>																																																												
(SEMATECH1)			For samples with more than 35 values: (ERI)																																																												
<table border="1"> <thead> <tr> <th>Significance</th> <th>0.20</th> <th>0.15</th> <th>0.10</th> <th>0.05</th> <th>0.01</th> </tr> </thead> <tbody> <tr> <td>Critical Value</td> <td><math>\frac{1.07}{\sqrt{N}}</math></td> <td><math>\frac{1.14}{\sqrt{N}}</math></td> <td><math>\frac{1.22}{\sqrt{N}}</math></td> <td><math>\frac{1.36}{\sqrt{N}}</math></td> <td><math>\frac{1.63}{\sqrt{N}}</math></td> </tr> </tbody> </table>			Significance	0.20	0.15	0.10	0.05	0.01	Critical Value	$\frac{1.07}{\sqrt{N}}$	$\frac{1.14}{\sqrt{N}}$	$\frac{1.22}{\sqrt{N}}$	$\frac{1.36}{\sqrt{N}}$	$\frac{1.63}{\sqrt{N}}$																																																	
Significance	0.20	0.15	0.10	0.05	0.01																																																										
Critical Value	$\frac{1.07}{\sqrt{N}}$	$\frac{1.14}{\sqrt{N}}$	$\frac{1.22}{\sqrt{N}}$	$\frac{1.36}{\sqrt{N}}$	$\frac{1.63}{\sqrt{N}}$																																																										
K-S+ and K-S-	Kolmogorov-Smirnov+ and Kolmogorov-Smirnov-	<p>Given a sample of <math>n</math> independent uniforms <math>U_i</math> over <math>[0, 1]</math>, the Kolmogorov-Smirnov+ statistic <math>D_n^+</math> and the Kolmogorov-Smirnov- statistic <math>D_n^-</math>, are defined by</p> $D_n^+ = \max_{1 \leq j \leq n} (j/n - U_{(j)})$ $D_n^- = \max_{1 \leq j \leq n} (U_{(j)} - (j-1)/n)$ <p>where the <math>U_{(j)}</math> are the <math>U_i</math> sorted in increasing order. Both statistics follows the same distribution function, i.e. <math>F_n(x) = P[D_n^+ \leq x] = P[D_n^- \leq x]</math></p>	The same from the K-S test.																																																												
(Simard)			As with the K-S test, the critical values are universal:																																																												
CVM	Cramér-von Mises	<p>Given a sample of <math>n</math> independent uniforms <math>U_i</math> over <math>[0, 1]</math>, the Cramér-von Mises statistic <math>W_n^2</math> is defined by <math>W_n^2 = 1/12n + \sum_{j=1}^n (U_{(j)} - (j-0.5)/n)^2</math>, where the <math>U_{(j)}</math> are the <math>U_i</math> sorted in increasing order. The distribution function (the cumulative probabilities) is defined as <math>F_n(x) = P[W_n^2 \leq x]</math></p>	<table border="1"> <thead> <tr> <th colspan="6">Significance</th> </tr> <tr> <th>N</th> <th>0.20</th> <th>0.15</th> <th>0.10</th> <th>0.05</th> <th>0.01</th> </tr> </thead> <tbody> <tr> <td>2</td> <td>0.138</td> <td>0.149</td> <td>0.162</td> <td>0.175</td> <td>0.186</td> </tr> <tr> <td>10</td> <td>0.125</td> <td>0.142</td> <td>0.167</td> <td>0.212</td> <td>0.32</td> </tr> <tr> <td>20</td> <td>0.128</td> <td>0.146</td> <td>0.172</td> <td>0.217</td> <td>0.33</td> </tr> <tr> <td>30</td> <td>0.128</td> <td>0.146</td> <td>0.172</td> <td>0.218</td> <td>0.33</td> </tr> <tr> <td>60</td> <td>0.128</td> <td>0.147</td> <td>0.173</td> <td>0.220</td> <td>0.33</td> </tr> <tr> <td>100</td> <td>0.129</td> <td>0.147</td> <td>0.173</td> <td>0.220</td> <td>0.34</td> </tr> </tbody> </table>	Significance						N	0.20	0.15	0.10	0.05	0.01	2	0.138	0.149	0.162	0.175	0.186	10	0.125	0.142	0.167	0.212	0.32	20	0.128	0.146	0.172	0.217	0.33	30	0.128	0.146	0.172	0.218	0.33	60	0.128	0.147	0.173	0.220	0.33	100	0.129	0.147	0.173	0.220	0.34												
Significance																																																															
N	0.20	0.15	0.10	0.05	0.01																																																										
2	0.138	0.149	0.162	0.175	0.186																																																										
10	0.125	0.142	0.167	0.212	0.32																																																										
20	0.128	0.146	0.172	0.217	0.33																																																										
30	0.128	0.146	0.172	0.218	0.33																																																										
60	0.128	0.147	0.173	0.220	0.33																																																										
100	0.129	0.147	0.173	0.220	0.34																																																										
(Simard)			(ReliaSoftCorp)																																																												
WG	Watson G	<p>Given a sample of <math>n</math> independent uniforms <math>U_i</math> over <math>[0, 1]</math>, the <math>G</math> statistic is defined by</p> $G_n = \frac{(n)^{1/2}}{(n)^{1/2}} \max_{1 \leq j \leq n} [j/n - U_{(j)} + \text{bar}(U) - 1/2]$ $= (n)^{1/2} (D_n^+ + \text{bar}(U) - 1/2)$ <p>where the <math>U_{(j)}</math> are the <math>U_i</math> sorted in increasing order, <math>\text{bar}(U)_n</math> is the average of the observations <math>U_i</math>, and <math>D_n^+</math> is the Kolmogorov-Smirnov+ statistic. The distribution function (the cumulative probabilities) is defined as <math>F_n(x) = P[G_n \leq x]</math></p>	From 2R Soft WG CDF code:																																																												
(Simard)			<table border="1"> <thead> <tr> <th colspan="6">Significance</th> </tr> <tr> <th>N</th> <th>0.20</th> <th>0.15</th> <th>0.10</th> <th>0.05</th> <th>0.01</th> </tr> </thead> <tbody> <tr> <td>2</td> <td>0.609</td> <td>0.636</td> <td>0.670</td> <td>0.718</td> <td>0.787</td> </tr> <tr> <td>10</td> <td>0.692</td> <td>0.728</td> <td>0.773</td> <td>0.843</td> <td>0.979</td> </tr> <tr> <td>20</td> <td>0.711</td> <td>0.747</td> <td>0.794</td> <td>0.866</td> <td>1.010</td> </tr> <tr> <td>30</td> <td>0.719</td> <td>0.755</td> <td>0.802</td> <td>0.875</td> <td>1.021</td> </tr> <tr> <td>60</td> <td>0.728</td> <td>0.765</td> <td>0.813</td> <td>0.887</td> <td>1.034</td> </tr> <tr> <td>100</td> <td>0.734</td> <td>0.770</td> <td>0.818</td> <td>0.893</td> <td>1.041</td> </tr> <tr> <td>1000</td> <td>0.745</td> <td>0.782</td> <td>0.831</td> <td>0.906</td> <td>1.055</td> </tr> <tr> <td>10000</td> <td>0.749</td> <td>0.786</td> <td>0.834</td> <td>0.909</td> <td>1.059</td> </tr> </tbody> </table>	Significance						N	0.20	0.15	0.10	0.05	0.01	2	0.609	0.636	0.670	0.718	0.787	10	0.692	0.728	0.773	0.843	0.979	20	0.711	0.747	0.794	0.866	1.010	30	0.719	0.755	0.802	0.875	1.021	60	0.728	0.765	0.813	0.887	1.034	100	0.734	0.770	0.818	0.893	1.041	1000	0.745	0.782	0.831	0.906	1.055	10000	0.749	0.786	0.834	0.909	1.059
Significance																																																															
N	0.20	0.15	0.10	0.05	0.01																																																										
2	0.609	0.636	0.670	0.718	0.787																																																										
10	0.692	0.728	0.773	0.843	0.979																																																										
20	0.711	0.747	0.794	0.866	1.010																																																										
30	0.719	0.755	0.802	0.875	1.021																																																										
60	0.728	0.765	0.813	0.887	1.034																																																										
100	0.734	0.770	0.818	0.893	1.041																																																										
1000	0.745	0.782	0.831	0.906	1.055																																																										
10000	0.749	0.786	0.834	0.909	1.059																																																										
WU	Watson U	<p>Given a sample of <math>n</math> independent uniforms <math>u_i</math> over <math>[0, 1]</math>, the Watson statistic <math>U_n^2</math> is defined by <math>W_n^2 = 1/12n + \sum_{j=1}^n [u_{(j)} - (j-0.5)/n]^2</math>, <math>U_n^2 = W_n^2 - n(\text{bar}(u)_n - 1/2)^2</math>, where the <math>u_{(j)}</math> are the <math>u_i</math> sorted in increasing order, and <math>\text{bar}(u)_n</math> is the average of the observations <math>u_i</math>. The distribution function (the cumulative probabilities) is defined as <math>F_n(x) = P[U_n^2 \leq x]</math></p>	From 2R Soft WU CDF code:																																																												
(Simard)			<table border="1"> <thead> <tr> <th colspan="6">Significance</th> </tr> <tr> <th>N</th> <th>0.20</th> <th>0.15</th> <th>0.10</th> <th>0.05</th> <th>0.01</th> </tr> </thead> <tbody> <tr> <td>2</td> <td>0.122</td> <td>0.132</td> <td>0.143</td> <td>0.154</td> <td>0.164</td> </tr> <tr> <td>10</td> <td>0.116</td> <td>0.130</td> <td>0.150</td> <td>0.183</td> <td>0.255</td> </tr> <tr> <td>20</td> <td>0.116</td> <td>0.131</td> <td>0.151</td> <td>0.185</td> <td>0.262</td> </tr> <tr> <td>30</td> <td>0.117</td> <td>0.131</td> <td>0.151</td> <td>0.185</td> <td>0.264</td> </tr> <tr> <td>60</td> <td>0.117</td> <td>0.131</td> <td>0.151</td> <td>0.186</td> <td>0.266</td> </tr> <tr> <td>100</td> <td>0.117</td> <td>0.131</td> <td>0.152</td> <td>0.186</td> <td>0.267</td> </tr> <tr> <td>1000</td> <td>0.117</td> <td>0.131</td> <td>0.152</td> <td>0.187</td> <td>0.268</td> </tr> <tr> <td>10000</td> <td>0.117</td> <td>0.131</td> <td>0.152</td> <td>0.187</td> <td>0.268</td> </tr> </tbody> </table>	Significance						N	0.20	0.15	0.10	0.05	0.01	2	0.122	0.132	0.143	0.154	0.164	10	0.116	0.130	0.150	0.183	0.255	20	0.116	0.131	0.151	0.185	0.262	30	0.117	0.131	0.151	0.185	0.264	60	0.117	0.131	0.151	0.186	0.266	100	0.117	0.131	0.152	0.186	0.267	1000	0.117	0.131	0.152	0.187	0.268	10000	0.117	0.131	0.152	0.187	0.268
Significance																																																															
N	0.20	0.15	0.10	0.05	0.01																																																										
2	0.122	0.132	0.143	0.154	0.164																																																										
10	0.116	0.130	0.150	0.183	0.255																																																										
20	0.116	0.131	0.151	0.185	0.262																																																										
30	0.117	0.131	0.151	0.185	0.264																																																										
60	0.117	0.131	0.151	0.186	0.266																																																										
100	0.117	0.131	0.152	0.186	0.267																																																										
1000	0.117	0.131	0.152	0.187	0.268																																																										
10000	0.117	0.131	0.152	0.187	0.268																																																										

## STATISTICS

When appropriate, 2R Soft calculates an array of statistics that can be both relevant and pertinent for users interested in analyzing the behavior of a data set.



The screenshot above is an example of what the user should expect to find in a **Statistics** tab. The following subsections of this document explain each of the statistics that are calculated by 2R Soft.

### MEAN

The arithmetic mean of a set of values is the quantity commonly called "the" mean or the average. Given a set of samples  $\{x_i\}$ , the arithmetic mean is: (Weisstein2)

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$$

### VARIANCE

For a series of data, the sample variance may be computed as: (Weisstein3)

$$s_N^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2$$

where  $\bar{x}$  is the sample mean.

Note that the sample variance  $s_N^2$  defined above is *not* an unbiased estimator for the population variance,  $\sigma^2$ . In order to obtain an unbiased estimator for  $\sigma^2$ , it is necessary to instead define a "bias-corrected sample variance": (Weisstein3)

$$s_{N-1}^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2$$

2R Soft uses the bias-corrected sample variance.

From the mathematical definition of variance, it is clear that this statistic expresses the dispersion of the data with respect to the mean. The larger the variance, the more spread out is the data.

---

## STANDARD DEVIATION

The standard deviation formula is very simple: it is the square root of the variance. It is the most commonly used measure of spread (Lane2). Hence, an unbiased estimator of the population standard deviation,  $\sigma$ , is given by:

$$s_{N-1} = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2}$$

---

## DIVISIONS BY ZERO AND ILLEGAL VALUES

This statistic expresses the proportion of failed iterations during the simulation model run as a percentage of the total number of simulations,  $N$ . A failed iteration is defined as one that didn't return a real number when the equation was evaluated. There are two possible reasons for a failed iteration to arise:

1. There was a division by zero.
2. One or more of the functions used in the equation received an illegal parameter. For example, **log(x)** is only defined for positive real numbers. If  $\mathbf{x}$  is a random variable and the generated value for  $\mathbf{x}$  is negative in an iteration, then **log(x)** cannot be evaluated, yielding a failed iteration.

The amount of usable generated values can be calculated from this statistic:

$$\text{Usable generated values} = (1 - \% \text{ divisions by zero and illegal values}) * N$$

For example, if the reported percentage of failed iterations is 5% for a simulation with  $N=100$ , then the simulation analysis (goodness-of-fit, statistics, histograms, convergence graphs, etc) is carried out with 95 data points.

---

## KURTOSIS

Kurtosis is a measure of whether the data are peaked or flat relative to a normal distribution. That is, data sets with high kurtosis tend to have a distinct peak near the mean, decline rather rapidly, and have heavy tails. Data sets with low kurtosis tend to have a flat top near the mean rather than a sharp peak. A uniform distribution would be the extreme case. For univariate data  $Y_1, Y_2, \dots, Y_N$ , the formula for kurtosis is: (SEMATECH3)

$$\text{kurtosis} = \frac{\sum_{i=1}^N (Y_i - \bar{Y})^4}{(N-1)s^4}$$

where  $\bar{Y}$  is the mean,  $s$  is the standard deviation, and  $N$  is the number of data points.

---

## SKEWNESS

Skewness is a measure of symmetry, or more precisely, the lack of symmetry. A distribution, or data set, is symmetric if it looks the same to the left and right of the center point. For univariate data  $Y_1, Y_2, \dots, Y_N$ , the formula for skewness is: (SEMATECH3)

$$\text{skewness} = \frac{\sum_{i=1}^N (Y_i - \bar{Y})^3}{(N-1)s^3}$$

where  $\bar{Y}$  is the mean,  $s$  is the standard deviation, and  $N$  is the number of data points. The skewness for a normal distribution is zero, and any symmetric data should have a skewness near zero. Negative values for the skewness indicate data that are skewed left and positive values for the skewness indicate data that are skewed right. By

skewed left, we mean that the left tail is long relative to the right tail. Similarly, skewed right means that the right tail is long relative to the left tail. Some measurements have a lower bound and are skewed right. For example, in reliability studies, failure times cannot be negative. (SEMATECH3)

---

## QUARTILES (Q1, MEDIAN, Q3)

The quartiles of a data set are formed by the two boundaries on either side of the median, which divide the set into four equal sections. The lowest 25% of the data being found below the first quartile value, also called the lower quartile (Q1). The median, or second quartile divides the set into two equal sections. The lowest 75% of the data set should be found below the third quartile, also called the upper quartile (Q3). These three numbers are measures of the dispersion of the data, while the mean, median and mode are measures of central tendency. (EncyclopediaOfStatistics)

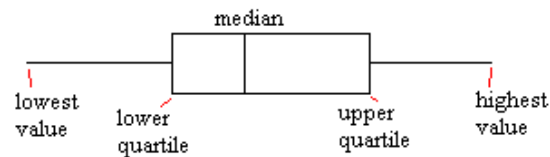
---

## BOX-WHISKER DIAGRAM

Given some data, a **box and whisker diagram** (or box plot) can be drawn to show the spread of the data. The diagram shows the quartiles of the data, using these as an indication of the spread. (MathsRevision.net)

The diagram is made up of a "box", which lies between the upper and lower quartiles. The median can also be indicated by dividing the box into two. (MathsRevision.net)

The "whiskers" are straight line extending from the ends of the box to the maximum and minimum values: (MathsRevision.net)

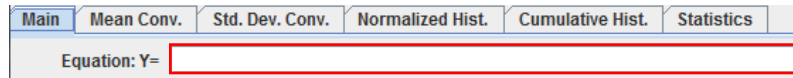


In 2R Soft, the Box-Whisker diagram also contains a black dot, which marks the arithmetic mean of the generated values.

## MAIN TAB

### EQUATION

2R Sim only supports explicit equations, and only one equation can be associated with a specific model. The equation is to be written in the **Main** tab:



The screenshot shows a software interface with several tabs: 'Main', 'Mean Conv.', 'Std. Dev. Conv.', 'Normalized Hist.', 'Cumulative Hist.', and 'Statistics'. The 'Main' tab is selected. Below the tabs, there is a text input field labeled 'Equation: Y=' which is currently empty and highlighted with a red rectangular border.

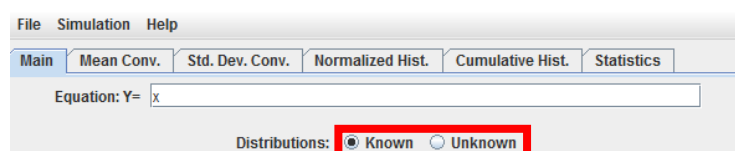
Refer to the [Equation Editor](#) section for information on supported functions and equation syntax in general.

### KNOWN VS UNKNOWN DISTRIBUTIONS

When declaring the variables involved in a simulation model, the user must decide if these have known or unknown probability distributions. The differences between both scenarios are:

Known Distributions	Unknown Distributions
<ul style="list-style-type: none"><li>• Depending on the distributions being used, the user will have to enter between 2 and 4 parameters per variable.</li><li>• 3 different types of simulation can be carried out: Monte Carlo, Latin Hypercube, and Orthogonal Latin Hypercube.</li><li>• The results obtained depend on the type of simulation and number of simulation iterations of the run. Furthermore, the results are never exactly the same, but can be considered practically identical if the simulation parameters being used are well chosen.</li></ul>	<ul style="list-style-type: none"><li>• The user only needs to input the mean and standard deviation for each variable.</li><li>• Only one type of simulation can be carried out: Rosenblueth <math>2^{k+1}</math>.</li><li>• The model will give exactly the same results every time it is analyzed.</li></ul>

To indicate the type of model that will be used, the user must select the correct radio button in the **Main** tab:



The screenshot shows the same software interface as before, but with the 'Equation: Y=' field containing the letter 'x'. Below the equation field, there is a section labeled 'Distributions:' with two radio buttons: 'Known' and 'Unknown'. The 'Known' radio button is selected and highlighted with a red rectangular border.



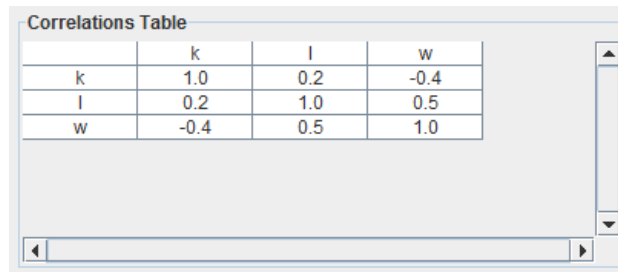
## CORRELATIONS MATRIX (CORRELATIONS TABLE)

The correlation coefficient is a single number between -1 and 1 that describes the dependence between two variables. The formula used to compute the correlation coefficient of two variables from experimental data is (Lane):

$$r = \frac{\sum XY - \frac{\sum X \sum Y}{N}}{\sqrt{(\sum X^2 - \frac{(\sum X)^2}{N})(\sum Y^2 - \frac{(\sum Y)^2}{N})}}$$

When two variables are independent from each other, their correlation coefficient is equal to 0. On the other hand, if two variables are perfectly dependent, their correlation coefficient is equal to 1 (as one increases the other one also increases) or -1 (as one increases the other one decreases, and vice-versa).

During a simulation run, random values of the different variables must be generated. Hence, 2R Sim must take correlations into account to make simulations as accurate as possible, since the value generated for a variable might influence the probability distribution of another variable in a simulation's iteration. This information is entered by the user in the form of a **Correlation Matrix** that is shown in the **Main** tab:

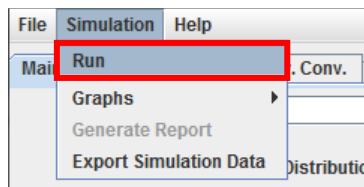


	k	l	w
k	1.0	0.2	-0.4
l	0.2	1.0	0.5
w	-0.4	0.5	1.0

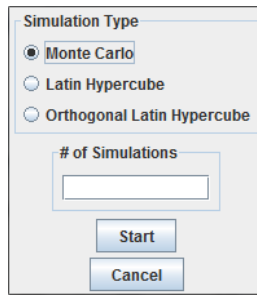
The resulting matrix is always symmetric and shows the multiple dependencies that exist between all the variables contained in the current model. To edit a correlation coefficient, the user must double-click over the corresponding cell and enter the new value.

## RUNNING A SIMULATION

When the simulation model is complete (equation, variables, correlation matrix), the simulation process can be started by navigating through the **Simulation** menu and selecting the **Run** option:

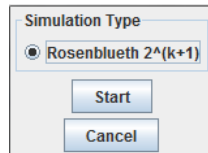


If the current model deals with known distributions, the following screen will appear:



After entering a valid number of simulations and selecting the appropriate simulation type, pressing the **Start** button will begin the simulation.

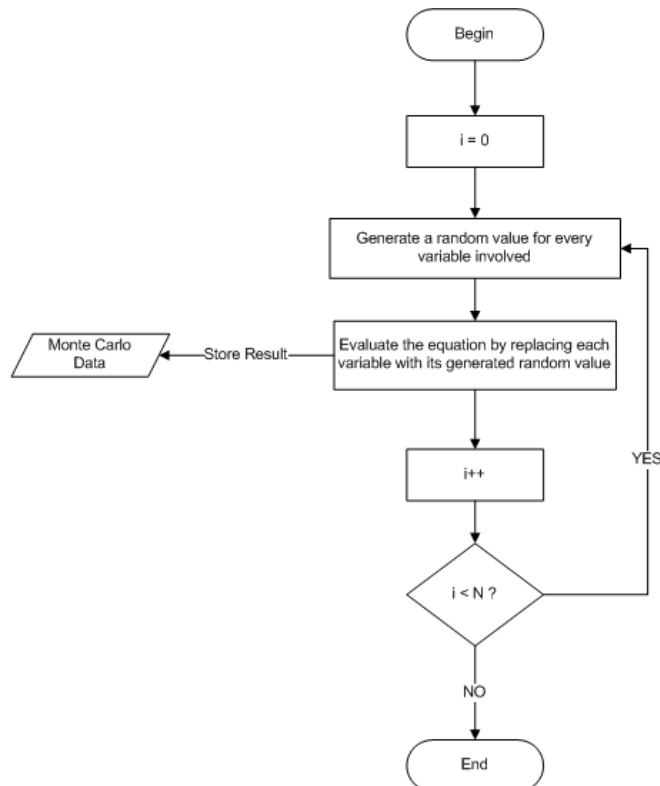
If the current model deals with unknown distributions, the following screen will appear:



In this case, the only action needed for the simulation to start is a left click over the **Start** button. This is due to the fact that there is a single simulation type available for models with unknown distributions, and the number of simulations is given by the amount of variables used in the equation, which is explained in the Rosenblueth section of this document.

## MONTE CARLO

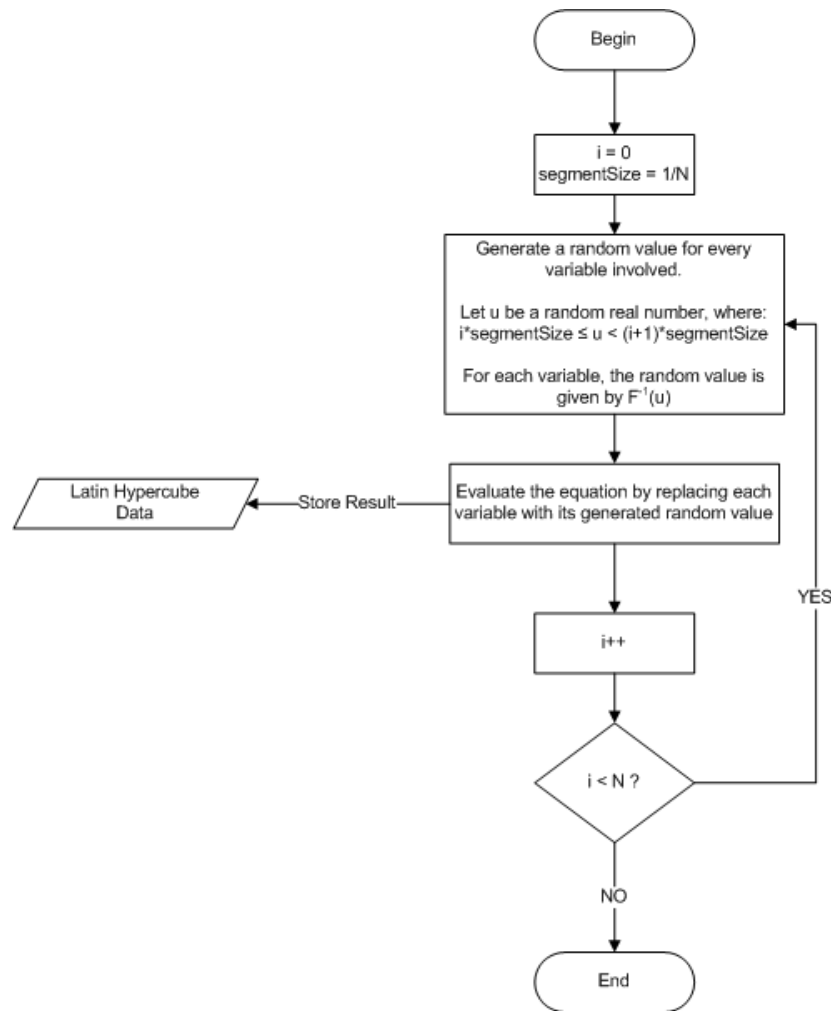
The Monte Carlo algorithm follows the flow chart shown below: (N is the number of simulations to be carried out)



Although the Monte Carlo algorithm is simple, one run can be time-consuming because the generation of random numbers and the evaluation of an equation are processor-intensive procedures. Nonetheless, each iteration is independent from the rest, which enables the use of a thread pool to run various iterations at the same time depending on the amount of processing cores available in the computer running the program. 2R Sim uses the thread pool strategy to speed up Monte Carlo runs.

## LATIN HYPERCUBE

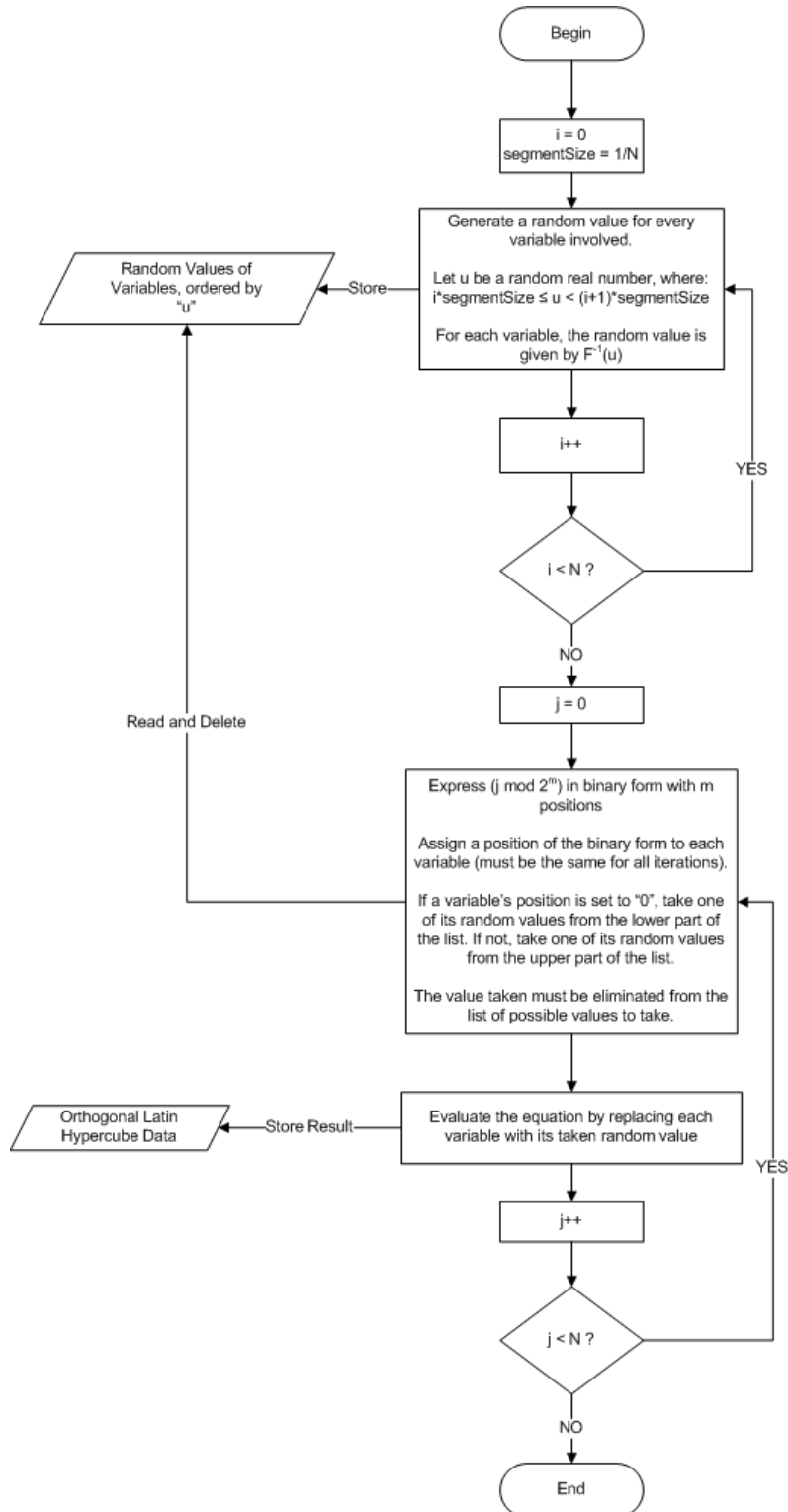
The Latin Hypercube algorithm uses a sampling process different from the one in the basic Monte Carlo algorithm, leading to a better coverage of the sampling space with a smaller number of iterations. As expected, the resulting flow chart is more elaborate than the one shown in the previous section: (N is the number of simulations to be carried out)



As the flow chart shows, the Latin Hypercube algorithm divides the range [0,1] in N partitions of the same size (segmentSize), and then takes one value, u, from each of those partitions to generate the random values of the variables. The thread pool strategy mentioned in the previous section is also employed by 2R Sim with this algorithm, taking into account that the iterations are still independent from each other.

## ORTHOGONAL LATIN HYPERCUBE

This algorithm refines the sampling process of the normal Latin Hypercube by dividing the sampling space into subspaces with the same probability, where the amount of samples taken from each of those subspaces is the same. If each variable's sampling space is divided into two sections (upper and lower), there will be  $2^m$  equally probable subspaces to be monitored, being  $m$  the number of variables being used. The resulting flow chart is: ( $N$  is the number of simulations to be carried out, and is a multiple of  $2^m$ )

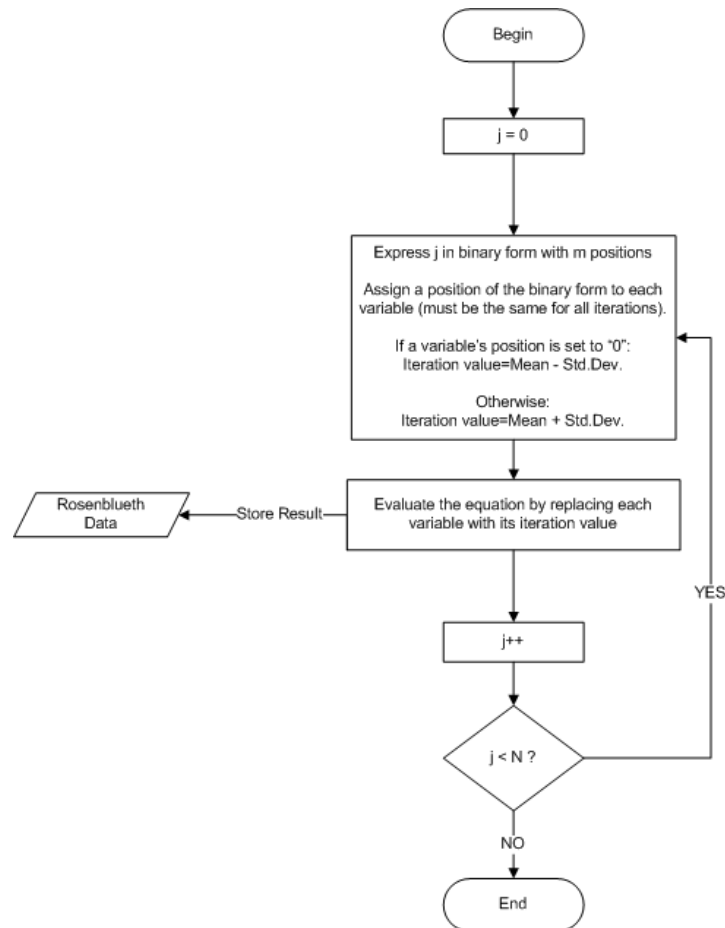


As shown in the flow diagram, the Orthogonal Latin Hypercube algorithm is comprised of two cycles: one that calculates random values for the variables and another one that picks generated random values according to the appropriate subspaces and evaluates the equation with the values picked. The binary form is used as a way to assure that each subspace is sampled the same number of times as the rest. As with the previous algorithms, the Orthogonal Latin Hypercube is optimized in 2R Sim by using a thread pool due to the independence of the iterations.

## ROSENBLUETH ( $2^{k+1}$ )

The Rosenblueth ( $2^{k+1}$ ) algorithm is a simple way to simulate the behavior of an equation containing variables with unknown distributions. The calculations involved only require the means and standard deviations of such variables.

Let  $N = 2^m$ , where  $N$  is the number of simulations to be carried out and  $m$  is the number of variables being used:



The flow chart shows that the idea of this algorithm is to evaluate the equation with all the possible “*Mean ± Standard Deviation*” permutations that can be generated with the variables being used. Taking into account that 2 possible states exist per variable (Mean+Std.Dev. and Mean-Std.Dev.), the amount of values to be generated is given by  $2^m$ .

This algorithm is also run by 2R Sim with a thread pool strategy to minimize the processing time.

## MEAN AND STANDARD DEVIATION CONVERGENCE GRAPHS

The mean and standard deviation convergence graphs are important tools that indicate whether the simulation was successful or not. One should only use the results from a simulation if there is a clear convergence in both statistics.

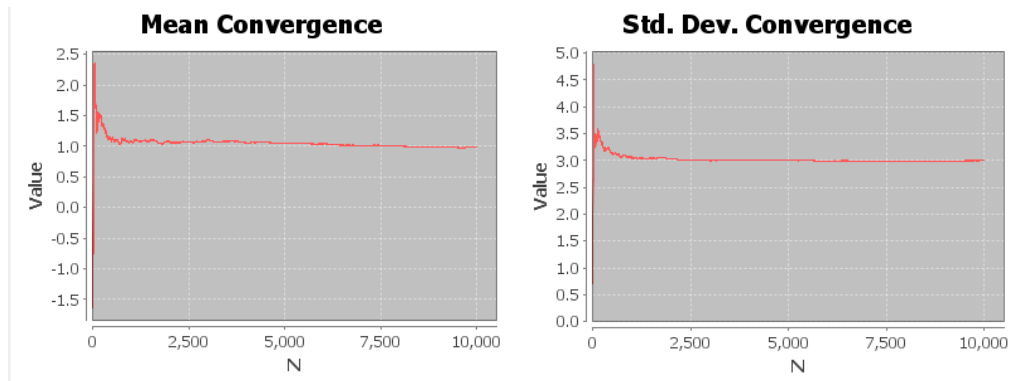
These statistics are calculated with the following equations (Weisstein): ( $N$  is the total number of simulations)

$$Mean_i = \frac{\sum_{k=0}^i result_k}{i+1}, \quad 0 \leq i < N$$

$$Std.Dev._i = \sqrt{\frac{\sum_{k=0}^i (result_k - Mean_i)^2}{i}}, \quad 1 \leq i < N$$

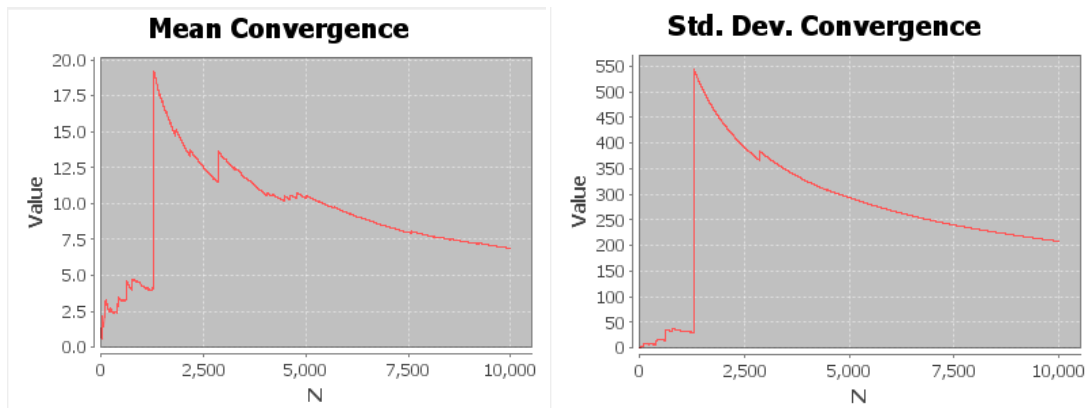
The standard deviation being calculated is an unbiased estimator for the population variance, since it is equivalent to the statistic known as  $s_{N-1}^2$ .

An example of a converging simulation:



Clearly, the simulation above is stable after 2,500 simulations and has a mean of approximately 1.0 and standard deviation of around 3.0.

An example of a diverging simulation:



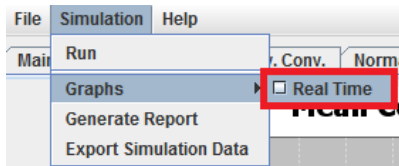
In this case, the simulation doesn't show a clear trend towards any value, even after 10,000 simulations have been performed.

If a simulation fails to converge, there are two possibilities:

1. **The number of simulations carried out was insufficient.** If this is true, the user should increase the amount of simulations and run the model again.
2. **The equation doesn't converge.** Some equations don't have a defined expected value and, consequently, are unpredictable. An example of such phenomenon is the equation  $1/x$  when  $x$  has an exponential distribution with lambda of 0.5.

## REAL TIME GRAPHS

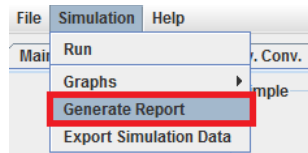
By default, 2R Sim displays the mean and standard deviation convergence graphs after the simulation is completed. However, 2R Sim is capable of generating the mentioned graphs dynamically while a simulation is running. To enable this behavior, navigate through the **Simulation** menu and activate the **Real Time** checkbox under the **Graphs** submenu:



Warning: even though this option can be visually appealing, it does require a larger amount of processing power. Thus, simulations run with real time graphs are bound to take a larger amount of time to complete.

## REPORT GENERATION

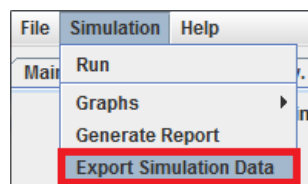
2R Sim is capable of exporting all the useful information that results from a simulation run to PDF format. To do this, a user must navigate through the **Simulation** menu and select the **Generate Report** option. This option will only be enabled if a successful run has been completed.



The user is then prompted for the new report's name and location before 2R Sim creates the PDF file.

## EXPORTING SIMULATION DATA

Even though 2R Sim provides a large amount of statistics and analysis tools, some users might want to analyze the data themselves with other software packages. For that reason, 2R Sim users are given the option to export the values generated during the simulation to XLSX (Excel 2007) format. The option to do this is located under the **Simulation** menu and is labeled as **Export Simulation Data**.



The resulting spreadsheet contains the values corresponding to the evaluation of the equation as a whole in each iteration.

EVENTS IN TIME – RANDOM SHOCKS

Random shocks are events with variable magnitudes and/or non-deterministic times of occurrence. These shocks come **one after the other**, meaning that they comprise a sequence of events in time. Nonetheless, they are completely independent from one another.

INPUT

- **Magnitudes** – an equation indicating the shock magnitude for the events to be generated. It can be as simple as “x” or more elaborate, such as “sin(y)+sqrt(z)/p”, where **x**, **y**, **z**, and **p** are random variables. Refer to the [Equation Editor](#) section for more information on equation syntax and variable management.
- **Time between events** – an equation indicating the time between one event and the next. It can be as simple as “x” or more elaborate, such as “sin(y)+sqrt(z)/p”, where **x**, **y**, **z**, and **p** are random variables. Refer to the [Equation Editor](#) section for more information on equation syntax and variable management.

Values will be generated until the user’s condition is satisfied. Such condition can be expressed in two possible ways:

- **t\_max**: data generation stops when a shock’s time of occurrence exceeds the specified time.
- **# of Events**: data generation stops when the specified number of shocks have been generated.

OUTPUT

The resulting shocks and their corresponding magnitudes are displayed in tabular form:

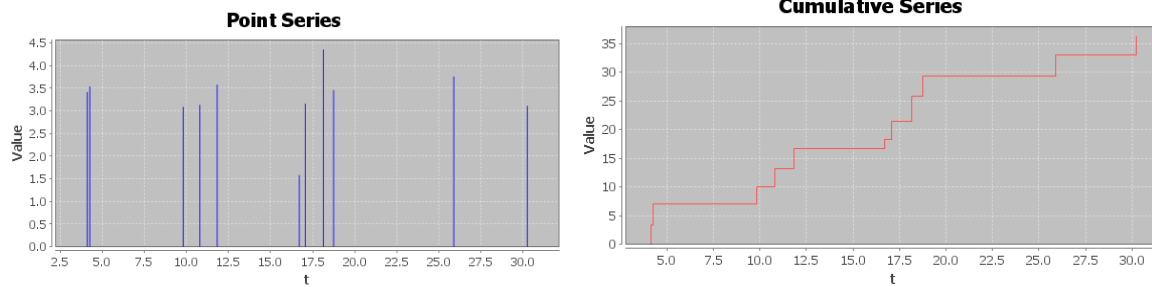
#	t	mag.	cum. mag.
1	4.1178907...	3.4127281...	3.4127281...
2	4.2656759...	3.5385163...	6.9512445...
3	9.8116238...	3.0856933...	10.036937...
4	10.805468...	3.1302027...	13.167140...
5	11.832457...	3.5775160...	16.744656...
6	16.709909...	1.5802645...	18.324921...
7	17.062256...	3.1575810...	21.482502...
8	18.139478...	4.3512190...	25.833721...
9	18.747661...	3.4551498...	29.288871...
10	25.885016...	3.7563309...	33.045202...
11	28.854558...	3.4884884...	36.533690...

[Export to Excel](#)

Data can be exported to excel with no hassle.



Two graphical representations of the data are shown:



The "Point Series" shows each event on its own, while the "Cumulative Series" shows the sum of shock magnitudes for every moment in time.

## EVENTS IN TIME – UNCORRELATED TIME SERIES

Uncorrelated time series follow a specific trend in time ( $\mu$ ) with some variability ( $\sigma$ ). Each unitary increase in time (from  $t=0$  to  $t=1$ ,  $t=1$  to  $t=2$ , etc) is considered as an independent event. Consequently, a 10-event uncorrelated time series goes from  $t=0$  to  $t=10$ . Both  $\mu$  and  $\sigma$  can be functions of time,  $t$ . The data generation algorithm is described below:

1.  $t=0$
2. Let  $M=\mu(t)$  and  $S=\sigma(t)$ . To determine the value of the time series at  $t$ , a normally distributed variable with mean  $M$  and standard deviation  $S$  is created and a random value of that variable is taken.
3.  $t$  increases by 1. If  $t$  is less than the number of events to be generated, step 2 is carried out again. Otherwise, data generation ends.

### INPUT

The input dialog box contains the following elements:

- A label "Sigma" next to a "Set" button.
- A label "Mu" next to a "Set" button.
- A label "# of Events" next to a text input field containing the value "1.0".
- A "Start" button.
- A "Cancel" button.

- **Sigma** – an equation expressing the variability of the time series at time,  $t$ . If  $\sigma$  is time-dependent, the variable " $t$ " should be part of the equation. Refer to the [Equation Editor](#) section for more information on equation syntax and variable management.
- **Mu** – an equation expressing the time series' trend as a function of time,  $t$ . If  $\mu$  is time-dependent, the variable " $t$ " should be part of the equation. Refer to the [Equation Editor](#) section for more information on equation syntax and variable management.
- **# of Events** – the amount of "steps" in time to be simulated.

## OUTPUT

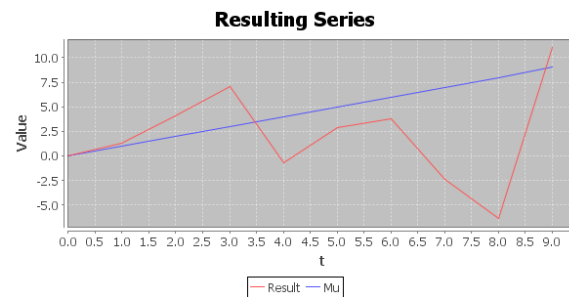
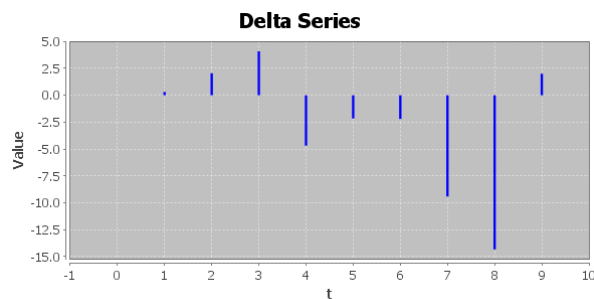
The resulting time series is displayed in tabular form, where the “res.” column contains the value generated on each  $t$ . The “delta” column is calculated as “res.- $\mu(t)$ ”, giving a point’s deviation from the trend.

t	mu	delta	res.
0.0	0.0	0.0	0.0
1.0	1.0	0.3028655...	1.3028655...
2.0	2.0	2.0411055...	4.0411055...
3.0	3.0	4.0858512...	7.0858512...
4.0	4.0	-4.6866432...	-0.6866432...
5.0	5.0	-2.1584629...	2.8415370...
6.0	6.0	-2.2069121...	3.7930878...
7.0	7.0	-9.4156847...	-2.4156847...
8.0	8.0	-14.331452...	-6.3314521...
9.0	9.0	1.9932450...	10.993245...

Export to Excel

Data can be exported to excel with no hassle.

In addition to the table, “Delta Series” and “Resulting Series” graphs are presented. The former displays the deviations with respect to  $\mu(t)$ , while the latter contrasts the resulting time series (red) with the basic trend (blue):



## SPATIAL EVENTS

Random spatial events can be generated in a 2-dimensional space. Their magnitude can be expressed as a random variable, although the default setting is for all of the events to have a magnitude of exactly 1. Most importantly, the (X,Y) coordinates of the events depend on the directional distribution and on the arbitrary boundaries selected by the user.

The basic algorithm for spatial event generation is as follows:

1. A direction (between 0 and  $2\pi$ ) is randomly selected.
2. Events are generated in that direction. This step depends on the “directional distribution” setting.
3. Steps 1 and 2 are repeated until an appropriate number of events have been generated.

## INPUT

Center X: 0.0  
Center Y: 0.0  
Magnitudes: Set  
# of Events: 1  
Directional Distribution of Events:  
 Distance between events  # of Events  
Set  
Boundaries:  
 Circle  
r = 1.0  
 Rect./Band (Blank = +/- infinity)  
X Max. Y Max.  
X Min. Y Min.  
Start  
Cancel

The **Center X** and **Center Y** coordinates define the point from which events will be generated in each random direction. Meanwhile, the **Magnitudes** random variable establishes the possible values for the magnitude of each event (refer to the [Probability Distribution Types](#) section for information on probability distributions and their parameters). In addition, the **# of Events** textbox is used to set the number of data points to be generated.

The directional distribution has 2 possible settings:

- **Distance Between Events** – the user selects a random variable to express the distance between one event and the next (refer to the [Probability Distribution Types](#) section for information on probability distributions and their parameters). Under this scheme, step 2 of the basic algorithm begins at the center point and continues in the current direction, **one event after the other**, until a boundary is reached.
- **# of Events** – the user selects a random variable to express the number of events per random direction (refer to the [Probability Distribution Types](#) section for information on probability distributions and their parameters). Under this scheme, step 2 of the basic algorithm evenly distributes the distance between the center point and the closest boundary in the current direction to fit the number of events given by the random variable.

Boundaries also have 2 alternatives:

- **Circle** - a circle of radius “r” around the center point is set to be the boundary; hence, no data point can fall beyond that circle.
- **Rect./Band** – this is a more flexible type of boundary. If **X Max** and/or **Y Max** are left blank, there is no upper limit in the **X** and/or **Y** coordinates of the events. Accordingly, if **X Min** and/or **Y Min** are left blank, there is no lower limit in the **X** and/or **Y** coordinates of the events. This way, the user can decide which dimensions to limit and how to limit them.

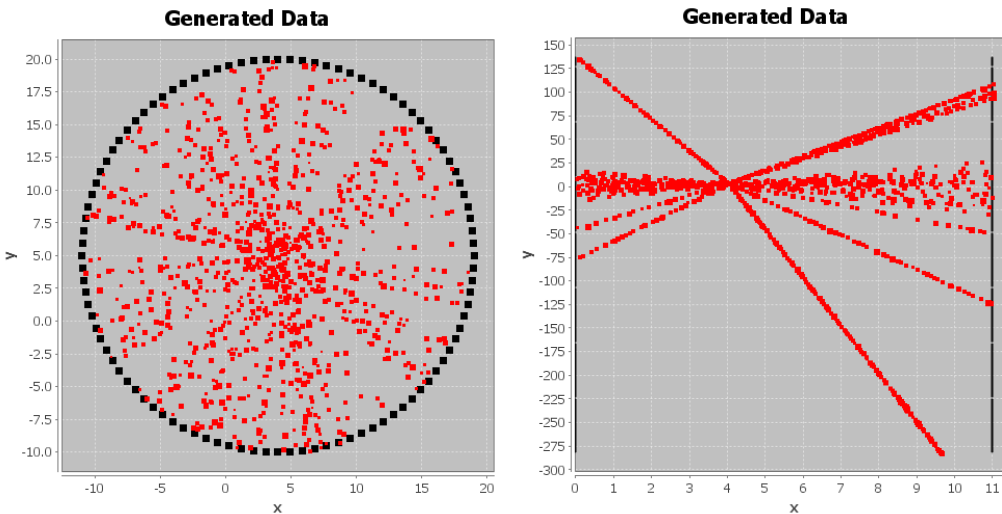
## OUTPUT

The resulting events are summarized in a table: (the “mag.” column shows the magnitude of the events)

#	x	y	mag.
1	2.9852938...	5.6148377...	106.92253...
2	0.7542502...	6.9666871...	99.604588...
3	-1.3819281...	8.2610550...	77.616981...
4	-4.1905903...	9.9628990...	105.30713...
5	-6.9042485...	11.607177...	97.338394...
6	3.8119143...	5.8664080...	94.184836...
7	3.1391114...	8.9656442...	126.65220...
8	2.4306474...	12.229152...	104.38658...
9	2.1869662...	13.351658...	81.654189...
10	1.5845344...	16.126733...	95.616889...

Export to Excel

A graphical representation of the spatial events is also shown: (black lines are boundaries, red boxes are events)



Events with higher magnitudes are drawn with larger boxes.

## RANDOM WALK

Random Walk is a special type of 2-dimensional event generation. In random walk simulations, an “event” isn’t a point in space, but rather a path that starts in a specified center coordinate and is generated as follows:

1. A random distance given by a **Step Size** variable is initially covered in one of 4 directions: Up, Down, Right, or Left, chosen at random.
2. Based on **p(right)**, the probability of making a right turn, the program randomly decides whether it is going to make a left or a right turn. A right turn constitutes a clockwise rotation of 90 degrees, while a left turn constitutes a counterclockwise rotation of the same magnitude.
3. A random distance given by a **Step Size** variable is covered in the new direction.
4. Steps 2 and 3 are repeated until a boundary is reached or until a specified number of steps have been generated, depending on the “boundary” settings.

## INPUT

The 'Boundaries' panel contains the following elements:

- A radio button labeled 'None' which is selected.
- A radio button labeled 'Circle'.
- A text input field labeled 'r =' with the value '1.0' entered.
- A radio button labeled 'Rect./Band (Blank = +/- infinity)'.
- Four text input fields: 'X Max.', 'Y Max.', 'X Min.', and 'Y Min.', all of which are currently blank.

Boundaries can be of 3 different types:

- **None** – in this case, the random walk simulation runs for a specified number of steps.
- **Circle** – in this case, a circular boundary of radius “*r*” is set around the center point and simulations stop when they reach that boundary.
- **Rect./Band** - this is a more flexible type of boundary. If **X Max** and/or **Y Max** are left blank, there is no upper limit in the **X** and/or **Y** coordinates of the events. Accordingly, if **X Min** and/or **Y Min** are left blank, there is no lower limit in the **X** and/or **Y** coordinates of the events. This way, the user can decide which dimensions to limit and how to limit them. Simulations stop when they reach a boundary.

If no boundary is set, the **Basic Input** pane shows the following options:

The 'Basic Input' panel shows the following settings:

- p(right)**: 1.0
- Center X**: 0.0
- Center Y**: 0.0
- #Steps**: 1
- Step Size**: Set (button)

- **p(right)** – the probability of making a right turn.
- **Center X** – x-coordinate of the center point.
- **Center Y** – y-coordinate of the center point.
- **#Steps** – number of steps to simulate.
- **Step Size** – a random variable indicating the amount of distance to cover between turns (refer to the [Probability Distribution Types](#) section for information on probability distributions and their parameters).

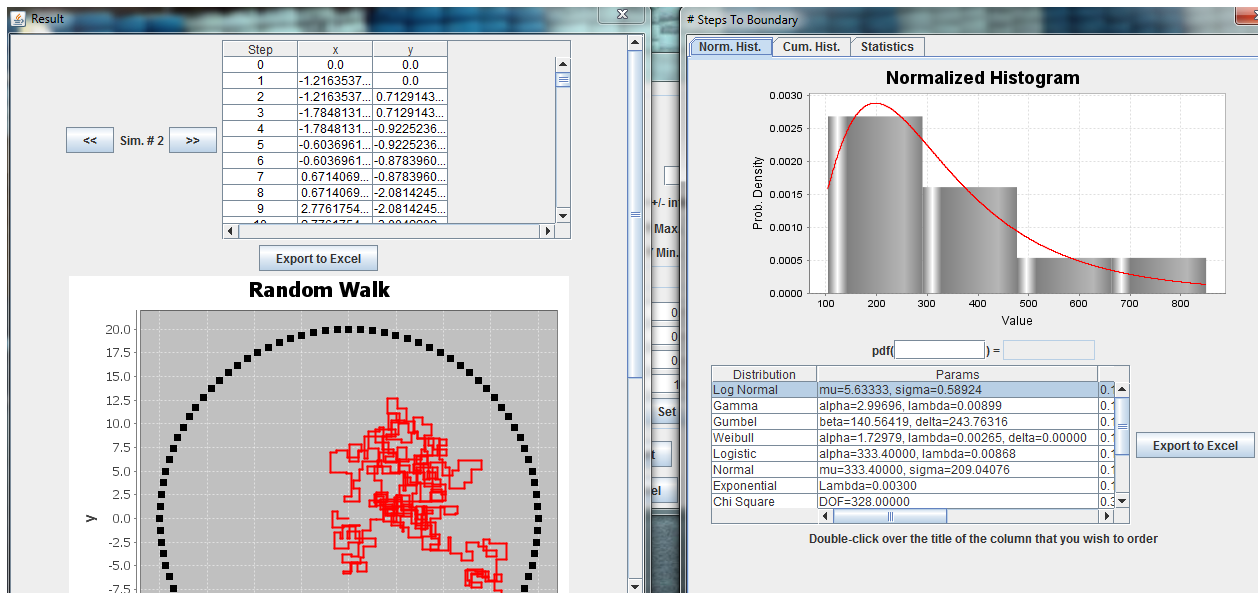
If a boundary is set, the **Basic Input** pane shows the following options:

The 'Basic Input' panel shows the following settings:

- p(right)**: 1.0
- Center X**: 0.0
- Center Y**: 0.0
- #Simulations**: 1
- Step Size**: Set (button)

- **# Simulations** – the number of random walks to carry out.
- The rest of the options are as explained above.

## OUTPUT



Two result windows are displayed after running a **BOUNDED** simulation. If no bounds have been set, only the window to the left will be shown.

On the window to the left:

- The “<<” and “>>” buttons are used to navigate between simulations. Each simulation is a full random walk.
- The table shows the x and y coordinates of each “decision point” comprising the random walk and can be easily exported to Excel.
- The “Random Walk” graph displays the entire simulation in red and the boundaries in black.

On the window to the right:

- Normalized and Cumulative histograms for the **# Steps To Boundary** measure are presented, based on all of the simulations run. Refer to the [Normalized and Cumulative Histograms](#) section for more information on how to interpret these graphs.
- Detailed statistics regarding the **# Steps To Boundary** measure can be found in the **Statistics** tab. Refer to the [Statistics](#) section for more information on how to interpret these statistics.

## RANDOM FIELDS

Random Fields are a way to model the value of a parameter **P** in a 2-dimensional space. The space is divided into sections. As explained in (Caro, Masad et al. 2011):

“Adjacent elements are more likely to have similar values of the parameter **P** than the values of elements that are further apart. Therefore, the value of **P** assigned to a given section should be somehow correlated with the values of **P** in neighbor cells. The distance at which this similarity is expected (i.e. the length of influence of the parameter **P**) is called the autocorrelation distance or correlation length of the random field, **LT**. In addition to the correlation length, a random field is also characterized by the mean value of **P** and a dispersion measure, e.g. standard deviation. The average value of **P** and its variability among different realizations always converge to the mean and standard deviation of the random field.”

2R Sim uses the method proposed by El-Kadi and Williams, which is based on **two main assumptions**: (1) the parameter of interest, **P**, has a **normal or lognormal distribution** within the space of interest and (2) P is characterized by an **exponential autocorrelation function**. The first assumption defines the process of determining probable values of P for each element of the space, whereas the second defines how the values of P at different locations are correlated (Caro, Masad et al. 2011).

For a detailed explanation of the calculations involved in the Random Field generation process, refer to (Caro, Masad et al. 2011).

## INPUT

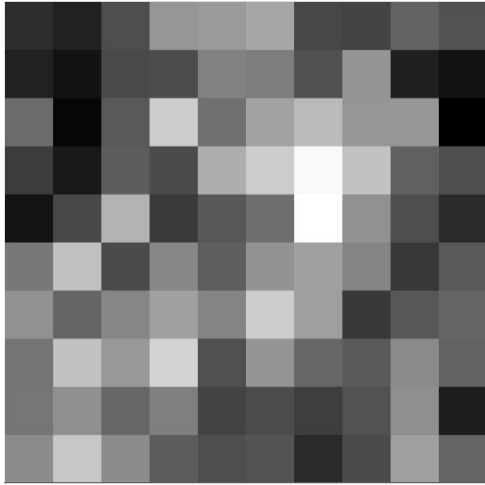
#Vertical Divs	<input type="text" value="10"/>
#Horizontal Divs	<input type="text" value="10"/>
Space Height	<input type="text" value="1.0"/>
Space Width	<input type="text" value="1.0"/>
Mu	<input type="text" value="0.0"/>
Sigma	<input type="text" value="1.0"/>
Vertical Autocorr. L	<input type="text" value="1.0"/>
Horizontal Autocorr. L	<input type="text" value="1.0"/>
# of Simulations	<input type="text" value="1"/>

- **#Vertical Divs** – the number of vertical divisions in which to divide the space.
- **#Horizontal Divs** – the number of horizontal divisions in which to divide the space.
- **Space Height** – the real-life height of the space being modeled (in meters, feet, or any other relevant measurement units).
- **Space Width** – the real-life width of the space being modeled. **Unit consistency is required. For this reason, the user is expected to employ the same measurement units used to define the Space Height.**
- **Mu** – the mean value of the parameter of interest, P.
- **Sigma** – the standard deviation of the parameter of interest, P.
- **Vertical Autocorr. L** – autocorrelation length in the vertical direction. **Unit consistency is required. For this reason, the user is expected to employ the same measurement units used to define the Space Height.**
- **Horizontal Autocorr. L** – autocorrelation length in the horizontal direction. **Unit consistency is required. For this reason, the user is expected to employ the same measurement units used to define the Space Width.**
- **# of Simulations** – the number of random fields to be generated.





A graphical representation of the random field in tab X of the Excel file can be found in the image with the name **X.jpg** located inside the output folder:



A **DARKER** section indicates a **LOWER** value of parameter **P**.

## EXAMPLES

### EXAMPLE 1 – MARKETING A NEW PRODUCT

A company wants to know how profitable it will be to market its new product, taking into account that there are many uncertainties associated with market size, expenses, and revenue. The governing equation is:

$$Profit = Income - Expenses$$

The income from the marketing effort is given by the **number of sales (S)** multiplied by the **profit per sale (P)**. The number of sales resulting from the marketing campaign is equal to the **number of monthly leads (L)** multiplied by the **conversion rate (R)**, where the conversion rate indicates the percentage of leads that result in a sale. Thus:

$$Income = L \times R \times P$$

The expenses can be divided into two types of costs: **fixed costs (K)** and **variable costs (V)**. Unlike fixed costs, variable costs depend on the **number of leads per month (L)** and the **cost of a single lead (C)** that is charged by the marketing company. As a result:

$$Expenses = K + L \times C$$

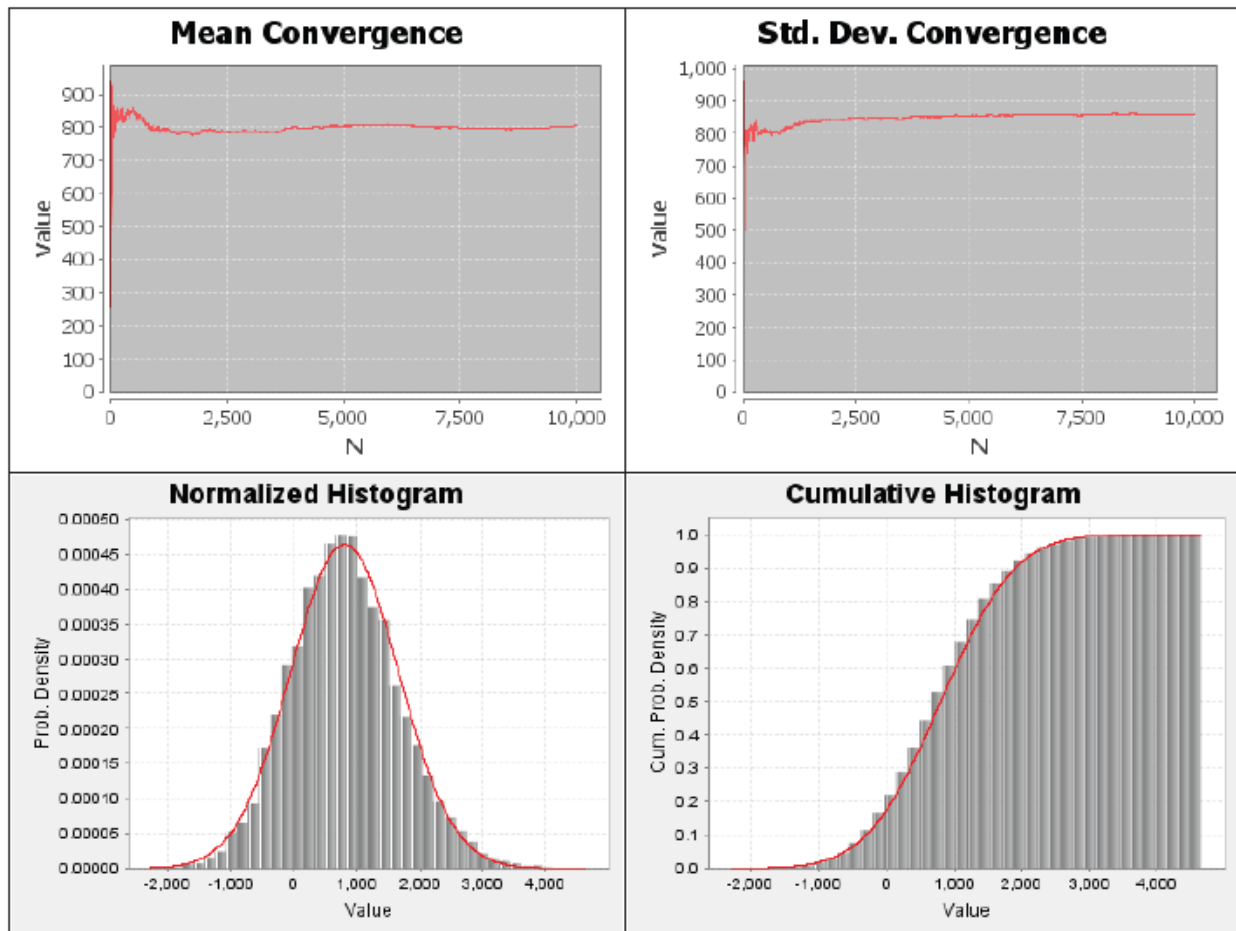
Consequently, the model to be run is expressed by:

$$Profit = L \times R \times P - (K + L \times C)$$

A market research and previous data have led to the following expected distributions for the 5 different variables of interest:

Variable	Distribution	Explanation
L	Uniform with 1200 as a minimum and 1800 as a maximum	1200 is guaranteed by the marketing company, while 1800 is the most optimistic prediction from the market research
R	Normal with mean of 3% and standard deviation of 1%	Drawn from previous marketing efforts
P	The profit per sale follows a triangular distribution with \$50 as its mode, \$47 as its minimum, and \$60 as its maximum	While the company has a margin of \$60 when the client uses cash as a payment method, the most popular payment method is credit card, which results in profits of \$50 per unit. Lastly, there are payments by check, which yield a unit profit of \$47.
K	Deterministic value of \$800	Value specified in the contract with the marketing company.
C	Uniform between \$0.20 and \$0.80	The cost of obtaining a lead depends on the medium used (internet, phone, etc) and on the hour of the day (extra hours or normal labor hours).

Running the Example 1 model 10,000 times in simple Monte Carlo mode:

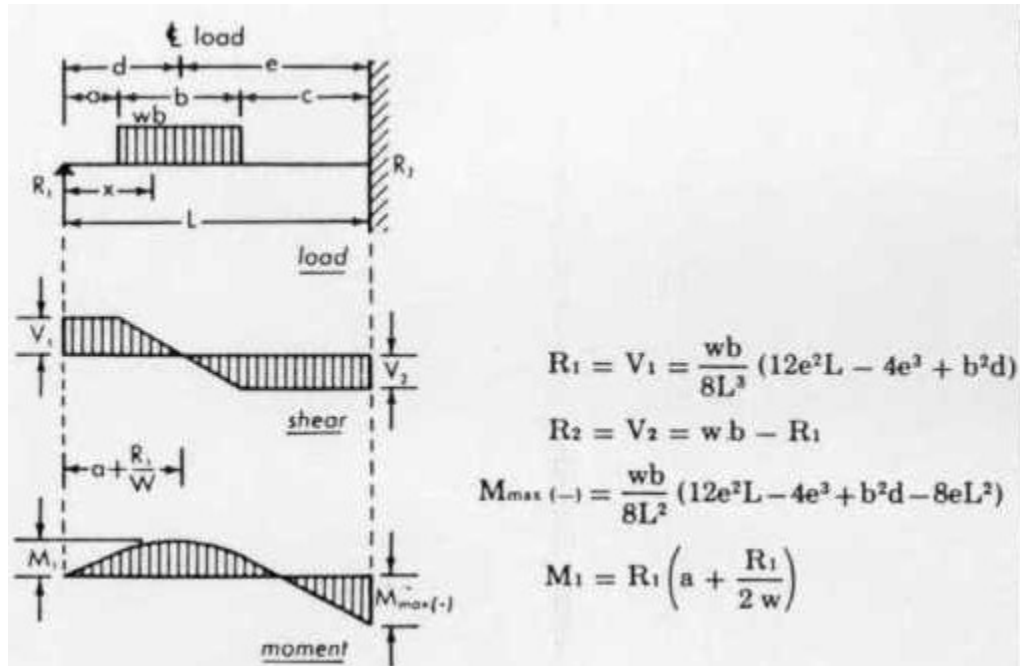


The Mean and Standard Deviation convergence graphs show that the simulation does indeed converge to specific values, indicating that the information of the normalized and cumulative histograms can be used for further analysis. According to the Anderson-Darling goodness-of-fit statistic, the best probability distribution for the **non-deterministic monthly profit** obtained through the marketing of the company's new product is a **Normal distribution with mean \$804.03 and standard deviation \$854.73**.

From the cumulative histogram, one may see that there is approximately a 16% chance of losing money by carrying out the marketing campaign throughout a month. Also, the expected return on investment is approximately 52% (\$804.03 from an average investment [or expenses] of \$1550). Thus, depending on the company's risk perception, the project may be considered attractive or not.

## EXAMPLE 2 – DESIGN MOMENT AND SHEAR STRESS FOR A BEAM

A beam is to be designed to withstand the following loads with 95% reliability:

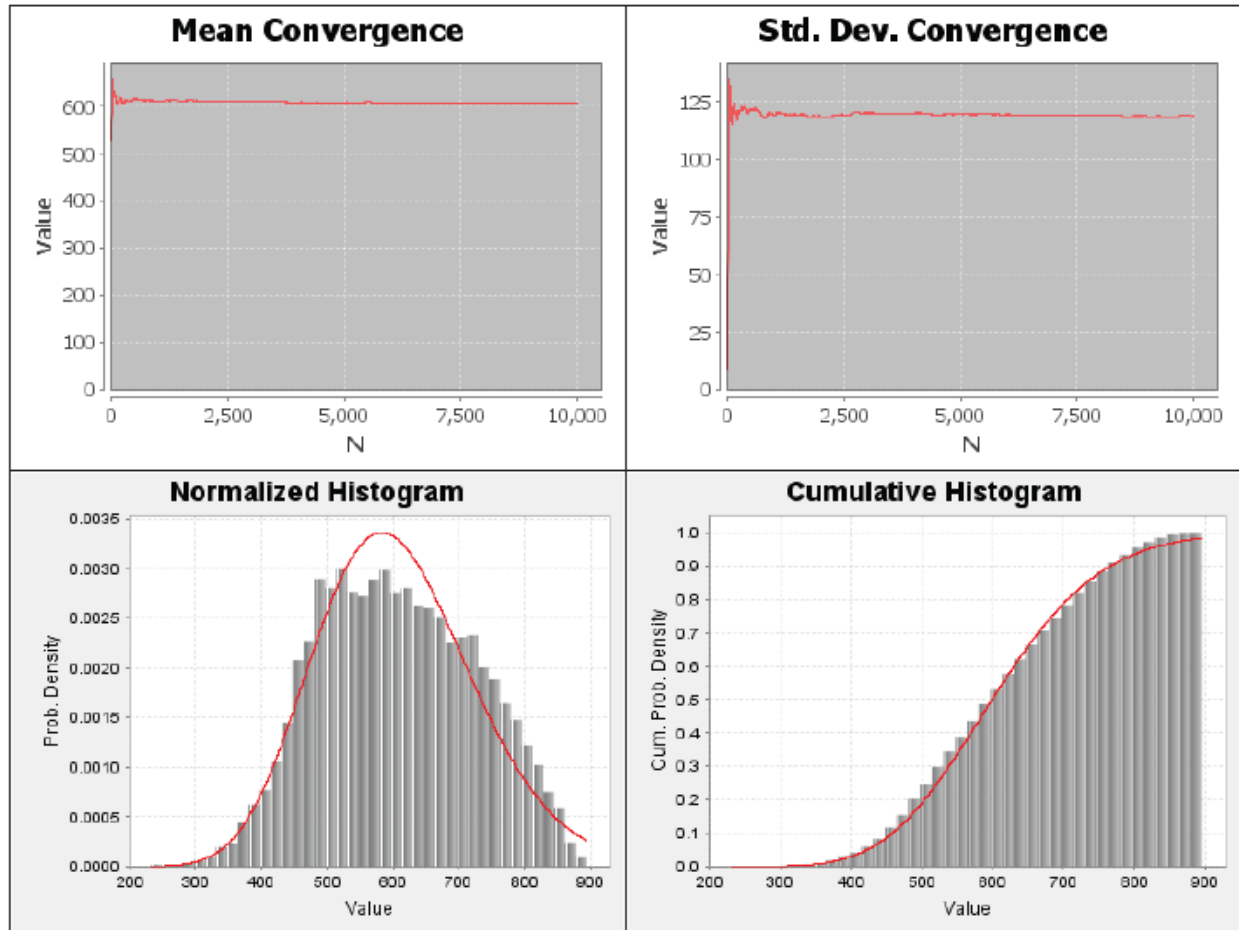


Taken from (Sprecece)

The **maximum moment** for the design needs to be determined when design parameters follow known distributions:

Variable	Distribution	Explanation
<b>w</b>	Uniform with a minimum of 10 kN/m (1 ton) and maximum of 18 kN/m	The expected dead load is 10 kN/m, while the maximum expected live load is 8 kN/m.
<b>b</b>	Beta distribution with an alpha of 5, beta of 3, lower bound of 5 meters, and upper bound of 20 meters	The lower and upper bounds come from the dimensions of the beam itself and from the objects to be placed on the beam, while the alpha and beta parameters are such that the distribution is skewed to the right.
<b>L</b>	Deterministic value of 20 meters	Given in the architectural designs.
<b>e</b>	Normal with mean of 10 meters and standard deviation of 1 meter.	The loads tend to be located on the center of the beam.
<b>d</b>	It is actually L-e	No distribution is needed for this variable, as it can be calculated from L and e.

Running the Example 2 model 10,000 times in simple Monte Carlo mode:



The Mean and Standard Deviation convergence graphs show that the simulation does indeed converge to specific values, indicating that the information of the normalized and cumulative histograms can be used for further analysis. According to the Anderson-Darling goodness-of-fit statistic, the best probability distribution for the **maximum moment on the beam** resulting from the variable loads to be sustained is a **Gamma distribution with an alpha of 25.23 and lambda of 0.04, where values are expressed in kN\*m.**

According to the cumulative histogram, the **design moment** should be approximately 825 kN\*m to achieve a 95% reliability.

## BIBLIOGRAPHY

Annis, C. "Goodness-of-Fit tests for Statistical Distributions." Retrieved 05/17/2010, from <http://www.statisticalengineering.com/goodness.htm>.

Caro, S., E. Masad, et al. (2011). "Stochastic micromechanical model of the deterioration of asphalt mixtures subject to moisture diffusion processes." International Journal for Numerical and Analytical Methods in Geomechanics **35**(10): 1079-1097.

EncyclopediaOfStatistics. "Quartiles." Retrieved 05/17/2010, from <http://books.google.com/books?id=56LkkdZPpyoC&pg=PT19&lpg=PT19>.

ERI. "Kolmogorov-Smirnov Test." Retrieved 05/17/2010, from <http://www.eridlc.com/onlinetextbook/index.cfm?fuseaction=textbook.appendix&FileName=Table7>.

Lane2, D. "Standard Deviation and Variance." Retrieved 05/17/2010, from <http://davidmlane.com/hyperstat/A16252.html>.

Lane, D. "Computing Pearson's Correlation Coefficient." Retrieved 05/17/2010, from <http://davidmlane.com/hyperstat/A51911.html>.

MathsRevision.net. "Box and Whisker Diagrams." Retrieved 05/17/2010, from <http://www.mathsrevision.net/alevel/pages.php?page=50>.

Microsystems, S. "Math JAVA J2SE." Retrieved 05/17/2010, from <http://java.sun.com/j2se/1.4.2/docs/api/java/lang/Math.html>.

MSDN, M. "Modulus Operator (%)." Retrieved 05/17/2010, from [http://msdn.microsoft.com/en-us/library/h6zfy7\(VS.80\).aspx](http://msdn.microsoft.com/en-us/library/h6zfy7(VS.80).aspx).

ReliaSoftCorp. "Critical Values for Cramér-von Mises Test." Retrieved 05/17/2010, from [http://www.weibull.com/RelGrowthWeb/Appendix\\_B\\_Critical\\_Values\\_for\\_Cramer-von\\_Mises\\_Test.htm](http://www.weibull.com/RelGrowthWeb/Appendix_B_Critical_Values_for_Cramer-von_Mises_Test.htm).

SEMATECH1, N. "Kolmogorov-Smirnov Goodness-of-Fit Test." Retrieved 05/17/2010, from <http://www.itl.nist.gov/div898/handbook/eda/section3/eda35g.htm>.

SEMATECH2, N. "Anderson-Darling Test." Retrieved 05/17/2010, from <http://www.itl.nist.gov/div898/handbook/eda/section3/eda35e.htm>.

SEMATECH3, N. "Measures of Skewness and Kurtosis." Retrieved 05/17/2010, from <http://www.itl.nist.gov/div898/handbook/eda/section3/eda35b.htm>.

Simard, R. "Package umontreal.iro.lecuyer.probdist." Retrieved 05/17/2010, from <http://www.iro.umontreal.ca/~simardr/ssj/doc/html/umontreal/iro/lecuyer/probdist/package-summary.html>.

Spreccace, A. "Beam Load Equations." Retrieved 05/17/2010, from <http://www.spreccace.com/node/43>.

Weisstein2, E. W. "Arithmetic Mean." Retrieved 05/17/2010, from <http://mathworld.wolfram.com/ArithmeticMean.html>.

Weisstein3, E. W. "Variance." Retrieved 05/17/2010, from <http://mathworld.wolfram.com/Variance.html>.

Weisstein, E. W. "Standard Deviation." Retrieved 05/17/2010, from <http://mathworld.wolfram.com/StandardDeviation.html>.