Ersatz

Ersatz Function Overview

version 1.35

Jan J Barendregt

Web: www.epigear.com
Email: info@epigear.com

Ersatz

Contents

Introduction

This *Ersatz Function Overview* gives a line-up of the functions that the Ersatz add-in adds to Excel, including a description of what the function does and its parameters, if any. For a more general introduction to the Ersatz software package, please consult the *Ersatz User Guide*.

Ersatz supports the Excel Function Wizard to guide you in your choice of Ersatz functions and their parameters. For help on how to use the Function Wizard, please consult the Excel Help. The Ersatz functions are all listed in the 'Ersatz' category of the Wizard. Unfortunately, Excel does not support tool-tip help on the parameters of add-in functions: this is only implemented for native Excel functions.

For this *Overview* I have grouped the Ersatz functions, whose names all start with 'Er', under five headings. The *Standard distribution functions* are mostly functions that return random deviates from the well-known distributions from any statistical textbook, such as the Normal and Gamma. Under the heading *Component functions* a number of functions are discussed that require groups of functions to collaborate. *Multivariate correlated random functions* lists the correlated Dirichlet and multinomial functions. Then there is a section on *Output and sensitivity input functions*, one on some non-random *Special functions*, and finally one on *Statistical functions*.

Most functions take parameters, and many of them put specific requirements on these parameters. For example, the Normal distribution requires that the standard deviation is greater than zero. When the requirements for the parameters are not met, all Ersatz functions will return the #NUM! error. Ersatz includes a distribution viewer, that displays a graph of the chosen distribution for given parameters.

Please note that Excel enforces the use of brackets for all functions, including those that take no parameters. For example, the ErUniform01 function, that returns a uniformly distributed deviate between 0 and 1, has to be entered as 'ErUniform01()'. If the brackets are omitted, Excel returns a meaningless large number instead of the random deviate.

Standard distribution functions

ErBernoulli
ErBeta
ErBinomial
ErChi2
ErDelaporte
ErDice (discussed under the heading 'Uniform')
ErEmpirical
ErExponential
ErGamma
ErGeometric
ErHypergeometric
ErInverseGaussian
ErLogistic
ErLoglogistic
ErLognormal
ErLognormal2
ErNegbinomial
ErNormal

ErPareto
ErPearson5
ErPearson6
ErPert
ErPoisson
ErPolya
ErRayleigh
ErRelativeRisk
ErSurvival
ErSurvival2
ErTriangle
ErUniform01
ErUniform
ErWeibull

Component random functions

ErNonparam & ErNonparamCom
ErRandomise & ErRandomiseCom
ErMultinomial & ErMultinomialCom
ErDirichlet & ErDirichletCom
ErCorrNormal & ErCorrNormalCom
ErRankCorr & ErRankCorrCom

Multivariate correlated random functions

ErDirichletCorr, ErDirichletCorrIn, & ErDirichletCorrOut
ErMultinomialCorr, ErMultinomialCorrIn, ErMultinomialCorrOut

Output and sensitivity input functions

ErOutput
ErRunOutput
ErSensInput
ErRunSensInput
ErData
ErDataArray
ErRunDataArray

Special functions

ErMinimize & ErMinimizeResult
ErOneOffCostDisc
ErAnnualCostDisc
ErLYDisc
ErHALYDisc
ErFixed
ErConditional
ErTruncate
ErIteration
ErSetItno
ErRunno
ErRunning

ErVersion
ErStore
ErRetrieve
ErStoreArray
ErRetrieveArray
ErCondStoreArray
ErCondRetrieveArray
ErSortedArray

Statistical functions

ErMean
ErStDev
ErMin
ErMax
ErCorrelation
ErPercentile

A note on Excel (and Ersatz) array functions

Some functions in Excel and some of the Ersatz functions do not return a single value but a whole array of values. To accommodate an array of values, the function output needs an Excel range instead of a single cell. To get the function to put the output in an Excel range, it must be entered as a so-called 'array formula'.
See the Excel Help on array formulas. Briefly: select the range of cells you want the output range to appear in, type your formula, and then press CTRL-Shift-Enter. This will result (if all went well) in your formula appearing in all the selected cells, embedded in curly braces.
In order to do this, you of course need to know the size of the function output array. In the function descriptions below, we indicate which functions are to be entered as array functions, and what the size of the output range should be. Ersatz functions that need to be entered as array functions are ErMinimize, ErDataArray, ErRunDataArray, ErRetrieveArray, ErCondRetrieveArray, ErDirichletCorrOut, and ErMultinomialCorrOut.

Standard distribution functions

Bernoulli

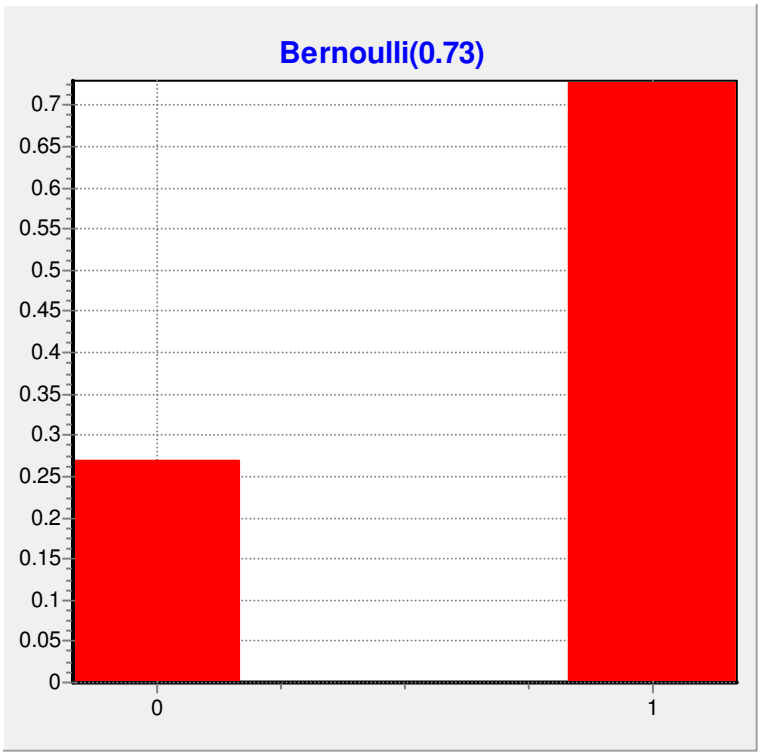
ErBernoulli(p): discrete, 0.0≤p≤1.0.

Returns a random deviate from the Bernoulli distribution.

Mean: p

.

StDev: $\sqrt{p(1-p)}$



Algorithm: generate a uniform random number between 0 and 1, if smaller than p then return 1, else return 0.
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Beta

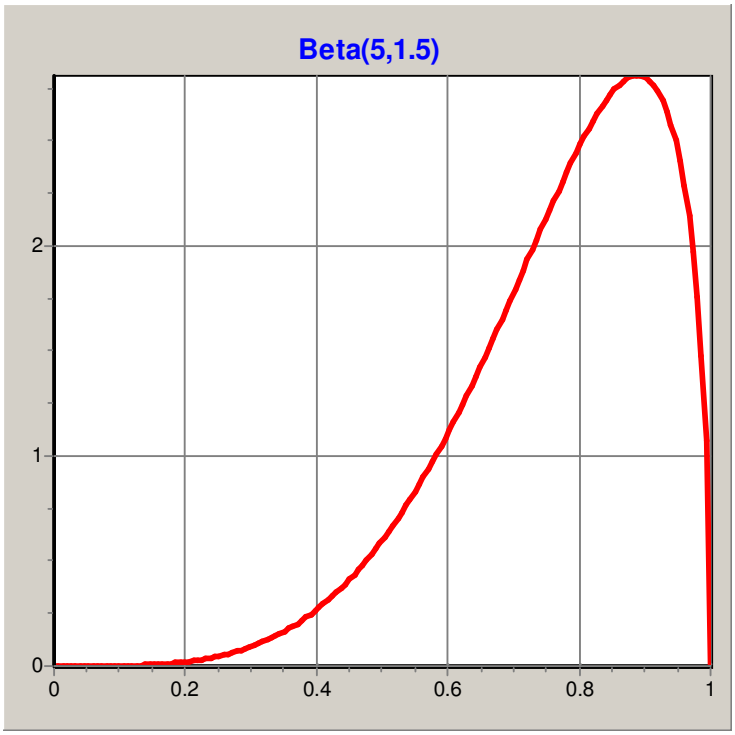
ErBeta(α1, α2): continuous, 0.0≤Beta≤1.0, α1, α2 continuous>0.

Returns a random deviate from the Beta distribution.

Mean: $\frac{\alpha 1}{\alpha 1+\alpha 2}$

.

StDev: $\sqrt{\frac{\alpha 1\alpha 2}{(\alpha 1+\alpha 2)^2(\alpha 1+\alpha 2+1)}}$



Note: the Beta distribution is a conjugate of the Binomial distribution, with parameter α1=Np and α2=N(1-p).

Algorithm based on generating two Gamma distributions (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

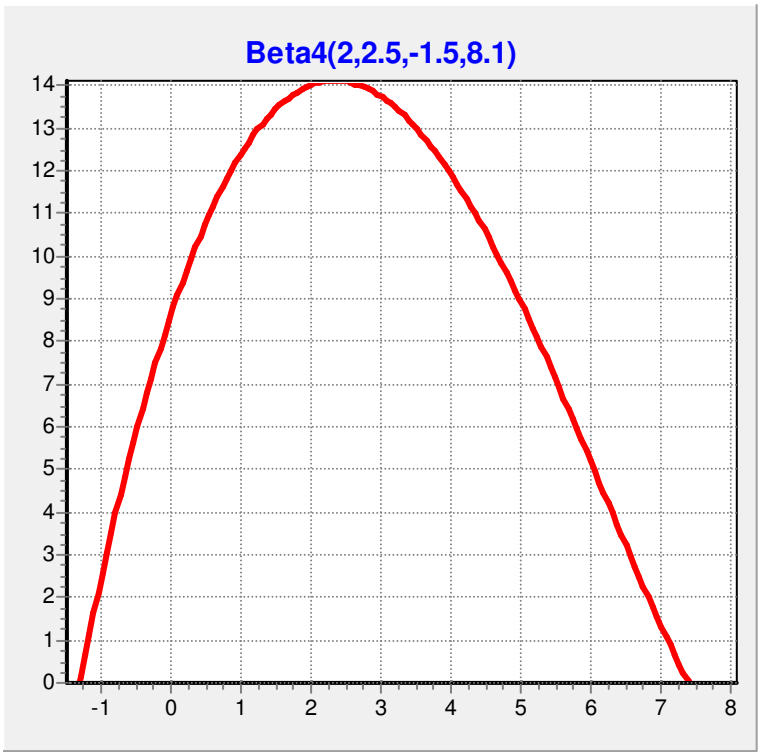
Beta4

ErBeta4(α1, α2,min,max): continuous, 0.0≤Beta≤1.0, α1, α2 continuous>0; min, max: continuous, min<max.

Returns a random deviate from a Beta distribution, rescaled between min and max.

Mean: $\min+(\max-\min)\frac{\alpha 1}{\alpha 1+\alpha 2}$

StDev: $(\max-\min)\sqrt{\frac{\alpha 1\alpha 2}{(\alpha 1+\alpha 2)^2(\alpha 1+\alpha 2+1)}}$



Note: the Beta4 distribution is a rescaled Beta function, which is useful when you need a random variable between two limits. See the *Ersatz User Guide* section on Choosing appropriate distributions on how to calculate the α1 and α2 parameters from a mean and standard deviation.

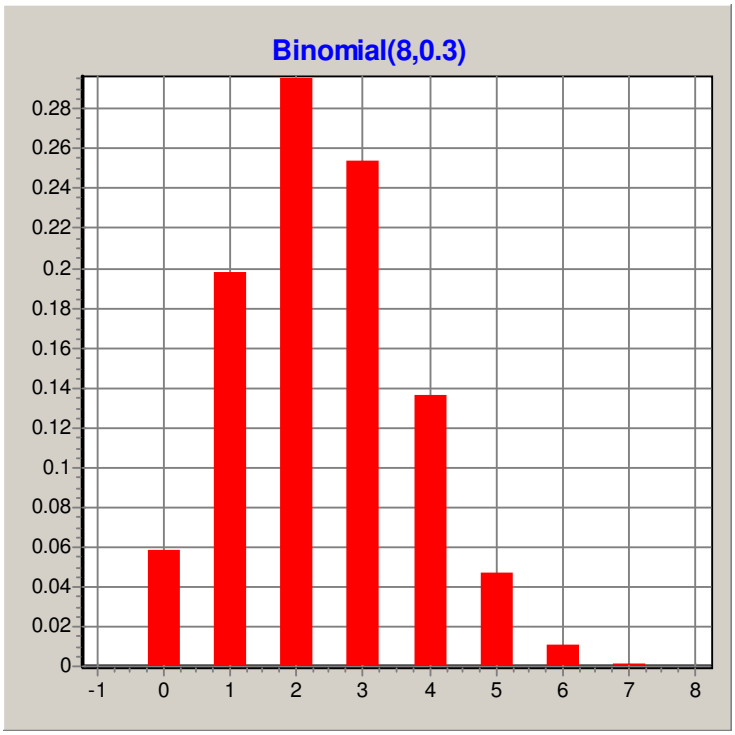
Algorithm based on generating two Gamma distributions (Law and Kelton 2000). Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Binomial

ErBinomial(N,p): discrete≥0, N discrete>0, p continuous 0.0≤p≤1.0.

Returns a random deviate from the Binomial distribution

Mean: Np
StDev: $\sqrt{Np(1-p)}$



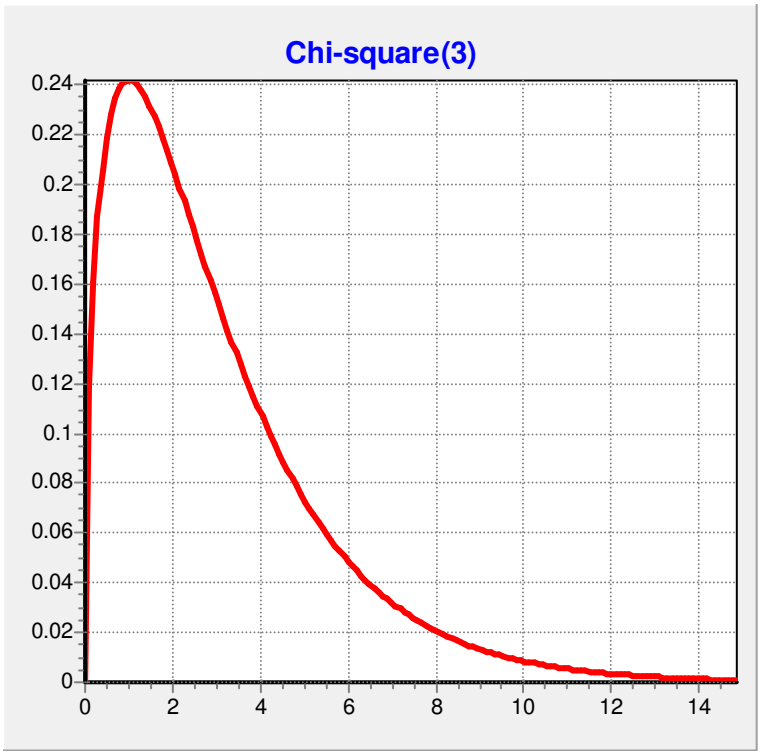
Algorithm based on (Press, Teukolsky et al. 1992).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Chi-square

ErChi2(k): continuous≥0, k discrete>0.

Returns a random deviate from the Chi Square distribution

Mean: k
StDev: $\sqrt{2k}$



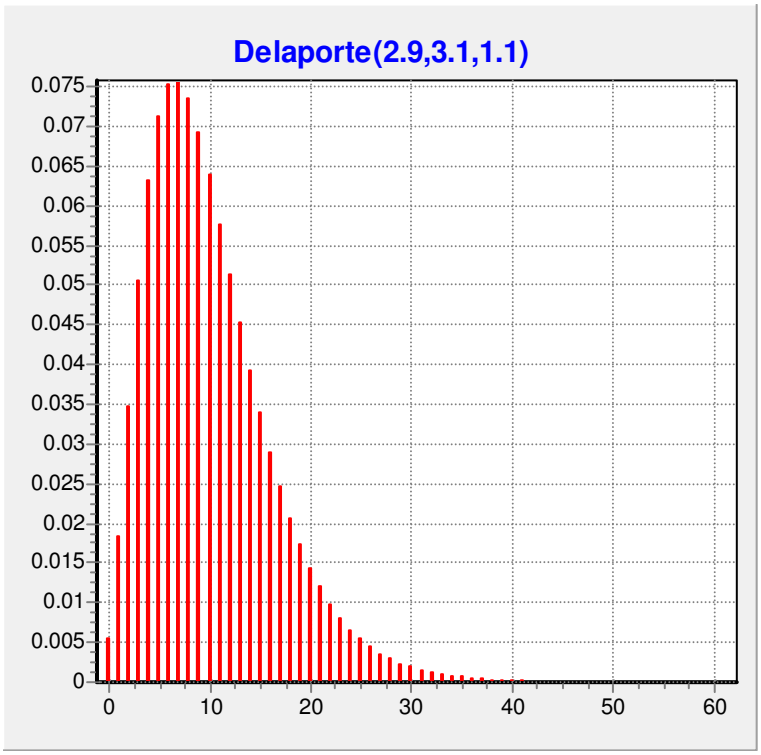
Algorithm based on (Devroye 1986).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Delaporte

ErDelaporte(α,β,γ): discrete ≥ 0, α continuous>0.0, β continuous>0.0, γ continuous>0.0.

Returns a random deviate from the Delaporte distribution.

Mean: αβ+ γ.
StDev: $\sqrt{\alpha\beta(1+\beta)+\gamma}$



Notes:

1. The Delaporte is a useful replacement for the Poisson distribution when the observed variance is larger than what the Poisson predicts. While the Poisson models events at a constant rate, the Delaporte models events at a randomly varying rate.
2. See also the Polya distribution.

Algorithm: Delaporte(α,β,γ)~Poisson(Gamma(α,β))+Poisson(γ).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Empirical

ErEmpirical("name",x,p,discrete): discrete or continuous. name: string; **x**, **p**: Excel ranges of equal length, $\sum_i p_i > 0$, $p_i \geq 0 \ \forall$ i; discrete: boolean.

ErEmpirical has two different modes, depending on the value of the 'discrete' parameter.

When discrete=TRUE this function returns a discrete value x_i from the range **x** with probability $\frac{p_i}{\sum_i p_i}$ (meaning that the values in **p** are normalised to probabilities summing up to 1).

When discrete=FALSE this function returns a continuous value from a distribution created by interpolation over the **x** range between the values given in the **p** range. This distribution is normalised to have an area under the curve equal to 1.

Notes:

1. The Excel ranges can be single or multiple column or row. If multiple, Ersatz reads the range column-wise.
2. Empty cells in the ranges are evaluated as 0.
3. Non-numerical values in the ranges cause this function to return #VALUE!.
4. ErEmpirical returns #NUM! when the ranges of **x** and **p** are of unequal length.

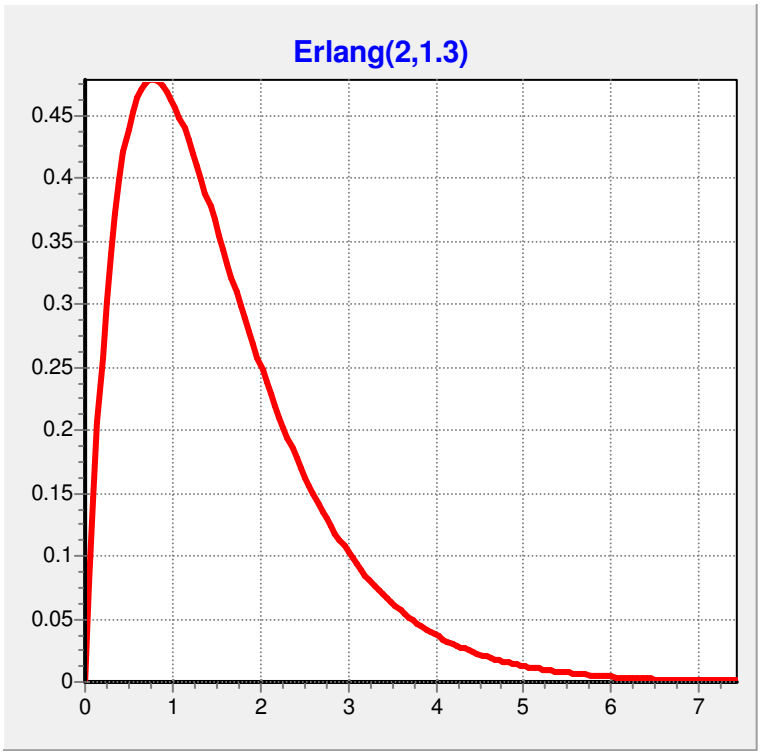
A note of warning. ErEmpirical offers the ultimate flexibility in the choice of random function: it allows to create your own distributions, based on empirical data, on combinations of parametric distributions, or both. See the Empirical workbook for some examples.
However, for the continuous variant of ErEmperical this flexibility comes with vulnerability. ErEmpirical with discrete=FALSE relies on numerical methods (in particular cubic spline interpolation and Romberg integration, see (Press, Teukolsky et al. 1992)), and it is quite possible to come up with input ranges that will cause it to go awry. In particular a flat curve (ie consecutive *p* values that are the same) and discontinuities (ie sharp bends) can cause ErEmprirical to throw a fit (returning #NUM!) or to deviate substantially from the input curve. It is therefore strongly recommended whenever you use this function in continuous mode to monitor its behaviour by putting an ErOutput function on its output.
Example: workbook 'Empirical'.

Erlang

ErErlang(m,β): continuous>0.0, m discrete>0, β continuous>0.0.

Returns a random deviate from the Erlang distribution

Mean: mβ.
StDev: $\beta\sqrt{m}$.



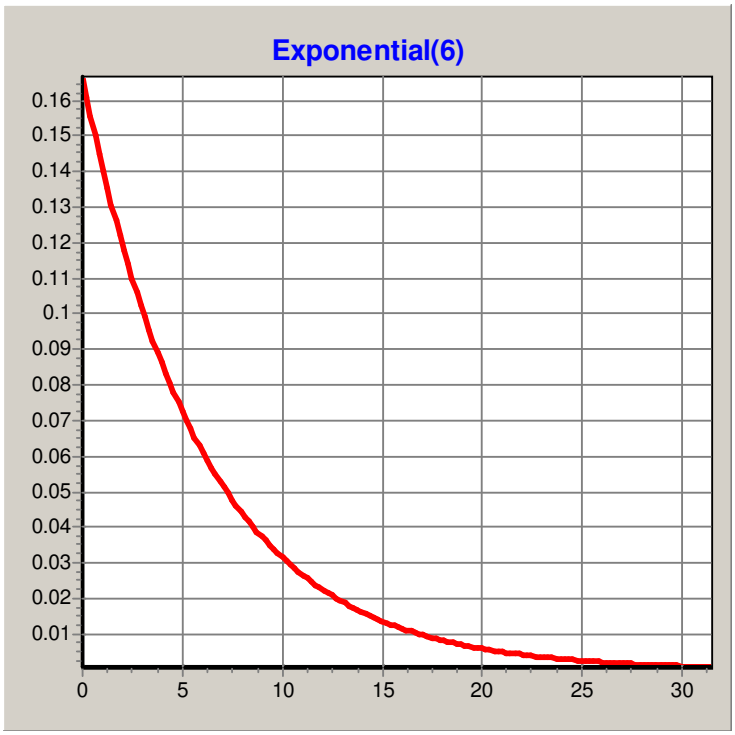
Algorithm is based on (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Exponential

ErExponential(β): continuous>0.0, β continuous>0.0.

Returns a random deviate from the Exponential distribution

Mean: β.
StDev: β.



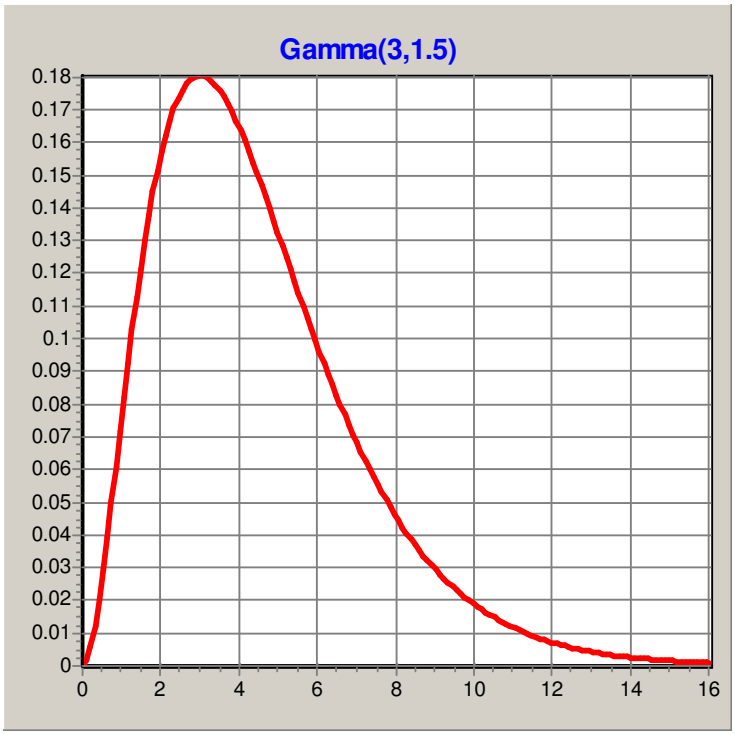
Algorithm is inverse-transform (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Gamma

ErGamma(α,β): continuous>0.0, α continuous>0.0, β continuous>0.0.

Returns a random deviate from the Gamma distribution.

Mean: αβ
StDev: $\beta\sqrt{\alpha}$



Algorithm based on the acceptance-rejection method (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

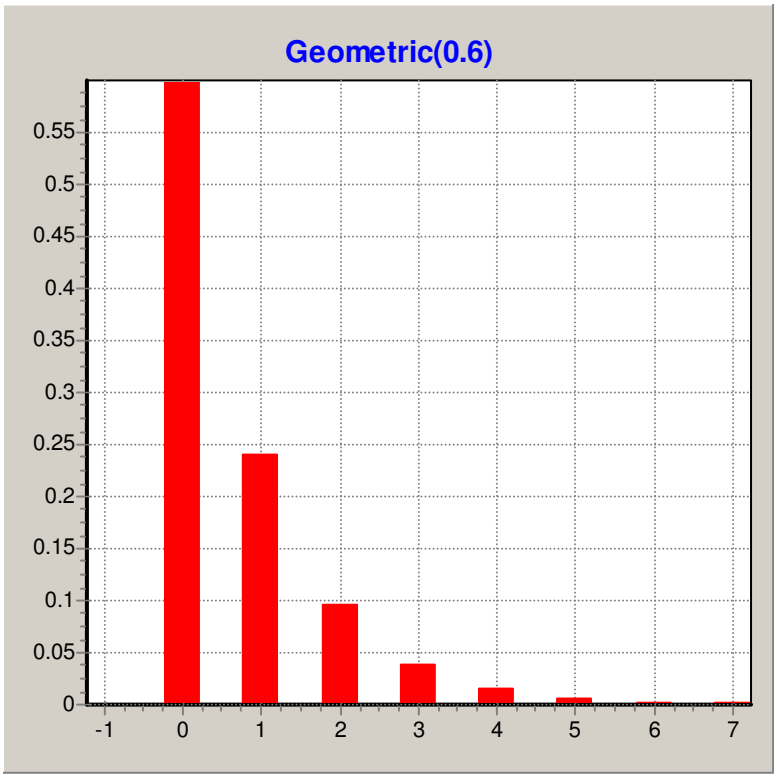
Geometric

ErGeometric(p): discrete≥0, p: continuous, 0<p≤1.

Returns a random deviate from the Geometric distribution

Mean: $\frac{1-p}{p}$

StDev: $\frac{\sqrt{1-p}}{p}$



Algorithm based on the inverse-transform method (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

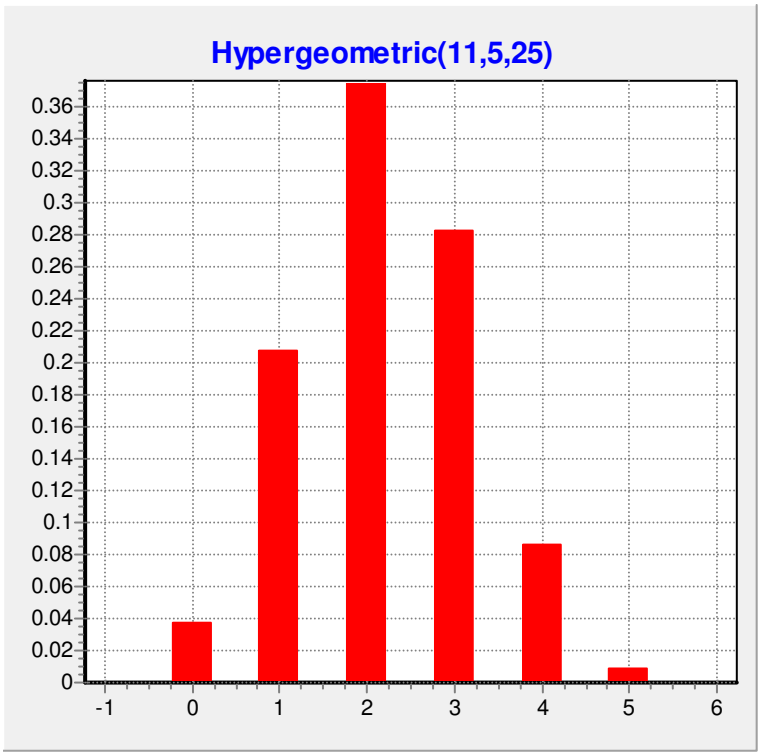
Hypergeometric

ErHypergeometric(n,D,M): discrete≥0, n: discrete 0≤n≤M, D: discrete 0≤D≤M , M: discrete ≥0.

Returns a random deviate from the Hypergeometric distribution

Mean: $\frac{nD}{M}$

StDev: $\sqrt{\frac{n(D/M)(1-D/M)(M-n)}{M-1}}$



Algorithm based on (Kachitvichyanukul and Schmeiser).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

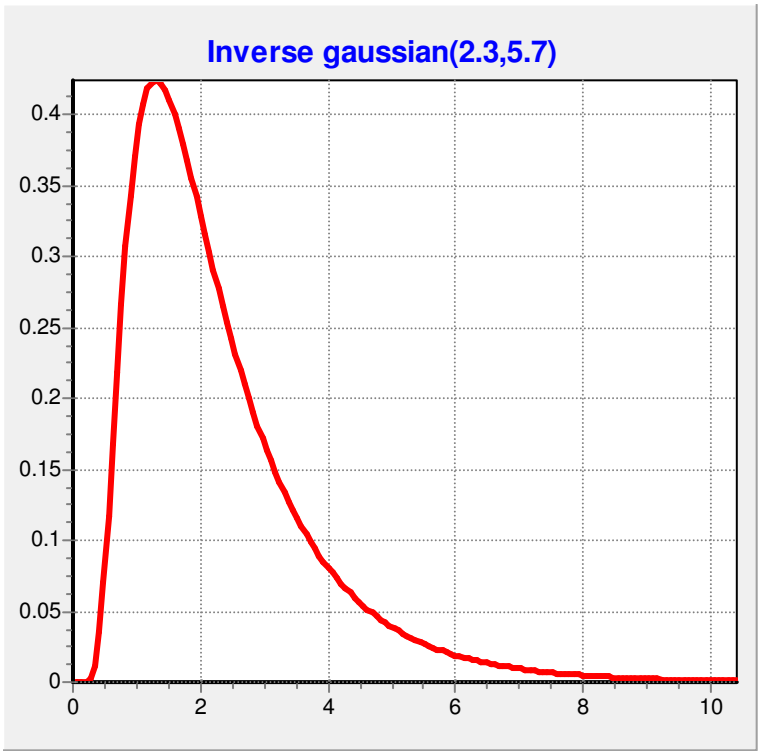
Inverse Gaussian

ErInverseGaussian(μ,λ): continuous, μ continuous, λ continuous>0.0.

Mean: μ

StDev: $\sqrt{\frac{\mu^3}{\lambda}}$

Returns a random deviate from the Inverse Gaussian distribution.



Algorithm based on (Gentle 2003).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

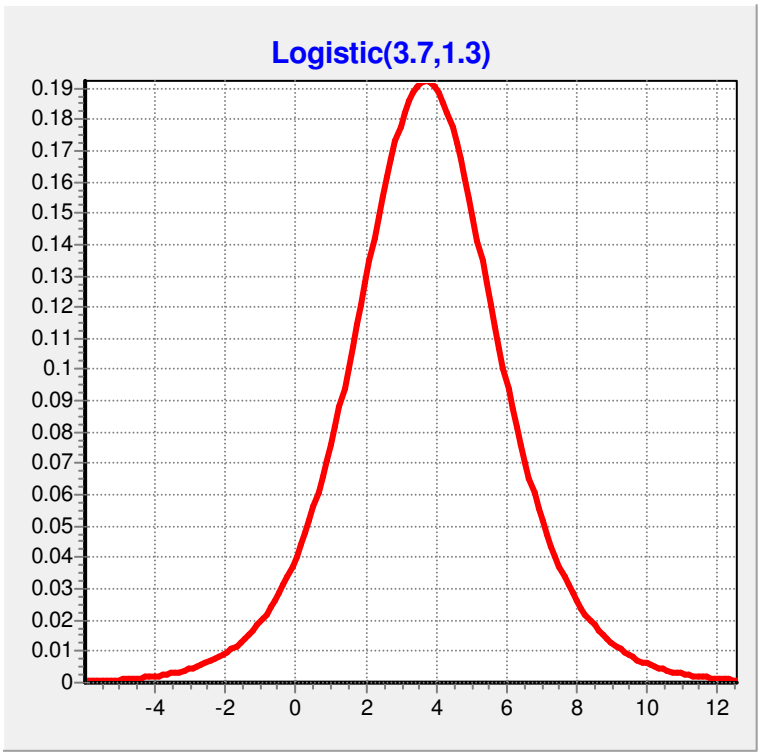
Logistic

ErLogistic(μ,s): continuous, μ continuous, s continuous>0.0.

Mean: μ

StDev: $\pi\, s\sqrt{\frac{1}{3}}$

Returns a random deviate from the Logistic distribution.



Algorithm based on inverse CDF method.
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

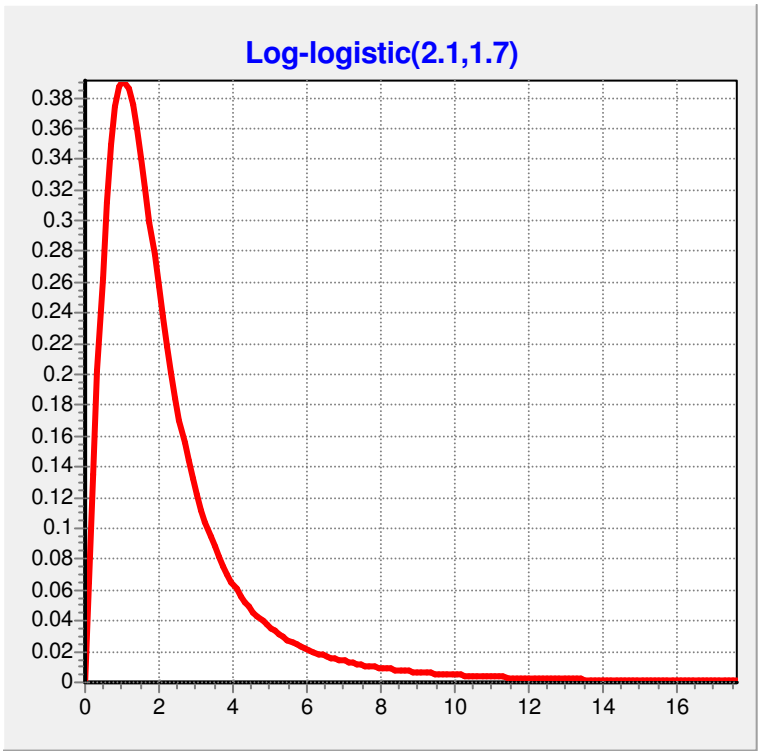
Log-logistic

ErLoglogistic(α,β): continuous>0.0; α, β continuous>0.0.

Mean: $\frac{\beta\frac{\pi}{\alpha}}{\sin\left(\frac{\pi}{\alpha}\right)}$

StDev: $\beta\sqrt{\frac{\pi}{\alpha}\left(\frac{2}{\sin\left(2\frac{\pi}{\alpha}\right)}\right)-\frac{\pi}{\alpha}\left(\frac{\alpha}{\pi}\right)^2}$ if α>2

Returns a random deviate from the Log-logistic distribution.



Algorithm based on (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Lognormal

The Lognormal function comes in two different parametrisations, but they are otherwise identical.

ErLognormal(mean,std): continuous, mean continuous>0.0, std continuous>0.0.

Mean: mean
StDev: std

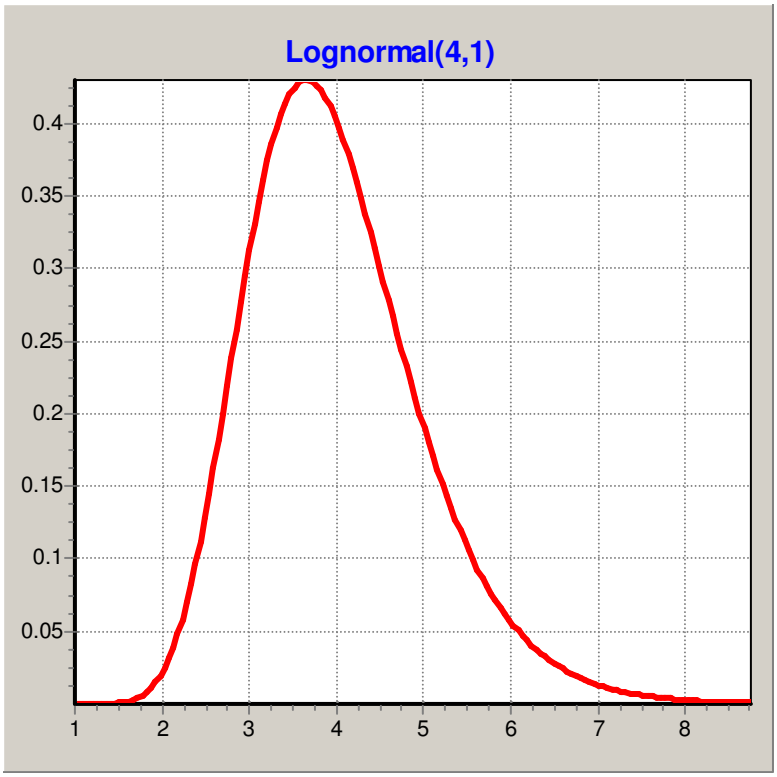
ErLognormal2(μ,σ): continuous, μ continuous, σ continuous>0.0.

Mean: $\exp\left(\mu+\frac{1}{2}\sigma^2\right)$

StDev: $\sqrt{\exp(2\mu+2\sigma^2)-\exp(2\mu+\sigma^2)}$

Returns a random deviate from the Lognormal distribution.

Note: Excel's LOGNORMDIST function uses the same parametrisation as ErLognormal2.



Algorithm based on (Press, Teukolsky et al. 1992).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

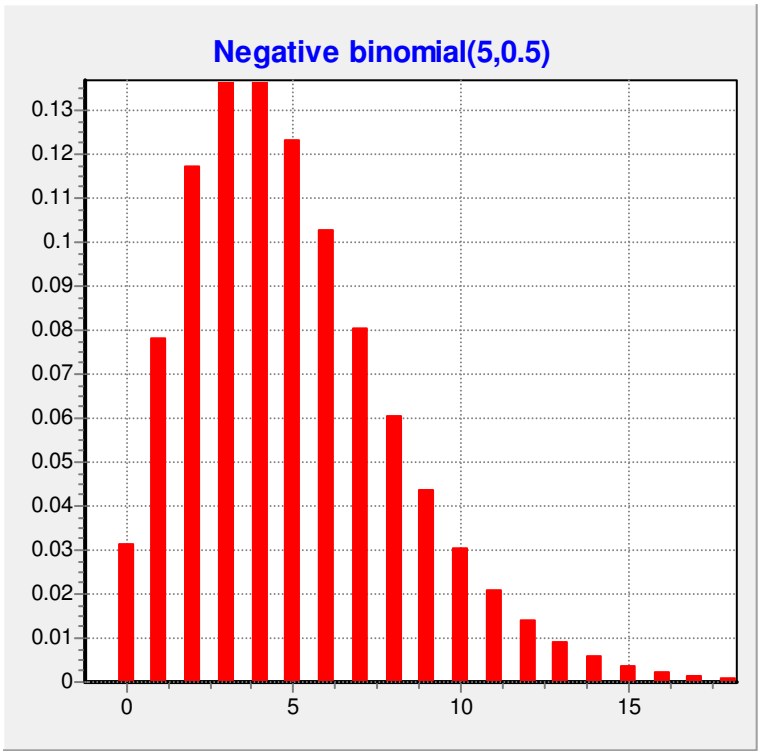
Negative binomial

ErNegbinomial(s,p): discrete≥0, s discrete>0, p continuous 0.0≤p≤1.0.

Returns a random deviate from the Negative binomial distribution.

Mean: $\frac{s(1-p)}{p}$

StDev: $\frac{\sqrt{s(1-p)}}{p}$



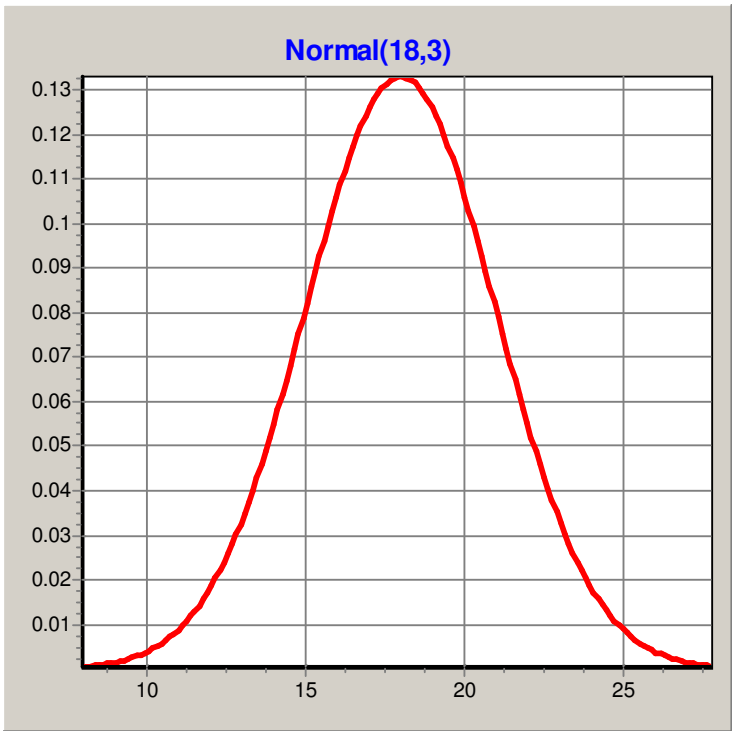
Based on a convolution algorithm (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Normal

ErNormal(μ,σ): continuous, μ continuous, σ continuous>0.0.

Returns a random deviate from the Normal distribution

Mean: μ.
StDev: σ.



Algorithm based on the ratio of uniforms method (Press, Teukolsky et al. 2007).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

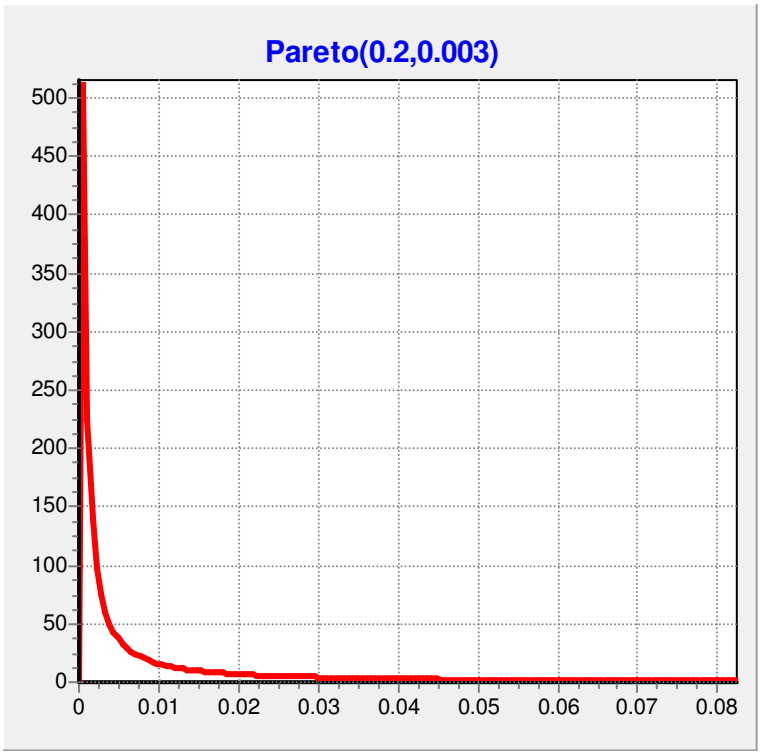
Pareto

ErPareto(a,b): continuous>0.0; a, b continuous>0.0.

Returns a random deviate from the Pareto distribution

Mean: $\frac{ab}{a-1}$, if a>1.

StDev: $\sqrt{\frac{ab^2}{(a-2)(a-1)^2}}$, if a>2.



Algorithm based on (Gentle 2003).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

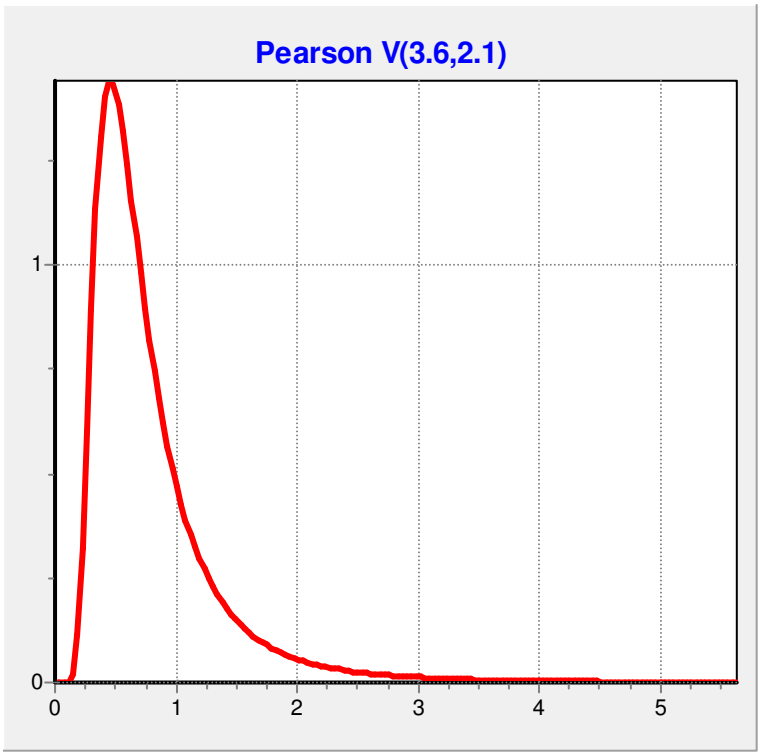
Pearson V

ErPearson5(α,β): continuous>0.0; α, β continuous>0.0.

Returns a random deviate from the Pearson V distribution

Mean: $\frac{\beta}{\alpha-1}$, if α>1.

StDev: $\frac{\beta}{(\alpha-1)\sqrt{\alpha-2}}$, if α>2.



Algorithm based on (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

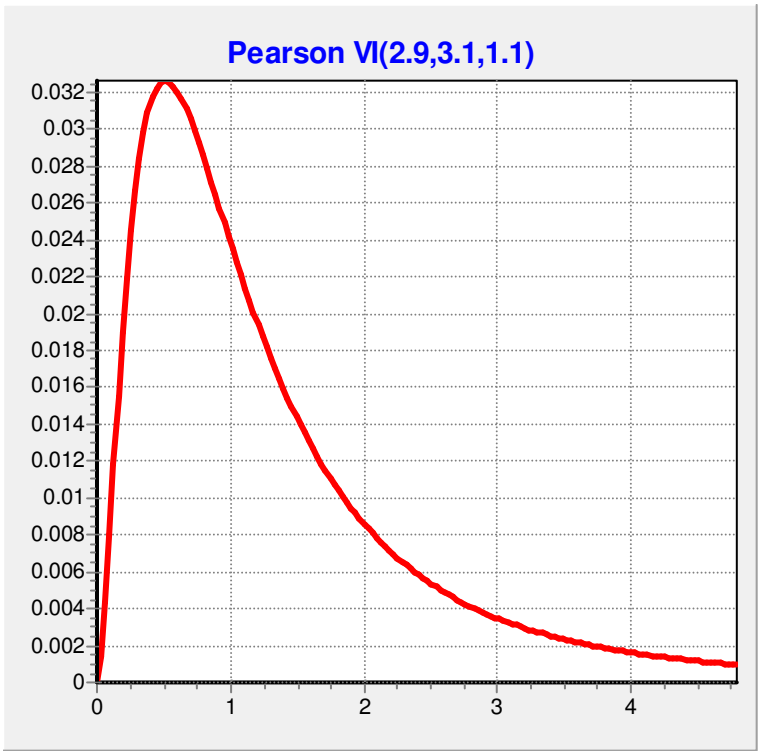
Pearson VI

ErPearson6(α1,α2,β): continuous, α1, α2, β continuous>0.0.

Returns a random deviate from the Pearson VI distribution

Mean: $\frac{\beta\alpha 1}{\alpha 2-1}$, if α2>1.

StDev: $\left(\frac{\beta}{\alpha 2-1}\right)\sqrt{\frac{\alpha 1+\alpha 2-1}{\alpha 2-2}}$, if α2>2.



Algorithm based on (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

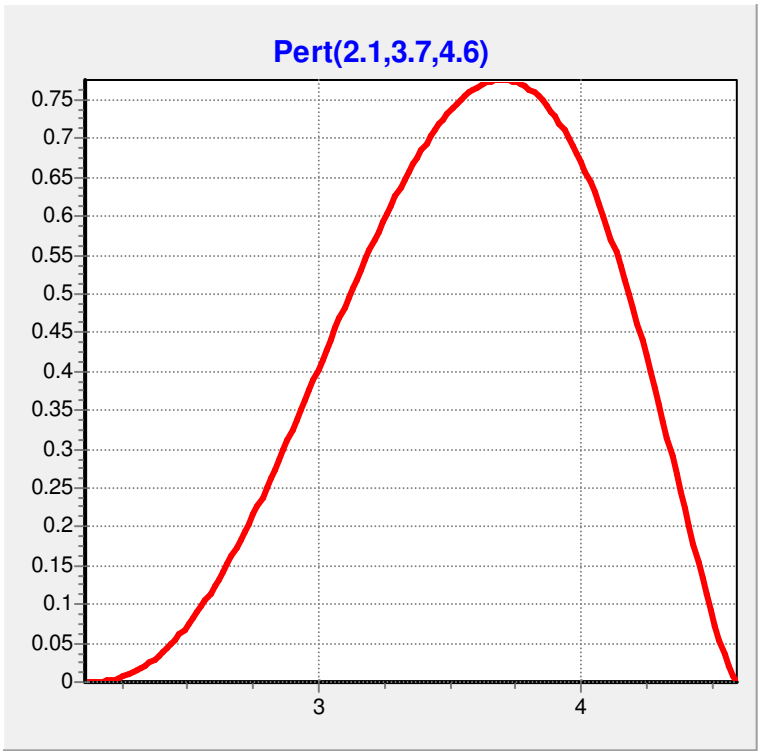
Pert

ErPert(a,m,b): continuous, a≤x≤b; a, m, b continuous; a≤m≤b, a<b.

Returns a random deviate from the Beta Pert distribution.

Mean: $a+\left(\frac{4m+b-5a}{6}\right)$

StDev: $(b-a)\sqrt{\frac{(4m+b-5a)(5b-a-4m)}{7(6b-6a)^2}}$



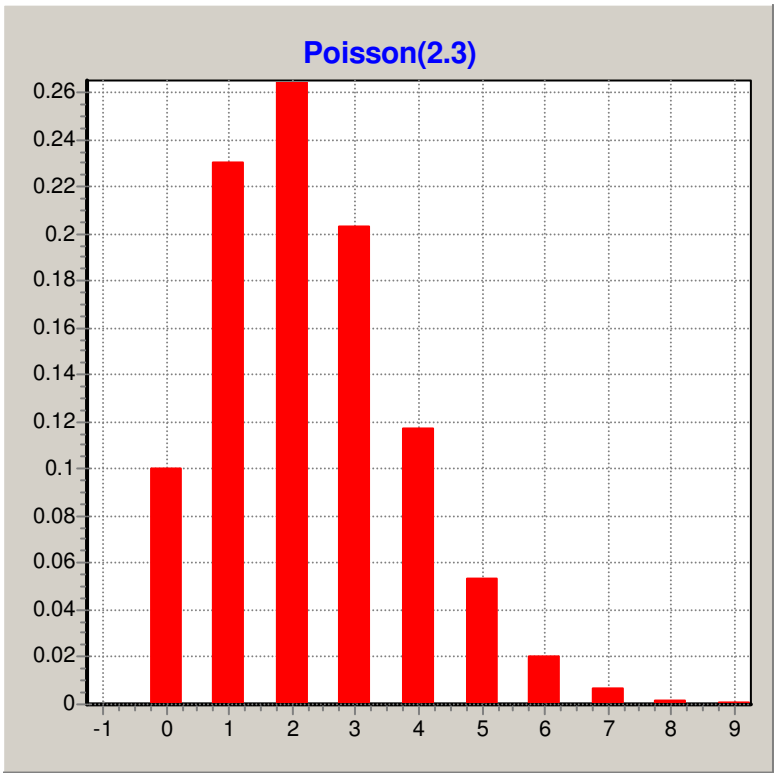
Note: the Pert distribution is a se-scaled and re-parametrised version of the Beta distribution. It returns a random deviate from a distribution between a and b with mode m.
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Poisson

ErPoisson(ν): discrete ≥ 0, ν continuous >0.0.

Returns a random deviate from the Poisson distribution.

Mean: ν.
StDev: $\sqrt{\nu}$



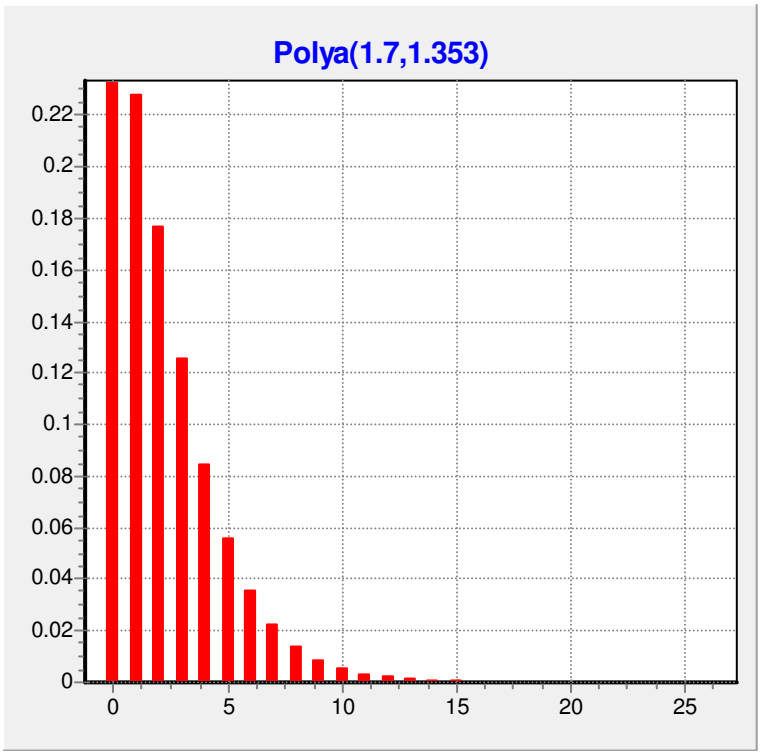
Algorithm based on (Press, Teukolsky et al. 1992).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Polya

ErPolya(α,β): discrete ≥ 0, α continuous>0.0, β continuous>0.0.

Returns a random deviate from the Polya distribution.

Mean: αβ.
StDev: $\sqrt{\alpha\beta(1+\beta)}$



Notes:

1. The Polya is a useful replacement for the Poisson distribution when the observed variance is larger than what the Poisson predicts. While the Poisson models events at a constant rate, the Polya models events at a randomly varying rate.
2. See also the Delaporte distribution.

Algorithm: Polya(α,β)~Poisson(Gamma(α,β)).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

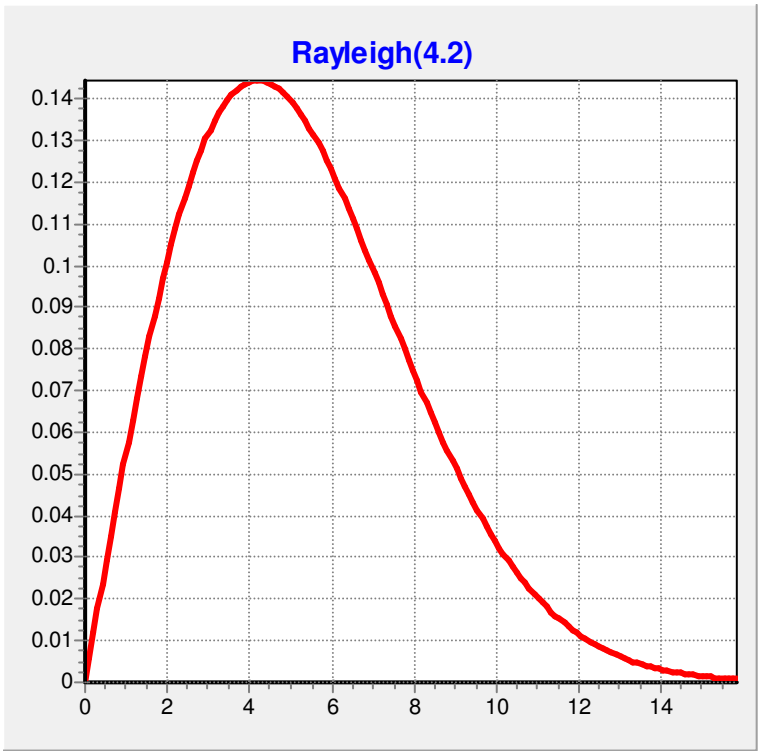
Rayleigh

ErRayleigh(σ): continuous ≥ 0, σ continuous >0.

Returns a random deviate from the Rayleigh distribution.

Mean: $\sigma\sqrt{\frac{\pi}{2}}$.

StDev: $\sigma\sqrt{\frac{4-\pi}{2}}$



Algorithm based on inverse CDF method.
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

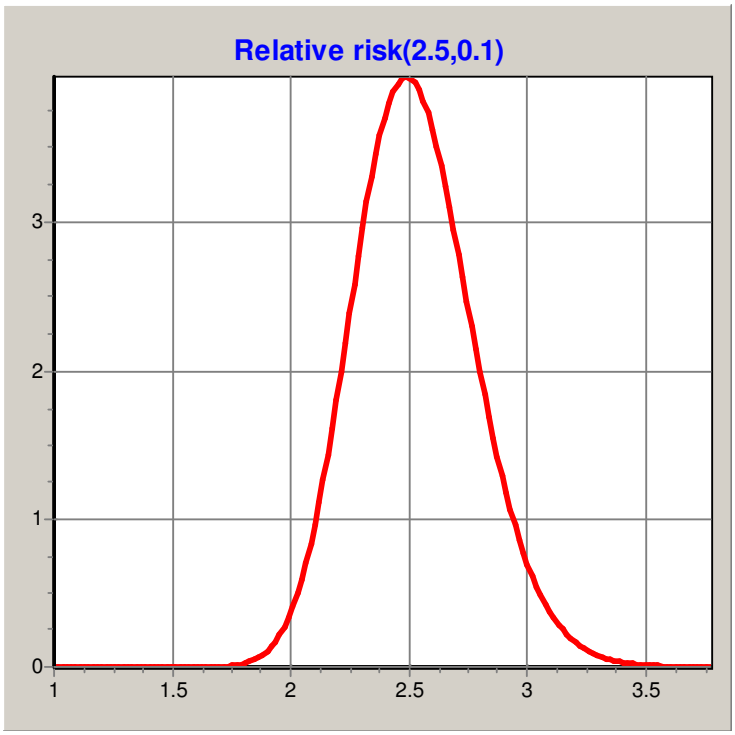
Relative Risk

ErRelativeRisk(RR,SE[ln(RR]): continuous>0.0, RR, SE[ln(RR)] continuous>0.0.

Returns a corrected random deviate assuming that ln(RR) is normally distributed with a standard deviation of SE[ln(RR)]. See notes below.

Mean: RR

StDev: $\sqrt{\exp\left(2\ln(RR)+2\mathrm{SE}[\ln(RR)]^2\right)-\exp\left(2\ln(RR)+\mathrm{SE}[\ln(RR)]^2\right)}$



Notes:

1. See the *Ersatz User Guide*, section *Choosing appropriate distributions* on how to obtain an estimate of the SE[ln(RR)].
2. The assumption that the ln(RR) has a normal distribution causes RR to have a skewed lognormal distribution. As a consequence the mean of random draws from this distribution will be larger than RR (the median will equal RR). The size of the difference depends on the size of SE[ln(RR)]. ErRelativeRisk corrects for this, and ensures that the mean of the random draws equals RR. The uncertainty interval of the random draws will be shifted a bit due to this correction.

The assumption that the ln(RR) is normally distributed and the calculation of SE[ln(RR)] are based on (Rothman 1986), the correction to make the mean of the distribution equal RR is based on (Barendregt 2010), and the algorithm for normally distributed deviates is based on (Press, Teukolsky et al. 1992).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

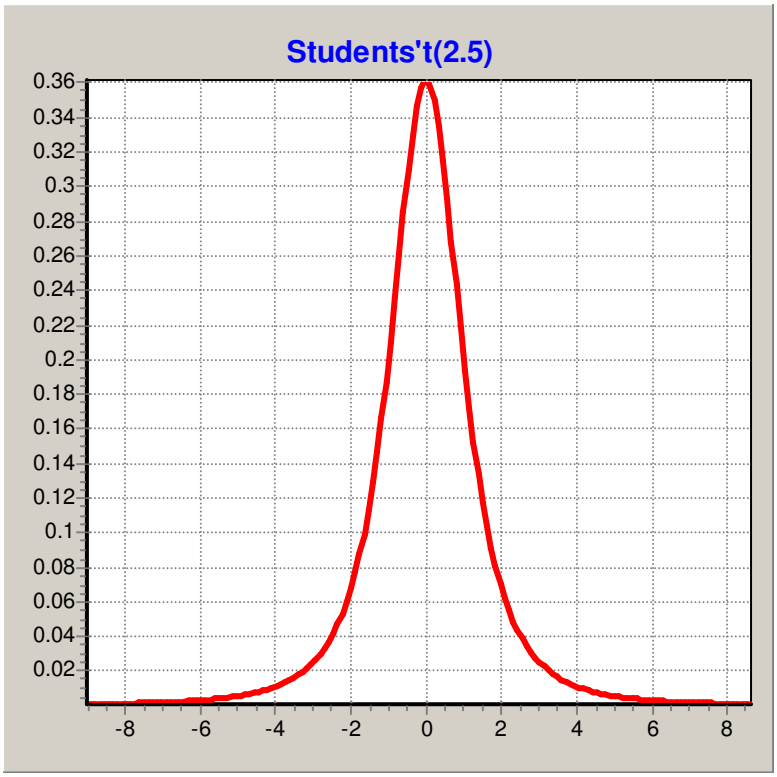
Student's t

ErStudentst(a): continuous. a: continuous >0.

Returns a random deviate from the t-distribution.

Mean: 0

StDev: $= \sqrt{\frac{a}{a-2}} \quad a > 2$
not defined otherwise



Notes:

1. For integer values the *a* parameter is often called 'degrees of freedom'.
2. For a<2 the analytical solution for the standard deviation is not defined, however, Ersatz will report a standard deviation based on numerical methods.

Algorithm based on (Devroye 1986).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Survival

The survival function comes in two variants:

ErSurvival("name",x,p,ct): continuous. name: string; **x**, **p**: Excel ranges of equal length, **x**: discrete, 0,1,2....; **p**: continuous: $\sum_i p_i > 0$, $0 \le p_i \le 1 \ \forall$ i; ct: continous ≥ 0.

ErSurvival2("name",x,p,ct,rnd): continuous. Same parameters as ErSurvival with an additional rnd:continuous, 0<rnd<1.

name is a string with 0<length≤25;
x is an array of times since start of follow-up;
p is an array of conditional failure probabilities (with p_i the probability to fail in the time interval x_i-x_{i+1}, given survival to x_i);
ct is the follow-up time from which the remaining survival is drawn, given survival from time 0 until time ct.
rnd (ErSurvival2 only) is a number between 0 and 1. For ErSurvival2 to return a randomly drawn survival time, rnd needs to be a randomly drawn number from ErUniform01().

Mean and StDev have no closed form equations.

Notes:

1. The Excel ranges can be single or multiple column or row. If multiple, Ersatz reads the range column-wise.
2. Empty cells in the ranges are evaluated as 0.
3. Non-numerical values in the ranges cause this function to return #VALUE!.
4. The function returns #NUM! when the ranges of **x** and **p** are of unequal length.
5. The function returns #NUM! when the **name** string is empty.
6. The function returns (highest value in **x**)+1 when survival is longer than that highest value in **x**.

The ErSurvival2 function is particularly useful for microsimulation applications: it allows to reduce randomness by simulating pairs of persons that are the same in every respect (including their lucky survival draw) except for the survival function. See the example workbooks Survival and SurvivalFix for the contrast between using ErSurvival and ErSurvival2.

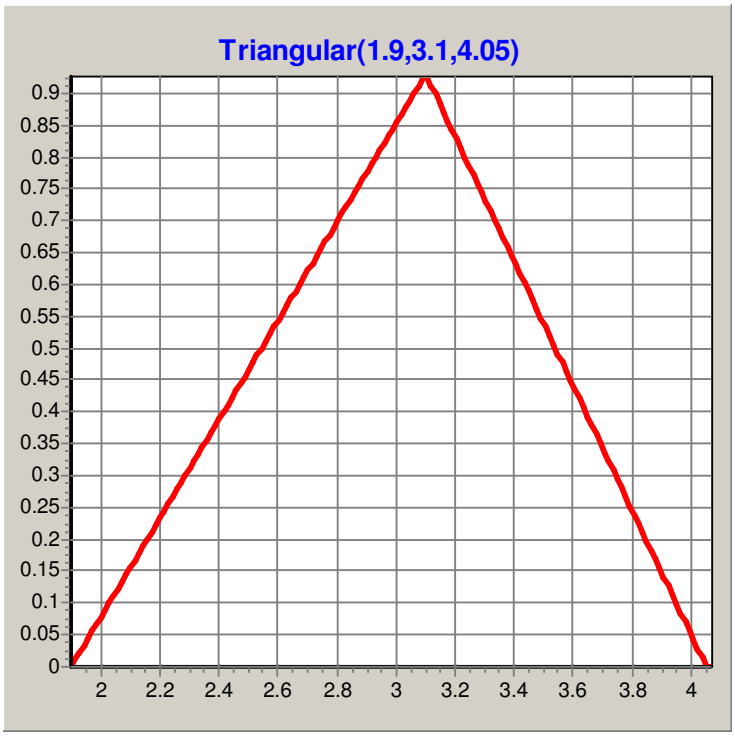
Triangular

ErTriangle(min,most,max): continuous, min, most max: continuous; min<most<max.

Returns a random deviate from the Triangular distribution.

$$\text{Mean}: \frac{\min + \text{most} + \max}{3}$$

$$\text{StDev}: \sqrt{\frac{\min^2 + \text{most}^2 + \max^2 - \min.\text{most} - \max.\text{most} - \min.\max}{18}}$$



Algorithm based on (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Uniform

ErUniform01(): continuous.

Returns a uniformly distributed random deviate between 0 and 1 (exclusive).

Mean: 0.5.

StDev: $\sqrt{\frac{1}{12}}$

Note: make sure that the brackets are included, even if this function takes no parameters.

This function directly returns the result from the random number generator that the user has chosen. See the topic on Random numbers in the *Ersatz User Guide* for details.

ErUniform(min,max): continuous, min, max: continuous, min<max.

Returns a uniformly distributed random deviate between min and max (exclusive).

Mean: $\frac{\text{max - min}}{2}$

StDev: $\sqrt{\frac{(\text{max - min})^2}{12}}$

ErDice(sides): discrete ≥ 1, sides: discrete >1.

Returns a uniformly distributed discrete random deviate between 1 and sides (inclusive).

Mean: $\frac{\text{sides} + 1}{2}$

StDev: $\sqrt{\frac{\text{sides}^2 - 1}{12}}$

Examples: workbook ‘FunctionLineup’, worksheet ‘StandardDistributions’.

Weibull

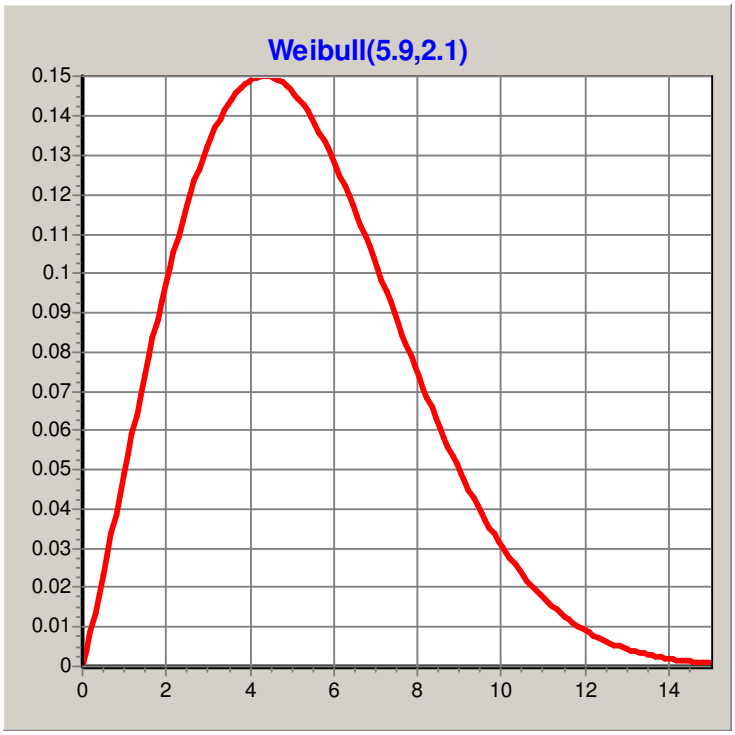
ErWeibull(α,β): continuous>0.0, α continuous>0.0, β continuous>0.0.

Returns a random deviate from the Weibull distribution.

$$\text{Mean}: \left(\frac{\alpha}{\beta}\right)\Gamma\left(\frac{1}{\beta}\right)$$

$$\text{StDev}: \alpha\left[\sqrt{\Gamma\left(\frac{1}{2\beta}\right)} - \Gamma\left(\frac{1}{\beta}\right)\right]$$

Note: in the Ersatz implementation α is the scale and β the shape parameter, in Excel it is the other way around.



Algorithm based on (Law and Kelton 2000).
Example: workbook 'FunctionLineup', worksheet 'StandardDistributions'.

Component functions

Introduction

The Ersatz component functions are a diverse lot, but have one thing in common: there is some sort of dependence between the results from a group of so called 'component' functions. The dependence between function outcomes is achieved by a 'master' function that takes care of the coordination, and a number of 'component' functions that take their cue from the master function. So the functions in this section come in related pairs: a master function, such as ErMultinomial or ErNonparam, each with its own component function, that takes its name from the master with an attached 'Com', in case of ErMultinomial and ErNonparam ErMultinomialCom and ErNonparamCom respectively.
For the user the master function may seem rather superfluous, because it only returns its 'name' parameter. In fact, it is the function that behind the scenes does all the hard work, while the component functions only return values provided by the master function. It is therefore crucial that on each recalculation the master function is executed before the related component functions. The way to achieve that in Excel is to make the component functions depend on the master function, and that is done by linking the 'name' parameter of the component function to the master function's output (which is why it returns its name).
There is no limit to the number of master functions of the same kind in a spreadsheet, however each must have a unique name: it is a fatal error when two master functions have the same name.

Setting up a system of one master function and a bunch of connected component functions can be a bit tricky. The reader is referred to the example workbook 'ComponentFunctions' that comes with the installation and is accessible through the Windows Startmenu in the Ersatz Examples entry.
There are six different component functions. They implement non-parametric bootstrapping, randomisation, the Multinomial and Dirichlet distributions, and correlated Normal and rank correlated arbitrary distributions.

Non-parametric bootstrapping

The master function is:

ErNonparam("name",recnum,fieldnum): name; recnum discrete ≥2, fieldnum discrete>0.

Here 'name' must be a unique string with 0<length≤25, 'recnum' is the number of records, and 'fieldnum' is the number of fields in each record.
The component function is:

ErNonparamCom("name",rec,field,val): continuous; rec: discrete 1≤rec≤recnum, field: discrete 1≤field≤fieldnum, val: continuous.

Here 'name' must be linked to the ErNonparam output, 'rec' is the number of the current record, 'field' is the number of the current field, and 'val' is the value of the current record and field.

Note: Non-parametric bootstrapping relies on random sampling of the original data set with replacement. Internally Ersatz uses the ErDice function to achieve this, with the number of records as parameter. For a discussion of non-parametric bootstrapping, see (Efron and Tibshirani 1993).
Example: workbook 'ComponentFunctions', worksheet 'NonParametric'.

Randomisation

The master function is:

ErRandomise("name",recnum,fieldnum): name; recnum discrete ≥2, fieldnum discrete>0.

Here 'name' must be a unique string with 0<length≤25, 'recnum' is the number of records, and 'fieldnum' is the number of fields in each record.
The component function is:

ErRandomiseCom("name",rec,field,val): continuous; rec: discrete 1≤rec≤recnum, field: discrete 1≤field≤fieldnum, val: continuous.

Here 'name' must be linked to the ErRandomise output, 'rec' is the number of the current record, 'field' is the number of the current field, and 'val' is the value of the current record and field.

Note: A randomisation test relies on random sampling of the original data set without replacement. Ersatz assigns a random number from the ErUniform01 function to each record, and then ranks the records by the size of this number. For a discussion of randomisation, see (Manly 1997).
Example: workbook 'ComponentFunctions', worksheet 'Randomisation'.

Multinomial

The master function is:

ErMultinomial("name",catnum,N): name; catnum discrete ≥2, N discrete>0.

Here 'name' must be a unique string with 0<length≤25, 'catnum' is the number of categories, and 'N' is population number (the sum of numbers of people in each category).
The component function is:

ErMultinomialCom("name",cat,cases): discrete≥0, cat: discrete 1≤cat≤catnum, cases: discrete>0.

Here 'name' must linked to the ErMultinomial output, 'cat' is the number of the category, and 'cases' is the number of people in this category.

Notes:

1. The multinomial distribution is a generalisation of the binomial distribution to more than two categories.

2. Each ErMultinomialCom function returns a random deviate with a binomial distribution, however the ErMultinomial function makes sure that the returned random deviates always sum to N.

Algorithm based on (Devroye 1986).
Example: workbook 'ComponentFunctions', worksheet 'Multinomial'.

Dirichlet

The Dirichlet distribution is a generalization of the Beta distribution to multiple categories. The master function is:

ErDirichlet("name",catnum,N): name; catnum discrete ≥2, N double>0.

Here 'name' must be a unique string with 0<length≤25, 'catnum' is the number of categories, and 'N' is population number (the sum of numbers of people in each category).
The component function is:

ErDirichletCom("name",cat,cases): continuous≥0.0, ≤1.0, cat: discrete 1≤cat≤catnum, cases: double>0.

Here 'name' must be linked to the ErDirichlet output, 'cat' is the number of the category, and 'cases' is the number of people in this category.

Notes:
1. The Dirichlet distribution is a conjugate of the Multinomial distribution.
2. Each ErDirichletCom function returns a random deviate between 0 and 1 with a Beta distribution, however the ErDirichlet function makes sure that the returned random deviates always sum to 1.

Algorithm based on (Devroye 1986).
Example: workbook 'ComponentFunctions', worksheet 'Dirichlet'.

Correlated Normal distributions

The master function is:

ErCorrNormal("name",C,n,corm): name; **C** excel range, n discrete>0, corm:boolean.

Here 'name' must be a unique string with 0<length≤25, 'C' is a correlation or covariance matrix with only the lower triangle significant, n is the number of Normal distributions, and 'corm' is a boolean with ErCorrNormal expecting a correlation matrix when TRUE, and a covariance matrix when FALSE.
The component function is:

ErCorrNormalCom("name",num,mu,sig): continuous, num: discrete 1≤num≤n, mu, sig: continuous, sig>0.

Here ‘name’ must linked to the ErCorrNormal output, ‘num’ is the number of the Normal distribution, and ‘mu’ and ‘sig’ are the parameters of the Normal.

Notes:

1. The ErCorrNormal function accepts either a covariance or a correlation matrix, depending on the corm boolean parameter. Only the information in the lower triangle is used, the rest can be left blank.
2. In both cases the matrix must be valid, i.e. positive semi definite. If it is not, it is a fatal error.
3. Ersatz has an option to check the covariance or correlation matrix for validity, and to suggest a similar matrix that is valid, see the *Ersatz User Guide* topic on correlated random deviates.

Algorithm based on (Law and Kelton 2000), see the *Ersatz User Guide* topic on correlated random deviates for more details.
Example: workbook ‘ComponentFunctions’, worksheet ‘CorrNormal’.

Rank correlated arbitrary distributions

The master function is:

ErRankCorr(“name”,C,n): name; C: Excel range, n discrete>0.

Here ‘name’ must be a unique string with 0<length≤25, ‘C’ is a correlation matrix with only the lower triangle significant, n is the number of distributions.
The component function is:

ErRankCorrCom(“name”,num,func): continuous, num: discrete 1≤num≤n, func: an embedded Ersatz random function.

Here ‘name’ must linked to the ErRankCorr output, and ‘num’ is the number of the distribution.

Notes:

1. The ErRankCorr function accepts a correlation matrix of which only the information in the lower triangle is used, the rest can be left blank.
2. The matrix must be valid, i.e. positive semi definite. If it is not, it is a fatal error.
3. Ersatz has an option to check the correlation matrix for validity, and to suggest a similar matrix that is valid, see the *Ersatz User Guide* topic on correlated random deviates.
4. Ersatz enforces that the random function that returns ‘func’ is entered as an embedded function, e.g. ErRankCorrCom(A3, 1,ErWeibull(5,2.1)), where ‘A3’ is the cell address of the associated ErRankCorr function. It is a fatal error when the random function is not embedded.
5. The function is valid only with a limited number of Ersatz random functions: ErUniform01, ErUniform, ErDice, ErNormal, ErExponential, ErWeibull, ErLognormal, ErGamma, ErBeta, ErRelativeRisk, ErBinomial, ErPoisson,

ErGeometric, ErNegBinomial, ErEmpirical, ErSurvival, and ErStudentst. It is a fatal error to embed a function that is not in this list.

For the algorithm used, see the *Ersatz User Guide* topic on correlated random deviates.
Example: workbook ‘ComponentFunctions’, worksheet ‘RankCorr’.

Correlated multinomial and Dirichlet distributions

Introduction

In the previous section we discussed functions for correlated random numbers from univariate distributions, but multivariate distributions such as the Dirichlet can be correlated as well. However, the functions discussed in the previous section that allow correlated random draws from univariate distributions are not suited for multivariate distributions.
This section describes special functions for random draws from correlated multinomial and Dirichlet distributions. See the *Ersatz User Guide* topic on correlated random deviates from these distributions for the algorithm used.
As with the component functions of the previous section, these correlated distributions functions do not come alone, however for these multivariate distributions not two but three types of functions collaborate: a master function (which does all the heavy lifting), and input and output functions for as many distributions that are correlated.

Correlated Dirichlet distributions

The master function is:

ErDirichletCorr("name",C,n): name; C: Excel range, n discrete>0.

Here 'name' must be a unique string with 0<length≤25, 'C' is a correlation matrix with only the lower triangle significant, n is the number of distributions.
The input function is:

ErDirichletCorrIn("name",inp,num): continuous, inp: Excel range with the parameters of the Dirichlet; num: discrete 1≤num≤n.

Here 'name' must linked to the ErDirichletCorr output, and 'num' is the number of the distribution.

ErDirichletCorrOut("name",num): Excel range of reals; num: discrete 1≤num≤n.

Here 'name' must linked to the corresponding ErDirichletCorrIn function output, and 'num' is the number of the distribution. This function needs to be entered as an Excel array function, see the Introduction on Excel array functions.

Notes:

1. This function requires that the correlated Dirichlet distributions all have the same number of categories. It is a fatal error if they don't.
2. The ErDirichletCorr function accepts a correlation matrix of which only the information in the lower triangle is used, the rest can be left blank.
3. The matrix must be valid, i.e. positive semi definite. If it is not, it is a fatal error.
4. Ersatz has an option to check the correlation matrix for validity, and to suggest a similar matrix that is valid, see the *Ersatz User Guide* topic on correlated random deviates.

Example: workbooks ‘CorrMultivariate.xls’ and ‘ResinBias.xls’.

Correlated multinomial distributions

The master function is:

ErMultinomialCorr(“name”,C,n): name; C: Excel range, n discrete>0.

Here ‘name’ must be a unique string with 0<length≤25, ‘C’ is a correlation matrix with only the lower triangle significant, n is the number of distributions.
The input function is:

ErMultinomialCorrIn(“name”,inp,num): continuous, inp: Excel range with the parameters of the Dirichlet; num: discrete 1≤num≤n.

Here ‘name’ must linked to the ErMultinomialCorr output, and ‘num’ is the number of the distribution.

ErMultinomialCorrOut(“name”,num): Excel range of reals; num: discrete 1≤num≤n.

Here ‘name’ must linked to the corresponding ErMultinomialCorrIn function output, and ‘num’ is the number of the distribution. This function needs to be entered as an Excel array function, see the Introduction on Excel array functions.

Notes:

1. This function requires that the correlated multinomial distributions all have the same number of categories. It is a fatal error if they don’t.
2. The ErMultinomialCorr function accepts a correlation matrix of which only the information in the lower triangle is used, the rest can be left blank.
3. The matrix must be valid, i.e. positive semi definite. If it is not, it is a fatal error.
4. Ersatz has an option to check the correlation matrix for validity, and to suggest a similar matrix that is valid, see the *Ersatz User Guide* topic on correlated random deviates.

Example: workbook ‘CorrMultivariate.xls’.

Output and sensitivity input functions

Introduction

Unlike the functions discussed so far, the functions discussed in this section are not returning randomly drawn values, but are recording and reporting the behaviour of the model. They come in three main flavours:

1. Output functions. These functions record each value a particular variable takes on, and report summary statistics, graphs, and complete output through the Ersatz application. This is the most common way to obtain the results from your model.
2. Sensitivity input functions. These functions allow associating any variable in your model with a unique name, and the ability to be used in multi- and uni-variate sensitivity analysis.
3. Worksheet output functions[1]. Like the plain output functions, these functions record the values a particular variable takes on, but instead of reporting to the Ersatz application, they put the results into the Excel worksheet. This is useful when you need the output values for further processing in the spreadsheet.

In addition to these three main flavours, these functions come in single and multiple run versions. See the *Ersatz User Guide* for details on the 'Multiple runs' and 'Multiple run output mode' options.

Output

ErOutput("name",value): continuous; name: string≤225 characters, value: continuous.

Returns value.

Notes:

1. The 'name' parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErOutput functions have the same name, or when the 'name' string is empty.
2. The 'value' parameter should be an outcome of interest from your model. This outcome will be stored and at the end of the run summary statistics will be calculated.

Examples: workbook 'FunctionLineup', all worksheets.

Multiple run output

ErRunOutput("name",value): continuous; name: string≤225 characters, value: continuous.

Returns value.

Notes:

[1] Please note that the worksheet output functions are not available in the Ersatz trial version.

1. This function is useful when the 'Multiple runs' and 'Multiple run output mode' options are chosen. See the *Ersatz User Guide* for details on these options.
2. The 'name' parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErRunOutput functions have the same name, or when the 'name' string is empty.
3. The 'value' parameter should be an outcome of interest from your model. The function stores a single number at the end of each run, and at the end of multiple runs summary statistics across those runs will be calculated.
4. You can use the Ersatz statistical functions to determine what value is stored at the end of each run, e.g. the sum or mean of an outcome over all iterations. See the section on Statistical functions below.

Example: workbook 'BreastCaMarkovMC', worksheet 'Data & Results".

Sensitivity input

ErSensInput("name",value): continuous; name: string≤225 characters, value: continuous.

Returns value.

Notes:

1. Ersatz does sensitivity analysis on all input vatiables by default, however you can choose to limit the analysis to inputs that are assigned to this function through the 'value' parameter. This has the added benefit that in the sensitivity analysis outcomes the name of the input will be shown instead of its cell address (the default).
2. The 'name' parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErSensInput functions have the same name, or when the 'name' string is empty.

Example: workbook 'BreastCaMarkovSC', worksheet 'Data & Results".

Multiple run sensitivity input

ErRunSensInput("name",value): continuous; name: string≤225 characters, value: continuous.

Returns value.

Notes:

1. This function is useful when the 'Multiple runs' and 'Multiple run output mode' options are chosen, and when some input functions are randomly drawn only once in each run (this will generally mean you are doing either multiple age group macrosimulation, as in BreastCaMarkovMC.xls, or microsimulation, as in BreastCaMicroUnc.xls). See the *Ersatz User Guide* for details on these options.
2. Ersatz does sensitivity analysis on all input vatiables by default, however you

can choose to limit the analysis to inputs that are assigned to this function through the 'value' parameter. This has the added benefit that in the sensitivity analysis outcomes the name of the input will be shown instead of its cell address (the default).

3. The 'name' parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErRunSensInput functions have the same name, or when the 'name' string is empty.

Example: workbook 'BreastCaMarkovMC', worksheet 'Data & Results".

Single output value to worksheet

ErData(inp,itnum): continuous; inp: continuous; itnum: integer, 1≤itnum≤number of iterations in the run.

Returns at the end of a run the value of inp at iteration itnum.

Notes:

1. The 'inp' parameter should be an outcome of interest from your model. The value of that outcome at iteration 'itnum' will be returned at the end of the run, during the run the function returns '0'.
2. This function is useful when you need the value of a specific iteration for further processing in the spreadsheet. See the ErDataArray function below if you need output from all iterations in the worksheet. You could use this function for that purpose, but it will slow down things considerably.

Example: workbook 'Output2Workbook', worksheet 'DataFunctions'. Please note that this function and workbook are not available in the Ersatz trial version.

Multiple output values to worksheet

ErDataArray("name",innum): Excel range; name: string≤225 characters, innum: continuous.

Returns an array of values in the Excel range.

Notes:

1. The 'name' parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErDataArray functions have the same name, or when the 'name' string is empty.
2. The 'innum' parameter should be an outcome of interest from your model. The function stores the value of innum at each iteration, and at the end of the run it returns the values in the Excel range.
3. You need to enter this function as an array formula. See the Note on array formulas in the Introduction. The function assumes that the Excel range for output is a single column. So if you want to get the values of 1000 iterations in the worksheet, you need to enter this function in a single column range of 1000 rows.
4. If the length of the Excel range is less than the number of iterations, the

function returns as many as will fit, starting from iteration 1. If the length of the Excel range is longer than the number of iterations, the excess cells will return #N/A.
5. The function returns #NUM! before an Ersatz run.

Example: workbook 'Output2Workbook', worksheet 'DataFunctions'. Please note that this function and workbook are not available in the Ersatz trial version.

Multiple run multiple output values to worksheet

ErRunDataArray("name",value): Excel range; name: string≤225 characters, value: continuous.

Returns an array of values in the Excel range.

Notes:
1. The 'name' parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErRunDataArray functions have the same name, or when the 'name' string is empty.
2. The 'innum' parameter should be an outcome of interest from your model. The function stores a single number at the end of each run, and at the end of multiple runs it returns the values in the Excel range.
3. You need to enter this function as an array formula. See the Note on array formulas in the Introduction. The function assumes that the Excel range for output is a single column. So if you want to get the values of 500 runs in the worksheet, you need to enter this function in a single column range of 500 rows.
4. If the length of the Excel range is less than the number of runs, the function returns as many as will fit, starting from run 1. If the length of the Excel range is longer than the number of runs, the excess cells will return #N/A.
5. The function returns #NUM! before an Ersatz run.
6. This function is useful when the 'Multiple runs' and 'Multiple run output mode' options are chosen. See the *Ersatz User Guide* for details on these options.
7. You can use the Ersatz statistical functions to determine what value is stored at the end of each run, e.g. the sum or mean of an outcome over all iterations. See the section on Statistical functions below.

Example: workbook 'BreastCaMarkovMC', worksheet 'Data & Results". Please note that this function is not available in the Ersatz trial version and the workbook will show the #NAME? error.

Special functions

Introduction

The functions in this section are 'special' in the sense that they are non-random. Otherwise they are a mixed bag. They provide the user with, among other things, functions to do optimization, to store results of an iteration or run and retrieve them at the next iteration or run, and some basic tools to control the uncertainty analysis or microsimulation. As an example of the latter, ErFixed allows to emulate common random numbers when combined with the ErIteration function (see the section on Random numbers in the *Ersatz User Guide*), or, in combination with ErConditional, to cycle through a number of age groups, all sharing the same randomly drawn effect size.
In many cases these things can be achieved as well by using Excel macros. While Ersatz offers the possibility to run Excel macros on each iteration or run, these functions allow the user to avoid that, with, as a rule, a considerable speed advantage.

Optimization

ErMinimize("name",ir): Excel range; name: string≤225 characters, **ir**: Excel range.

ErMinimizeResult("name",lf): continuous ≥0; name: string≤225 characters, **lf**: continuous ≥0.

These two functions provide access to the four multivariate optimization algorithms implemented in Ersatz, see the section on Optimization in the *Ersatz User Guide*. The functions always work in tandem, and both need to refer to the same name in order to do so.
In addition to the name parameter, ErMinimize takes an Excel range as an input parameter. These values act as the starting point of the minimization. The output of ErMinimize is a range of the same dimensions as its input range.
The ErMinimizeResult function takes, in addition to the name parameter, the outcome of a loss function **lf** as a parameter. The loss function is often modelled as the sum of squared differences between the target and current outcome.

Notes:

1. The ErMinimize function needs to be entered as an array formula in order to return an Excel range. See the note on array formulas in the Introduction. The output range needs to have the same size as the ErMinimize input range.
2. You can have multiple pairs of ErMinimize and ErMinimizeResult functions in a workbook, but each pair needs to have a unique name.
3. Only one pair of optimization functions can be active at one time, and its name needs to be entered in the designated box on the Optimization tab of Ersatz.

Examples: workbook 'OpimizationIt' and workbook 'OptimizationRun'.

Discounted One-off Costs

ErOneOffCostDisc(by,cy,co,r): continuous; **by, cy**: continuous with by≤cy; **co**: continuous ≥0; **r**: continuous ≥0.

Returns the costs **co** incurred at time **cy**, discounted from base year **by** with discount rate **r**.

Notes:
1. The discount rate r is entered as a fraction, not as a percentage.
2. The units of by and cy are years.
3. If r<0 or by>cy the function returns #NUM!.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Discounted Annual Costs

ErAnnualCostDisc(by,sy,ly,co,r): continuous; **by, sy, ly**: continuous with by≤sy≤ly; **co**: continuous ≥0; **r**: continuous ≥0.

Returns the sum of annual costs **co** incurred between **sy** and **ly**, discounted from base year **by** with discount rate **r**.

Notes:
1. The discount rate r is entered as a fraction, not as a percentage.
2. The units of by, sy, and ly are years.
3. If r<0 or by>sy the function returns #NUM!.
4. If ly<sy the function returns 0.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Discounted Life Years

ErLYDisc(by,sy,ly,r): continuous; **by, sy, ly**: continuous with by≤sy≤ly; **r**: continuous ≥0.

Returns the number of life years between **sy** and **ly**, discounted from base year **by** with discount rate **r**.

Notes:
4. The discount rate r is entered as a fraction, not as a percentage.
5. The units of by, sy, and ly are years.
6. If r<0 or by>sy the function returns #NUM!.
7. If ly<sy the function returns 0.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Discounted Health Adjusted Life Years

ErHALYDisc(by,sy,ly,r,ag,ha): continuous; **by, sy, ly**: continuous with by≤sy≤ly; **r**: continuous ≥0; **ag**: Excel range of discrete single years of age; **ha**: Excel range of age-specific health weights, both ranges of equal length.

Returns the health adjusted number of life years between **sy** and **ly**, discounted from base year **by** with discount rate **r**. The health adjustment is done using the age-specific weights given in the **ha** range, with the corresponding ages given in the **ag** range.

Notes:

5. The discount rate r is entered as a fraction, not as a percentage.
6. The units of by, sy, and ly are years.
7. If r<0 or by>sy the function returns #NUM!.
8. If the ag and ha Excel ranges have different lengths the function returns #NUM!.
9. If the ag Excel range have different lengths the function returns #NUM!.
10. If ly<sy the function returns 0.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Fixed

ErFixed(x,num): continuous, **x**: Excel range, num:discrete>0.

Returns the value x_{num} of the range **x**.

Notes:

1. The Excel ranges can be single or multiple column or row. If multiple, Ersatz reads the range column-wise.
2. If num is greater than the length of **x** ErFixed wraps around, i.e. it starts at the first element of **x** again. This happens again at each multiple of the length of **x**.
3. When not running, ErFixed returns the first number of the Excel range.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Conditional

ErConditional(con,value): continuous; con:boolean, value: continuous.

Returns value if con is TRUE, if con is FALSE the function result is the same as on the previous iteration.

Notes:

1. ErConditional returns 0.0 as long as in a run con has not at least once been TRUE.
2. See the topic on 'Conditional firing' in the *Ersatz User Guide* for applications of this function.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Truncate

ErTruncate(min,max,func): continuous; min, max: continuous, min<max; func: an embedded Ersatz random function.

ErTruncate will resample the random function that produces 'func' until min≤func≤max. If the number of tries exceeds the maximum, this function returns #NUM!. The maximum number of tries can be set through Options|Calculation.

Notes:

1. Ersatz enforces that the random function that returns 'func' is entered as an embedded function, e.g. ErTruncate(0, 25,ErNormal(15,5)). It is a fatal error when this is not the case.
2. When Ersatz is not running, the ErTruncate function returns the mean of the embedded function (with the 'Show mean values while not running option' on) or a randomly drawn value from this function (with that option off) when this value is between min and max, when it is lower it will return min, when higher max.
3. The function is valid only with a limited number of Ersatz random functions: ErNormal, ErExponential, ErWeibull, ErLognormal, ErGamma, ErBeta, ErRelativeRisk, ErBinomial, ErPoisson, ErGeometric, ErNegBinomial, ErEmpirical, ErSurvival, and ErStudentst. It is a fatal error to embed a function that is not in this list.
4. This function is provided only reluctantly, because it invites bad modelling practices. When you need this function to avoid certain values from a random distribution function, it usually means you made a poor choice of distribution. See the section on 'Good modelling practice' in the *Ersatz User Guide* for a discussion of choosing appropriate distributions.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Iteration

ErIteration(): discrete>0.

Returns the number of the current iteration.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Set iteration number

ErSetItno(num): discrete>0; num: discrete>0.

This function will set the number of iterations to 'num'. It overrides the number of iterations set in Ersatz's Calculation panel (see the *Ersatz User Guide*).

Examples: workbooks 'BreastCaMicro', 'BreastCaMarkovMC'.

Run number

ErRunno(): discrete>0.

Returns the number of the current run.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Running

ErRunning(): boolean.

ErRunning returns TRUE when Ersatz is running the spreadsheet, FALSE otherwise.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'. You will have to uncheck the 'No screen updates while running' checkbox to see that this function is actually working.

Version

ErVersion(): string.

Returns the version number of the Excel add-in.

Note: This function is useful for documentation of results, and for purposes of support.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Store a single number

ErStore("name",inp): string; name: string≤225 characters, inp: continuous.

Returns the name parameter.

Notes:

1. The 'name' parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErStore functions have the same name, or when the 'name' string is empty.
2. The function stores the inp value, which can be retrieved on the next iteration by the ErRetrieve function.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Retrieve a single number

ErRetrieve("name"): continuous; name: string≤225 characters.

Returns the number stored by ErStore in the previous iteration.

Notes:
1. The ‘name’ parameter should be linked to the corresponding ErStore function.
2. The function returns 0 on the first iteration.

Example: workbook ‘FunctionLineup’, worksheet ‘SpecialFunctions’.

Store an array of numbers

ErStoreArray(“name”,inarray): string; name: string≤225 characters, inarray: Excel range.

Returns the name parameter.

Notes:
1. The ‘name’ parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErStoreArray functions have the same name, or when the ‘name’ string is empty.
2. The function stores the values in the inarray Excel range, which can be retrieved on the next iteration by the ErRetrieveArray function.

Example: workbook ‘FunctionLineup’, worksheet ‘SpecialFunctions’.

Retrieve an array of numbers

ErRetrieveArray(“name”): Excel range; name: string≤225 characters.

Returns the array stored by ErStoreArray in the previous iteration.

Notes:
1. The ‘name’ parameter should be linked to the corresponding ErStoreArray function.
2. You need to enter this function as an array formula. See the Note on array formulas in the Introduction.
3. The function returns an array of 0s on the first iteration.

Example: workbook ‘FunctionLineup’, worksheet ‘SpecialFunctions’.

Conditional store an array of numbers

ErCondStoreArray(“name”,condition,inarray,mode): string; name: string≤225 characters, condition: boolean, inarray: Excel range, mode: integer {0,1,2} .

Returns the name parameter.

Notes:
1. The ‘name’ parameter must be a unique string with 0<length≤225. It is a fatal error when two or more ErCondStoreArray functions have the same name, or

when the 'name' string is empty.
2. The function stores the values in the 'inarray' Excel range when the 'condition' parameter is TRUE.
3. The 'mode' parameter determines how the 'inarray' range is stored. It can take on the following values:
   0: replace the previously stored array;
   1: multiply with the previously stored array;
   2: add to the previously stored array.
   On the first iteration for each value of the 'mode' parameter the 'inarray' range is stored as such.
4. The stored values can be retrieved using the ErCondRetrieveArray function.

Example: workbook 'ConditionalStore'.

Conditional retrieve an array of numbers

ErCondRetrieveArray("name",condition): Excel range; name: string≤225 characters, condition: boolean.

Returns the array stored by ErCondStoreArray.

Notes:
1. The 'name' parameter should be linked to the corresponding ErCondStoreArray function.
2. You need to enter this function as an array formula. See the Note on array formulas in the Introduction.
3. This function returns the array stored by ErCondStoreArray when the 'condition' parameter is TRUE.
4. The function returns an array of #NUM! errors before the workbook is run. You may want to protect against the 'ErOutput function not accounted for' error by using the Excel IsError function (see example workbook).

Example: workbook 'ConditionalStore'.

Sort an array of numbers

ErSortedArray(inarray,mode): Excel range; inarray: Excel range, mode: integer.

Returns the input array sorted in either ascending or descending order, depending on the mode parameter.

Notes:
1. The function returns the input array defined by the 'inarray' parameter sorted in ascending order when the 'mode' parameter is 1, for other values the output is sorted in descending order.
2. You need to enter this function as an array formula. See the Note on array formulas in the Introduction.

Example: workbook 'FunctionLineup', worksheet 'SpecialFunctions'.

Statistical functions

Introduction

The statistical functions in this section are different from the other Ersatz functions in that they return a meaningful value only at the end of a run. Their input is one value per iteration, and they return a value that summarises these inputs.

Mean

ErMean(inp): continuous, inp:continuous.

Returns the mean of the input values.

Example: workbook ‘Output2Workbook’, worksheet ‘StatisticalFunctions’. Please note that this function and workbook are not available in the Ersatz trial version.

Standard deviation

ErStDev(inp): continuous, inp:continuous.

Returns the standard deviation of the input values.

Example: workbook ‘Output2Workbook’, worksheet ‘StatisticalFunctions’. Please note that this function and workbook are not available in the Ersatz trial version.

Minimum

ErMin(inp): continuous, inp:continuous.

Returns the lowest of the input values.

Example: workbook ‘Output2Workbook’, worksheet ‘StatisticalFunctions’. Please note that this function and workbook are not available in the Ersatz trial version.

Maximum

ErMax(inp): continuous, inp:continuous.

Returns the highest of the input values.

Example: workbook ‘Output2Workbook’, worksheet ‘StatisticalFunctions’. Please note that this function and workbook are not available in the Ersatz trial version.

Total

ErTotal(inp): continuous, inp:continuous.

Returns the sum of the input values.

Example: workbook ‘Output2Workbook’, worksheet ‘StatisticalFunctions’. Please note that this function and workbook are not available in the Ersatz trial version.

Correlation

ErCorrelation(X,Y): continuous, X,Y:continuous or discrete.

Returns the Pearson product moment correlation of the input values using:

$$\rho=\frac{\sum_i\left(X_i-\bar{X}\right)\left(Y_i-\bar{Y}\right)}{\sqrt{\sum_i\left(X_i-\bar{X}\right)^2\sum_i\left(Y_i-\bar{Y}\right)^2}}$$

Examples: workbook ‘ComponentFunctions’, worksheets ‘CorNormal’ and ‘RankCorr’, and workbooks ResinBias and CorrMultivariate.

Percentile

ErPercentile(inp,perc): continuous, inp, perc:continuous, 0≤perc≤100.

Returns the perc percentile of the input values.

Example: workbook ‘Output2Workbook’, worksheet ‘StatisticalFunctions’. Please note that this function and workbook are not available in the Ersatz trial version.

References

Barendregt, J. J. (2010). "The effect size in uncertainty analysis." Value in Health **13**(4): 388-391.

Devroye, L. (1986). Non-uniform random variate generation. New York, Springer Verlag.

Efron, B. and R. J. Tibshirani (1993). An introduction to the bootstrap. London, Chapman & Hall Inc.

Gentle, J. E. (2003). Random number generation and Monte Carlo methods. New York, Springer.

Kachitvichyanukul, V. and B. Schmeiser (1985). "Computer generation of hypergeometric random variates." Journal of Statistical Computation and Simulation **22**: 127-145.

Law and Kelton (2000). Simulation analysis.

Manly, B. F. J. (1997). Randomization, Bootstrap and Monte Carlo Methods in Biology. London, Chapman & Hall.

Press, W. H., S. A. Teukolsky , et al. (1992). Numerical Recipes in FORTRAN 77: The Art of Scientific Computing. Cambridge, Cambridge University press.

Press, W. H., S. A. Teukolsky , et al. (2007). Numerical Recipes: The Art of Scientific Computing. Cambridge, Cambridge University press.

Rothman, K. J. (1986). Modern Epidemiology. Boston, Little, Brown.