

Microsoft®

Function Improvements in Microsoft Office Excel 2010

October 2009





This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2009 Microsoft Corporation. All rights reserved.

Microsoft, Microsoft Office Excel are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Contents

1	Introduction	1
1.1	Background	1
1.2	Process for Implementing Changes	6
2	Improved Accuracy	7
2.1	Approximation Methods	7
2.2	User Functions	8
2.3	Helper Functions	17
3	Consistent Function Library	32
3.1	Accurate and Consistent Function Names	32
3.2	Function Details	33
4	References	36

1 Introduction

Microsoft® Office Excel® is the one of the most popular spreadsheet applications that is used by scientists, mathematicians and business professionals around the world. Excel's function library is widely used by these professionals to create solutions to perform statistical analysis, modeling and various other tasks. Based on customer feedback, Microsoft has made improvements in Excel 2010 that features an updated function library providing users with an accurate and consistent set of functions while maintaining compatibility with earlier versions of Excel. This paper will provide a background on the issues addressed and an overview of the changes made to the Excel function library so that educators and members of the academic and Excel communities are able to understand the work done. Changes to algorithms were made in Excel 2010 product in partnership with Frontline Systems, the Numerical Algorithms Group, and ScienceOps to implement the new, more accurate versions of many of these Excel functions. Microsoft also worked in conjunction with the Numerical Algorithms Group to validate the changes.

1.1 Background

Members of the academic community have provided feedback on the Excel function library in various academic literatures going back to 1997. Some significant papers¹ were contributed by Knüsel, McCullough, Wilson, Heiser and Yalta.

Excel 2003 included a number of changes that consisted of improvements made to six categories of Excel's statistical functions that gives at least some assessment of the effect of using earlier versions of Microsoft Excel highlighted by Knüsel, Wilson, and McCullough. These changes are described here:

<http://support.microsoft.com/default.aspx?kbid=828888&product=xl2003>

The changes made in Excel 2003 addressed several areas reported in scientific literature. The function accuracy improvement work started in Excel 2003, and is continuing right through to the release of Excel 2010. Another area of improvement made in Excel 2010 is around the consistency of the names and definitions of Excel functions. The set of concerns reported in certain papers, including the work of D.A. Heiser, and noted by academics and educators in their correspondence with the Excel team, were the consistency of the names and definitions of Excel's functions. Their

¹ Knusel, L. On the Accuracy of Statistical Distributions in Microsoft Excel 97, *Computational Statistics and Data Analysis*, 26, 375-377, 1998.

Knüsel, L. On the Accuracy of Statistical Distributions in Microsoft Excel 2003. *Computational Statistics and Data Analysis*, 48, 445-449, 2005

McCullough, B.D. & B. Wilson, On the accuracy of statistical procedures in Microsoft Excel 97, *Computational Statistics and Data Analysis*, 31, 27-37, 1999.

McCullough, B.D. Microsoft Excel's Not the Wichmann-Hill random number generators. *Computational Statistics & Data Analysis*, 52, Issue 10, 4587-4593, 2008.

McCullough, B.D., Heiser, D.A. On the accuracy of statistical procedures in Microsoft Excel 2007. *Computational Statistics & Data Analysis*, 52, Issue 10, 4570-4578, 2008.

concerns were that some of Excel's functions maybe defined differently than other similarly named functions in the Excel function library, and that other Excel functions' definitions may be inconsistent with industry standard practices.

1.1.1 Improved Accuracy

Algorithms were generally improved either to provide better accuracy or to return results when they previously did not. Acceptable results were obtained in the vast majority of cases in Excel 2007 and earlier. Less accurate results were generally seen when input values were in extreme ranges; however, Excel 2010 has made significant accuracy improvements in these scenarios.

The algorithms for calculating the following statistical distributions have been modified or redesigned for greater accuracy:

Beta distribution	BETADIST, BETAINV
Binomial distribution	BINOMDIST, CRITBINOM
Chi-squared distribution	CHIDIST, CHIINV
Exponential distribution	EXPONDIST
F distribution	FDIST, FINV
Gamma distribution	GAMMADIST, GAMMAINV
Hypergeometric distribution	HYPGEOMDIST
Lognormal distribution	LOGNORMDIST, LOGINV
Negative Binomial distribution	NEGBINOMDIST
Normal distribution	NORMDIST, NORMINV
Standard Normal distribution	NORMSDIST, NORMSINV
Poisson distribution	POISSON
Student's t distribution	TDIST, TINV
Weibull distribution	WEIBULL

The following financial functions have improved accuracy:

Cumulative interest paid on a loan	CUMIPMT
Cumulative principal paid on a loan	CUMPRINC
Interest payment for an investment	IPMT
Internal rate of return for a series of cash flows	IRR

Payment for a loan	PMT
Payment on principal for an investment	PPMT

The accuracy of these additional functions has been improved:

Hyperbolic arcsine	ASINH
Convert function	CONVERT
Error function	ERF
Complementary error function	ERFC
Natural logarithm of the gamma function	GAMMALN
Geometric mean	GEOMEAN
MOD function	MOD
Random number function	RAND
Sample standard deviation	STDEVS
Sample variation	VARS

1.1.2 Consistent Function Library

Excel 2010 offers users a set of functions that are internally consistent and consistent with best practices. This is achieved by adding new functions to the Excel function library to form a complete and consistent set of functions that behave predictably. New functions make use of the new, more accurate algorithms.

The table below shows the new set of statistical distribution functions:

Distribution	PDF/PMF	Left-tailed CDF	Right-tailed/2-tailed CDF	Inverse left-tail CDF	Inverse right-tailed/2-tailed CDF
Beta	BETA.DIST	BETA.DIST		BETA.INV	
Binomial	BINOM.DIST	BINOM.DIST		BINOM.INV	
Chi-squared	CHISQ.DIST	CHISQ.DIST	CHISQ.DIST.RT	CHISQ.INV	CHISQ.INV.RT
Exponential	EXPON.DIST	EXPON.DIST			
F	F.DIST	F.DIST	F.DIST.RT	F.INV	F.INV.RT
Gamma	GAMMA.DIST	GAMMA.DIST		GAMMA.INV	
Hyper-geometric	HYPGEOM.DIST	HYPGEOM.DIST			
Lognormal	LOGNORM.DIST	LOGNORM.DIST		LOGNORM.INV	
Negative	NEGBINOM.DIST	NEGBINOM.DIST			

Binomial					
Normal	NORM.DIST	NORM.DIST		NORM.INV	
Standard Normal	NORM.S.DIST	NORM.S.DIST		NORM.S.INV	
Poisson	POISSON.DIST	POISSON.DIST			
Student's t	T.DIST	T.DIST	T.DIST.RT T.DIST.2T	T.INV	T.INV.2T
Weibull	WEIBULL.DIST	WEIBULL.DIST			

The table below lists new functions in the consistent function library:

Function name	Description
CEILING.PRECISE	Consistent with mathematical definition. Rounds up towards positive infinity regardless of the sign of the number being rounded.
CHISQ.TEST	Name for existing CHITEST function that is internally consistent with naming of other hypothesis test functions.
CONFIDENCE.NORM	Name for existing CONFIDENCE function that is internally consistent with naming of other confidence function.
CONFIDENCE.T	Consistent definition with industry best practices. Confidence function assuming a Student's t distribution.
COVARIANCE.P	Name for existing COVAR function that is internally consistent with naming of other covariance function.
COVARIANCE.S	Internally consistent name with other functions that act on a population or a sample.
F.TEST	Name for existing FTEST function that is internally consistent with naming of other hypothesis functions.
FLOOR.PRECISE	Consistent with mathematical definition. Rounds down towards negative infinity regardless of the sign of the number being rounded.
MODE.MULT	Consistent with user expectations. Returns multiple modes for a range.
MODE.SNGL	Name for existing MODE function that is internally consistent with naming of other mode function.
PERCENTILE.EXC	Consistent with industry best practices, assuming percentile is a value between 0 and 1, exclusive.
PERCENTILE.INC	Name for existing PERCENTILE function that is internally

	consistent with naming of other percentile function.
PERCENTRANK.EXC	Consistent with industry best practices, assuming percentile is a value between 0 and 1, exclusive.
PERCENTRANK.INC	Name for existing PERCENTRANK function that is internally consistent with naming of other PERCENTRANK function.
QUARTILE.EXC	Consistent with industry best practices, assuming percentile is a value between 0 and 1, exclusive.
QUARTILE.INC	Name for existing QUARTILE function that is internally consistent with naming of other quartile function.
RANK.AVG	Consistent with industry best practices, returning the average rank when there is a tie.
RANK.EQ	Name for existing RANK function that is internally consistent with naming of other rank function.
STDEV.P	Name for existing STDEVP function that is internally consistent with naming of other standard deviation function.
STDEV.S	Name for existing STDEV function that is internally consistent with naming of other standard deviation function.
T.TEST	Name for existing TTEST function that is internally consistent with naming of other hypothesis functions.
VAR.P	Name for existing VARP function that is internally consistent with naming of other variance function.
VAR.S	Name for existing VAR function that is internally consistent with naming of other variance function.
Z.TEST	Name for existing ZTEST function that is internally consistent with naming of other hypothesis functions.

1.1.3 Compatibility with Previous Versions of Excel

Workbooks created in previous versions of Excel will calculate in Excel 2010 with improved accuracy while retaining the existing function names. Users of Excel 2010 will be able to use functions with either the old names or the new names but, as with other newly introduced functions, formulas with new Excel 2010 functions will result in #NAME? errors when opened and recalculated in previous versions of Excel.

In order to aid users, especially users not familiar with the definitions of the existing Excel functions, in understanding the meaning of functions, while still enabling the usage of the existing functions, changes have been made to the user interface in Excel 2010 to make the consistent functions easier to find. Existing functions for which a new, consistently-named or consistently-defined version exists will be placed in a new function category called Compatibility, and entries for these functions within the formula auto-complete user interface will appear below the new version with an icon and tooltip that differentiate between the consistent functions and the compatibility functions.

1.1.4 Out-of-Scope Improvements to Accuracy and Consistency

As this document lays out, the changes to Excel 2010 target known inaccuracies and inconsistencies in Excel's function library. Issues outside of the function library, such as issues with the accuracy of the Analysis ToolPak, trend lines in charts, or the internal representation of numbers as double-precision floating point numbers, are beyond the scope of the work that was done in Excel 2010.

1.2 Process for Implementing Changes

The changes made to Excel 2010 were created as a result of lengthy process and partnership with three different companies with expertise in the area of numerical algorithms: Frontline Systems, the Numerical Algorithms Group (NAG), and ScienceOps.

Frontline Systems Inc., headquartered in Incline Village (Lake Tahoe), Nevada, is best known for having developed the Solver in Microsoft Excel, and its Premium Solver and Risk Solver add-ins for advanced optimization and Monte Carlo simulation in Microsoft Excel. Frontline serves about 5,000 customers, mostly large corporations, who use its software to reduce costs and control risks in operations, marketing and finance. Frontline has implemented all of the Excel built-in functions in its PSI Interpreter, used for high-performance optimization and simulation in Microsoft Excel. For more information please go to www.solver.com.

The Numerical Algorithms Group (NAG) is dedicated to applying its exceptional expertise in numerical engineering to delivering high-quality computational software and high performance computing services. For almost 40 years NAG experts have worked closely with world-leading researchers in academia and industry to create powerful, reliable and flexible software which today is relied on by tens of thousands of individual users, as well as numerous independent software vendors. NAG serves

its customers from offices in the UK, US, Japan and Taiwan. For more information please go to www.nag.co.uk.

ScienceOps develops, validates and optimizes scientific software and informatics systems. They have a skilled team of scientists and engineers that have expertise in high-performance scientific computing and in a broad spectrum of scientific and engineering disciplines. For more information please go to www.scienceops.com.

Frontline Systems and NAG submitted new algorithms with improved accuracy for functions identified by the Excel team. These new algorithms and the relevant literature were reviewed by the Excel team with the support and assistance of ScienceOps, including comparisons of accuracy against reference software and performance against previous versions of Excel. Algorithms were chosen on the basis of accuracy given acceptable performance constraints. After the Excel team incorporated the new algorithms into the codebase, NAG was further engaged to perform 3rd-party testing of the resulting implementations of functions in Excel 2010.

2 Improved Accuracy

This section will go into the details of the algorithms we used to improve the accuracy of Excel's statistical functions. In general, a sample declaration for the function is given, followed by a description of the function and how it is approximated.

2.1 Approximation Methods

2.1.1 Polynomial Approximations

A common measure of the quality of an approximation of a real-valued function $f(x)$ by a polynomial $p(x)$ over an interval $[a,b]$ is the uniform norm or minimax norm, defined by

$$\|f(x) - p(x)\|_{\infty} = \max_{x \in [a,b]} |f(x) - p(x)|$$

A best approximation $p(x)$ to $f(x)$ among all polynomials of a given degree in the sense of the minimax norm is called a minimax polynomial approximation to $f(x)$ on $[a,b]$.

Lagrange interpolation is the process of approximating $f(x)$ on $[a,b]$ with a polynomial $L_n(x)$ of degree n by requiring that $f(x)$ and $L_n(x)$ agree at $n+1$ points x_0, \dots, x_n the interval $[a,b]$ called the interpolation points of the approximation.

The choice of equally spaced interpolation points in the interval $[a,b]$ does not always lead to a good approximation in the sense of the uniform norm. On the other hand, a near-minimax approximation can be achieved using Lagrange interpolation in which the interpolation points are linear translations of the zeros of the Chebyshev polynomials $T_n(x)$ of the first kind.

To improve on this near-minimax polynomial approximation, an iterative procedure called the Remez exchange algorithm is utilized. This algorithm involves solving systems of linear equations in very high precision. If there is convergence, then the Remez approximation is the minimax polynomial approximation.

2.1.2 Rational Approximations

A rational function is a quotient of two polynomial functions. Approximations by rational functions are often better than approximations by simple polynomials. The Padé approximant $P_{m,n}(x)$ to $f(x)$ of order (m,n) is the rational function $P_{m,n}(x)=p(x)/q(x)$, where $\deg(p(x))=m$, $\deg(q(x))=n$, $q(x)$ has constant term 1, and $f(x)$ and $P_{m,n}(x)$ agree on the first $m+n+1$ derivatives at $x=0$, that is,

$$f^{(k)}(0) = P_{m,n}^{(k)}(0)$$

for $k=0,\dots,m+n$. $P_{m,n}(x)$ is the (m,n) -Padé approximation or the (m,n) -rational approximation of $f(x)$. Rational approximations have the further advantage that they can deal more naturally with the presence of poles in the function to be approximated.

2.1.3 Continued Fraction Approximations

A continued fraction is an expression of the form

$$a_0 + \frac{b_1}{a_1 + \frac{b_2}{a_2 + \frac{b_3}{a_3 + \dots}}}$$

For convenience, this is written in the condensed form

$$a_0 + \frac{b_1}{a_1 + \frac{b_2}{a_2 + \frac{b_3}{a_3 + \dots}}}$$

For more details on continued fraction approximations, refer to [1].

2.2 User Functions

2.2.1 Beta Distribution

Probability Density Function

The Beta probability density function (PDF)

$$f(x) = \begin{cases} \frac{(x-l)^{\alpha-1}(x-u)^{\beta-1}}{B(\alpha,\beta)(u-l)^{\alpha+\beta-1}}, & \text{for } x \in (l,u) \\ 0, & \text{otherwise} \end{cases}$$

with shape parameters α and β and lower and upper bounds l and u , respectively, is a very flexible probability distribution that can take on a variety of shapes. This distribution is used to model the time to completion in project management and in Bayesian statistics. In Excel's implementation, the lower and upper bound

parameters are optional and if these are not specified, they are taken to be 0 and 1, respectively. Excel uses the derivative of the regularized incomplete beta function in its implementation.

The beta PDF is declared as

```
double BetaPDF (double a, double b, double x, double ub, double lb)
```

where a and b are the shape parameters, ub is the upper bound and lb is the lower bound.

Cumulative Distribution Function

The beta cumulative distribution function (CDF) is

$$F(x) = I_z(\alpha, \beta)$$

Where

$$z = \frac{x - l}{u - l}$$

and I_z is the regularized incomplete beta function

This is declared as

```
double BetaCDF (double a, double b, double x, double ub, double lb)
```

Inverse Cumulative Distribution Function

The inverse of the beta CDF function is declared as

```
double BetaCDFInv (double a, double b, double p, double ub, double lb)
```

Excel uses the [InverseIBeta](#) to compute the z value and then rescales it to the interval [lb,ub].

2.2.2 Binomial Distribution

Probability Density Function

The binomial distribution with parameters n and p is a discrete distribution giving the probability of getting x successes in n independent Bernoulli trials, where p is the probability of success in each trial. The PDF of this distribution is

$$p(x) = \begin{cases} \binom{n}{x} p^x (1-p)^{n-x} & \text{for } x \in \{0, 1, \dots, n\} \\ 0 & \text{otherwise} \end{cases}$$

This function is declared as

```
double BinomialPDF (double n, double p, double x)
```

Excel uses an algorithm that is an enhancement of the procedure in Loader [15]. The PDF of the binomial distribution is computed by first dealing with special cases where explicit answers are known and then using an enhanced version of the algorithm in

[15]. A further enhancement is the use of [TwoSum](#) to get slightly more accurate results. The values of n and x are truncated to integers in Excel's implementation.

Cumulative Distribution Function

The binomial CDF is

$$F(x) = \begin{cases} 0 & x < 0 \\ \sum_{i=0}^{\lfloor x \rfloor} \binom{n}{i} p^i (1-p)^{n-i} & 0 \leq x < n \\ 1 & x \geq n \end{cases}$$

which is declared as

`double BinomialCDF (double n, double p, double x)`

Summing the above terms in the CDF leads to inaccuracy due to underflow/overflow and is time-consuming and so Excel uses the regularized incomplete beta function instead.

Inverse Cumulative Distribution Function

The Inverse CDF function is declared as

`double BinomialCDFInv (double n, double p, double y)`

and is computed using a bisection-based search.

2.2.3 Chi-squared Distribution

Probability Density Function

The chi-squared distribution has density function

$$f(x, k) = \begin{cases} \frac{1}{2^{\frac{k}{2}} \Gamma(\frac{k}{2})} x^{\frac{k}{2}-1} e^{-\frac{x}{2}}, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

where the parameter $k > 0$ is the number of degrees of freedom (truncated to an integer in Excel's implementation) and Γ is the gamma function. This PDF is declared as

`double ChiSquarePDF (double k, double x)`

Excel uses the subroutine [DTerm](#) to approximate this PDF.

Cumulative Distribution Function

The CDF of this distribution is

$$F(x) = P\left(\frac{k}{2}, \frac{x}{2}\right)$$

where P is the [GammaP](#) function. This is declared as

`double ChiSquareCDF (double k, double x)`

Complementary Cumulative Distribution Function

The complementary CDF function is declared as

```
double ChiSquareCDFComplement (double k, double x)
```

Excel uses the [GammaQ](#) function to compute this approximation.

Inverse Cumulative Distribution Function

The inverse CDF is declared as

```
double ChiSquareCDFInv (double k, double y)
```

Excel uses the [inverse of the GammaP](#) function in this implementation.

Inverse Complementary Cumulative Distribution Function

The inverse complementary CDF is declared as

```
double ChiSquareCDFInvComplement (double k, double y)
```

Excel uses [the inverse of the GammaQ](#) function in this implementation.

2.2.4 F-Distribution

Probability Density Function

The F-distribution has PDF

$$f(x) = \frac{(f_1)^{\frac{f_1}{2}} (f_2)^{\frac{f_2}{2}} x^{\frac{f_1}{2}-1}}{\text{Beta}\left(\frac{f_1}{2}, \frac{f_2}{2}\right) (f_1 x + f_2)^{\frac{f_1+f_2}{2}}}$$

for $x > 0$. The parameters $f_1, f_2 > 0$ are the degrees of freedom of the distribution, both truncated to an integer in Excel's implementation. The F-distribution is used to model the ratio of two chi-squared random variables with degrees of freedom f_1 and f_2 , respectively. This is declared as

```
double FDistPDF (double f1, double f2, double x)
```

Using the PDF directly is not a very stable approach to approximation. Excel uses logarithms and the [LogGammaError](#) function to obtain

$$f(x) = \frac{1}{x} \exp \left\{ \Delta \left(\frac{f_1}{2} \right) + \Delta \left(\frac{f_2}{2} \right) - \Delta \left(\frac{f_1 + f_2}{2} \right) + \log(2\sqrt{\pi}) + A_1 + B_1 \right\}$$

where Δ is the [LogGammaError](#) function and the terms A_1 and B_1 are computed as follows:

$$A_1 = \begin{cases} \frac{1}{2} \left\{ \text{LogXPlus1} \left(\frac{f_2}{f_1} \right) - \log(f_2) \right\} & \text{if } f_1 > f_2 \\ \frac{1}{2} \left\{ \text{LogXPlus1} \left(\frac{f_1}{f_2} \right) - \log(f_2) \right\} & \text{if } f_1 \leq f_2 \end{cases}$$

And

$$B_1 = \begin{cases} \frac{(f_1 + f_2)}{2} \left\{ \text{LogXPlus1} \left(\frac{f_2}{f_1 x} \right) - \text{LogXPlus1} \left(\frac{f_2}{f_1} \right) \right\} + \frac{f_2}{2} \log(x) & \text{if } f_1 x > f_2 \\ \frac{(f_1 + f_2)}{2} \left\{ \text{LogXPlus1} \left(\frac{f_1 x}{f_2} \right) - \text{LogXPlus1} \left(\frac{f_1}{f_2} \right) \right\} + \frac{f_1}{2} \log(x) & \text{if } f_1 x \leq f_2 \end{cases}$$

Cumulative Distribution Function

The CDF of the F-distribution is

$$F(x) = I_z \left(\frac{f_1}{2}, \frac{f_2}{2} \right)$$

Where I_z is the incomplete beta function and where

$$z = \frac{f_1 x}{f_1 x + f_2}$$

This function is declared as

```
double FDistCDF (double f1, double f2, double x)
```

Complementary Cumulative Distribution Function

The complementary CDF for the F-distribution is declared as

```
double FDistCDFComplement (double f1, double f2, double x)
```

Excel just calls the incomplete beta function for both the CDF and complementary CDF functions, making sure that the value returned does not have significant cancellation errors.

Inverse Cumulative Distribution Function

The inverse of the CDF for the F-distribution is declared as

```
double FDistCDFInv (double f1, double f2, double p)
```

Inverse Complementary Cumulative Distribution Function

The inverse of the complementary CDF for the F-distribution is declared as

```
double FDistCDFInvComplement (double f1, double f2, double q)
```

Excel uses the inverse of incomplete beta function here and then computes the value of x by calling [TwoSum](#) and [LogXPlus1](#), in order to maximize accuracy.

2.2.5 Hypergeometric Distribution

Probability Density Function

The PDF of the hypergeometric distribution is

$$p(x) = \frac{\binom{D}{x} \binom{M-D}{n-x}}{\binom{M}{n}}$$

Where x is an integer satisfying

$$\max\{0, M - D - n\} \leq x \leq \min\{D, n\}$$

This is declared as

```
double HypergeometricDistPDF (double n, double D, double M, double x)
```

Excel truncates the arguments n, D, M and x to an integer. Excel's implementation uses the fact that the PDF of a hypergeometric distribution can be written in the form

$$p(x) = \frac{\binom{D}{x} y^x (1-y)^{D-x} \binom{M-D}{n-x} y^{n-x} (1-y)^{M-D-n+x}}{\binom{M}{n} y^n (1-y)^{M-n}}$$

and if $0 \leq y \leq 1$, then

$$p(x) = \frac{\text{BinomialPDF}(D, x) \cdot \text{BinomialPDF}(M - D, n - x)}{\text{BinomialPDF}(M, n)}$$

where the probability of success in each case is y. An important issue here is the choice of the parameter y. For example, the choice $y=0.5$ is not very robust. Moreover, for some parameter values, the denominator is zero, which leads to division by zero. So, take

$$y = \frac{\max\{x, n - x\}}{M}$$

unless this leads to a zero denominator, in which case take instead $y=m/M$. Note that when Excel implements this function, the values n, D, M and x are truncated to integers.

Cumulative Distribution Function

The CDF of the hypergeometric distribution is

$$F(x) = \sum_{i=0}^{\lfloor x \rfloor} \frac{\binom{D}{i} \binom{M-D}{n-i}}{\binom{M}{n}}$$

Where $F(x) = 0$ for $x < \max\{0, M-D-n\}$ and $F(x) = 1$ for $x \geq \min\{D, n\}$. This is declared as

```
double HypergeometricDistCDF (double n, double D, double M, double x)
```

When Excel implements this function, the values n, D, M and x are truncated to integers. For the hypergeometric CDF, Excel developed a method based on first principles and experimentation: The series summation is performed using a method derived in the literature for a ratio of hypergeometric distribution PDF values.

Excel's implementation simply sums the individual terms of the following series to compute the CDF,

$$F(x) = \sum_{i=0}^x PDF(i) = PDF(x) \sum_{i=0}^x \frac{PDF(i)}{PDF(x)}$$

Excel sums the ratio of the PDF values, since this is more robust than simply summing the PDF values.

In the above summation, each individual term is of the form PDF(i)/PDF(x). This term is not computed by calling the PDF function, but Excel uses the following relation, described in [46]:

$$\frac{PDF(x+1)}{PDF(x)} = \frac{(D-x)(n-x)}{(x+1)(1-D+M-n+x)}$$

Using this relation is much faster than calling the PDF functions.

Finally, to make the overall implementation more robust, Excel computes the smaller of either the CDF or the complementary CDF; this takes advantage of the fact that the hypergeometric distribution is symmetric. The transition point $n \cdot D/M$ is determined by the mean of the hypergeometric distribution. When the value of x is less than the mean, Excel computes the CDF; otherwise, the complementary CDF is computed.

2.2.6 Lognormal Distribution

Probability Density Function

The lognormal distribution has PDF

$$f(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\ln(x)-\mu}{\sigma}\right)^2}$$

For $x > 0$. The natural log of a lognormal random variable is a normally distributed random variable with mean μ and standard deviation σ . This distribution is declared as

`double LognormDistPDF (double x, double LnMean, double LnStddev)`

To approximate this PDF, Excel uses the definition.

Cumulative Distribution Function

The CDF of the lognormal distribution is

$$F(x) = \Phi\left(\frac{\ln x - \mu}{\sigma}\right)$$

Where $\Phi(x)$ is the standard normal CDF. This is declared as

`double LognormDistCDF (double x, double LnMean, double LnStddev)`

2.2.7 Negative Binomial Distribution

Cumulative Distribution Function

The negative binomial distribution has CDF

$$F(x) = \begin{cases} \sum_{i=0}^{\lfloor x \rfloor} \binom{n+i-1}{n-1} p^n (1-p)^i, & x > 0 \\ 0, & \text{otherwise} \end{cases}$$

and is declared as

```
double NegBinomialCDF (double n, double p, double x)
```

In general, summing the individual terms of the above expression leads to inaccuracies in machine precision implementations and results in a time-consuming algorithm, so Excel uses the incomplete beta function for the algorithm.

2.2.8 Normal Distribution/Standard Normal Distribution

Probability Density Function

The normal distribution has PDF

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where μ is the mean and σ^2 is the variance of the distribution. A normal distribution with mean zero and variance one is also called a standard normal distribution.

The normal PDF is declared as

```
double NormDistPDF (double x, double mu, double sigma)
```

and is calculated directly from the definition.

Cumulative Distribution Function

The normal CDF is

$$\begin{aligned} F(x) &= \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt \\ &= \frac{1}{2} \text{ERFC} \left(-\frac{x-\mu}{\sqrt{2}\sigma} \right) \end{aligned}$$

and is declared as

```
double NormDistCDF (double x, double mu, double sigma)
```

NormDistCDF calls the [ERF](#) function in its implementation.

Inverse Cumulative Distribution Function

The Inverse CDF is declared as

```
double NormDistCDFInv (double y, double mu, double sigma)
```

Excel uses the subroutine [InverseStandardNormal](#) to compute an appropriate value of z , which is then scaled as per the given mean and standard deviation.

2.2.9 Poisson Distribution

Probability Density Function

The Poisson distribution has PDF

$$p(x) = \begin{cases} \frac{e^{-\lambda} \lambda^x}{x!} & \text{for } x = 0, 1, \dots \\ 0 & \text{otherwise} \end{cases}$$

and is declared as

```
double PoissonPDF (double lambda, double x)
```

Evaluating this PDF directly leads to serious cancellation errors when the parameter λ is large and so Excel calls **DTerm** when at least one of λ or x is large and uses the algorithm described in Loader [15] in other cases. When Excel implements this function, the value of x is truncated to an integer.

Cumulative Distribution Function

The CDF of this distribution is

$$f(x) = \begin{cases} e^{-\lambda} \sum_{i=0}^{\lfloor x \rfloor} \frac{\lambda^i}{i!}, & x \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

and is declared as

```
double PoissonCDF (double lambda, double x)
```

Summing the terms leads to inaccurate results and is not very fast. Therefore, Excel uses the **GammaQ** function to approximate this CDF.

2.2.10 Student's t Distribution

Probability Density Function

The Student's t distribution with ν degrees of freedom has PDF

$$f(x) = \frac{\Gamma\left(\frac{1}{2}(\nu + 1)\right)}{\sqrt{\pi\nu}\Gamma\left(\frac{1}{2}\nu\right)} \left(\frac{\nu}{x^2 + \nu}\right)^{\frac{\nu+1}{2}}$$

where ν is a positive integer. The PDF is declared as

```
double StudentsTPDF (double nu, double x)
```

where ν is the degree of freedom and is truncated to an integer in Excel's implementation. The value of ν is truncated to an integer. Excel approximates the Student's t distribution PDF using a modification of the algorithm in [15]. In the special cases when $\nu=1$ or $\nu=2$,

Excel computes the PDF using

$$f(x) = \begin{cases} \frac{1}{\pi(1+x^2)} & \text{if } \nu = 1 \\ \frac{1}{(2+x^2)^{1.5}} & \text{if } \nu = 2 \end{cases}$$

Cumulative Distribution Function

The CDF for the Student's t distribution is

$$F(x) = \frac{1}{2} \left\{ 1 + I_{\frac{x^2}{x^2+\nu}} \left(\frac{1}{2}, \frac{\nu}{2} \right) \right\}$$

For $x \geq 0$, where I_a is the incomplete beta function. The CDF is declared as

`double StudentsTCDF (double nu, double x)`

This function returns the CDF at x of the Student's t distribution with $\text{nu} > 0$ degrees of freedom. Excel truncates nu to an integer in its implementation. Excel calls the [IBeta](#) helper function to calculate the Student's t distribution CDF.

Complementary Cumulative Distribution Function

The complementary CDF is declared as

`double StudentsTCDFComplement (double nu, double x)`

Here, Excel just calls the CDF function, making use of the fact that the PDF is symmetric about the origin. As before, the value of nu is truncated to an integer.

Inverse Cumulative Distribution Function

The inverse CDF is declared as

`double StudentsTCDFInv (double nu, double p)`

and is computed directly using the inverse of the incomplete beta function.

Inverse Complementary Cumulative Distribution Function

The inverse complementary CDF is declared as

`double StudentsTCDFInvComplement (double nu, double p)`

This is computed using the inverse CDF function along with the symmetric nature of the PDF.

2.3 Helper Functions

In this section, the various helper functions that are used to approximate other functions will be discussed.

2.3.1 Beta Function

Declaration

`double Beta (double a, double b)`

Description

This function approximates the beta function

$$B(a, b) = \int_0^1 t^{a-1}(1-t)^{b-1} dt$$

$$= \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

for positive parameters $a, b > 0$, where $\Gamma(x)$ is the gamma function. This function uses the [LogBeta](#) subroutine, which approximates the logarithm of the beta function.

2.3.2 DB0 Function

Declaration

`double DB0 (double x, double a)`

Description

This subroutine approximates the quantity

$$DB0(x) = x \log\left(\frac{x}{a}\right) + a - x$$

with sufficient accuracy when $v = (x-a)/(x+a)$ is close to 0.

Excel uses an implementation similar to that described in [15], but with a larger region of approximation and also makes a call to the [HyperbolicArcTangent](#) subroutine. This is especially important when v is close to zero, since then a more direct evaluation is not sufficiently accurate. Specifically, since

$$\log\left(\frac{x}{a}\right) = \log\left(\frac{1+v}{1-v}\right) = 2\text{arctanh}(v)$$

we have

$$DB0(x) = 2x\text{arctanh}(v) - (x - a)$$

and using the Taylor series expansion for the hyperbolic arctangent gives

$$DB0(x) = 2xv - v(x + a) + 2x \sum_{i=0}^{\infty} \frac{v^{2i+1}}{2i + 1}$$

2.3.3 DTerm Function

Declaration

`double DTerm (double a, double x)`

Description

This function approximates the quantity

$$g(a, x) = \frac{e^{-x}x^a}{\Gamma(x + 1)}$$

for $a, x > 0$, which is closely related to the derivative of the incomplete gamma function. This quantity appears in almost all algorithms for evaluating the incomplete gamma function and is responsible for a major part of the loss of accuracy.

The main concern in approximating $g(a, x)$ is to avoid catastrophic cancellation and overflow errors when both a and x are large. In such cases, Excel uses a modification of the Loader algorithm described in [15]. In all other cases, Excel uses the definition of this function directly and calls the subroutine **TwoSum**, which is used to add the terms obtained by taking the logarithm of both sides of the equation above for $g(a, x)$.

2.3.4 ERF and ERFC Functions

Declarations

```
double ERF (double x)
double ERFC (double x)
```

Description

This function approximates the error function

$$ERF(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

or the related complementary error function

$$ERFC(x) = 1 - ERF(x)$$

using rational approximations that depend on the range of the argument x .

2.3.5 ExpXMinus1

Declaration

```
double ExpXMinus1 (double x)
```

Description

This function, also known as the `expm1` function in some mathematical libraries, approximates the quantity $e^x - 1$ when x is close to zero. In this case, direct approximations to e^x return a quantity very close to 1 and this can result in unacceptable errors in the approximation of $e^x - 1$. The precise range of x -values for which to use the `ExpXMinus1` approximation of $e^x - 1$ depends on the algorithm in which the function is being used.

Excel uses a simple procedure based on the following approximation of Kahan:

$$\text{ExpXMinus1}(x) = (e^x - 1) \left(\frac{x}{\log e^x} \right)$$

along with one call to the **TwoSum** subroutine. When x is large enough so that subtractive cancellation will not occur, Excel uses $e^x - 1$ directly. When x is very small, Excel approximates $e^x - 1$ by x .

2.3.6 GammaLnStirling

Declaration

`double` GammaLnStirling (`double` x)

Description

This function approximates the logarithm $\log(\Gamma(x))$ of the gamma function

$$\Gamma(x) = \int_0^{\infty} t^{x-1} e^{-t} dt$$

For $x > 0$. In general, it is nontrivial to get an accurate approximation for the logarithm of the gamma function for x close to 1 or 2. However, this implementation gives accurate approximations for both of these important argument ranges.

Excel uses rational approximations in certain ranges along with a fast, accurate version of Stirling's formula described in [13], series 6.1.41. This series calls the subroutine [LogGammaError](#). For large values of x , Excel uses Stirling's formula, otherwise Excel uses rational approximations.

2.3.7 GammaP, GammaQ and Related Functions

Declaration

`double` GammaP (`double` a, `double` x)

`double` GammaQ (`double` a, `double` x)

Description

These functions approximate the regularized incomplete gamma function. The function GammaQ(a,x) approximates the regularized upper incomplete gamma function

$$Q(a, x) = \frac{1}{\Gamma(a)} \int_x^{\infty} t^{a-1} e^{-t} dt$$

and the function GammaP(a,x) approximates the regularized lower incomplete gamma function

$$P(a, x) = \frac{1}{\Gamma(a)} \int_0^x t^{a-1} e^{-t} dt = 1 - Q(a, x)$$

In both cases, the parameter x must be nonnegative. There are the following special values:

- 1) GammaQ(a,0)=1
- 2) GammaP(a,0)=0
- 3) GammaQ(0,x)=0
- 4) GammaP(0,x)=1
- 5) GammaQ(0.5,x)=ERFC(\sqrt{x})

- 6) $\text{GammaP}(0.5,x) = \text{ERF}(\sqrt{x})$
- 7) $\text{GammaQ}(1,x) = e^{-x}$
- 8) $\text{GammaP}(1,x) = 1 - e^{-x}$

Based on the values of the parameters a and x , different subroutines (described below) are used to compute the GammaQ and GammaP functions. This ensures that cancellations are avoided and that the selected subroutine converges for particular arguments. The selection of these subroutines is based on the algorithm described in [4], which provides a detailed analysis of the reasons for choosing the various approximations over different regions of the parameter space. It was found that Excel's overall algorithm achieves more accuracy than the procedures given in [12] and [3]. Excel calculates GammaQ from GammaP, or GammaP from GammaQ, depending on the values of a and x .

2.3.8 GammaP_CF

Declaration

`double GammaP_CF (double a, double x)`

Description

This function computes the continued fraction for the GammaP function. Here, Excel use a new continued fraction 06.06.10.0007.01 in the Wolfram functions site [37]:

$$\Gamma(a, x) = \Gamma(a) - \frac{x^a e^{-x}}{a - \frac{ax}{a+x+1 - \frac{(a+1)x}{a+x+2 - \frac{(a+2)x}{a+x+3 - \dots}}}}$$

2.3.9 GammaP_Series

Declaration

`double GammaP_Series (double a, double x)`

Description

This function computes the GammaP function using a modification of 6.5.29 in [13] (which is similar to equation 5.5 in [22])

$$P(a, x) = \frac{x^a e^{-x}}{\Gamma(a+1)} \sum_{i=0}^{\infty} \frac{\Gamma(a+1)}{\Gamma(a+i+1)} x^i$$

This series converges quickly when $a \geq x$.

2.3.10 GammaQ_AsymSeries

Declaration

`double GammaQ_AsymSeries (double a, double x)`

Description

This function approximates the GammaQ function using the asymptotic series 6.5.32 in [13]:

$$Q(a, x) = \frac{e^{-x}x^{a-1}}{\Gamma(a)} \left[1 + \frac{a-1}{x} + \frac{(a-1)(a-2)}{x^2} + \dots \right]$$

2.3.11 GammaQ_CF

Declaration

`double` GammaQ_CF (`double` a, `double` x)

Description

This function approximates the continued fraction for the GammaQ function. Here, Excel uses the continued fraction 6.5.31 in [13]:

$$Q(a, x) = \frac{x^a e^{-x}}{\Gamma(a)} \left[\frac{1}{x} + \frac{1-a}{1+x} + \frac{1}{x} + \frac{2-a}{1+x} + \frac{2}{x} + \dots \right]$$

2.3.12 GammaQ_HalfIntegerA

Declaration

`double` GammaQ_HalfIntegerA (`double` a, `double` x)

Description

This function approximates GammaQ for half-integers $0 < a < 20$, using the series

$$Q(a, x) = \text{ERFC}(\sqrt{x}) + \frac{e^{-x}}{\sqrt{x}} \sum_{m=1}^{|a|} \frac{x^m}{\Gamma(\frac{1}{2} + m)}$$

2.3.13 GammaQ_IntegerA

Declaration

`double` GammaQ_IntegerA (`double` a, `double` x)

Description

This function approximates GammaQ for integers $0 < a < 20$, using the series 06.06.06.0005.01 from the Wolfram Functions site [37]:

$$Q(a, x) = e^{-x} \sum_{n=0}^{a-1} \frac{x^n}{n!}$$

2.3.14 GammaQ_TemmeAsm

Declaration

`double` GammaQ_TemmeAsm (`double` a, `double` x)

Description

This function computes the GammaQ function when a and x are large and close to each other. When both parameters are large, the regularized incomplete gamma functions behave like error functions. These asymptotic formulas are given by

$$P(x, a) = \frac{1}{2}ERF(\sqrt{y}) + \frac{e^{-y}}{\sqrt{2\pi a}}T(a, \lambda) \text{ for } x > a$$

$$Q(x, a) = \frac{1}{2}ERF(\sqrt{y}) - \frac{e^{-y}}{\sqrt{2\pi a}}T(a, \lambda) \text{ for } x \leq a$$

$$\lambda = \frac{x}{a}$$

$$y = -a(1 + \log(\lambda) - \lambda)$$

The expansions of $P(x,a)$ and $Q(x,a)$ are from Temme [22], which is based on asymptotic approximations for the GammaQ and GammaP functions given in [6], [8]. Excel chooses to use the expansion of $T(a,\lambda)$ in [22], since this is much less involved and is comparable in accuracy to [4]. In Excel's implementation, Excel computes $T(a,\lambda)$ as

$$T(a, \lambda) = \frac{1}{\text{LogGammaError}(a)} \sum_{k=0}^n b_k(a)z^k$$

$$\sigma = \frac{x}{a} - 1$$

$$z = \begin{cases} \sqrt{2(\sigma - \log(1 + \sigma))} & \text{if } x \geq a \\ -\sqrt{2(\sigma - \log(1 + \sigma))} & \text{if } x < a \end{cases}$$

where n is chosen to achieve the desired accuracy. The coefficients $b_k(a)$ are computed using the backward recursion:

$$b_{k-1}(a) = f_k + \frac{k+1}{a} b_{k+1}(a)$$

with initial values

$$b_{29}(a) = f_{30} \text{ and } b_{28}(a) = f_{29}$$

where the values f_k are first computed using the recursion

$$f_0 = 1$$

$$f_1 = 1/3$$

$$f_2 = \frac{1}{12}$$

$$f_3 = \frac{-2}{135}$$

$$f_k = \frac{k+1}{k+2} \left[\frac{k-1}{3k} f_{k-1} + \sum_{j=3}^{k-1} \frac{f_{j-1} f_{k-(j-1)}}{k-j+2} \right] \text{ for } k \geq 4$$

2.3.15 HyperbolicArcTangent

Declaration

`double` HyperbolicArcTangent (`double` x)

Description

This function approximates the inverse of the hyperbolic tangent function, defined by

$$\operatorname{arctanh}(x) = \int_0^x \frac{dt}{(1-t)^2} = \frac{1}{2} \log\left(\frac{1+x}{1-x}\right)$$

Excel approximates this function only for $|x| < 1$, but since the function $\operatorname{arctanh}$ is odd, we need only consider positive values of x . The approximation is done by summing a partial sum of the infinite series 4.6.33 in [13].

2.3.16 IBeta

Declaration

`double` IBeta (`double` a, `double` b, `double` x)

Description

This function approximates the regularized incomplete beta function, defined as

$$I_x(a, b) = \frac{1}{B(a, b)} \int_0^x t^{a-1} (1-t)^{b-1} dt$$

for $x \in (0, 1)$, where $B(a, b)$ is the (complete) beta function

$$B(a, b) = \int_0^1 t^{a-1} (1-t)^{b-1} dt = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

The parameters a and b are assumed to be positive. For this purpose, Excel also uses the property

$$I_{1-x}(a, b) = 1 - I_x(a, b)$$

Transitions

Different approaches to the approximation are taken as follows. When $a=1$ or $b=1$, Excel uses the relations

$$I_x(1, b) = 1 - (1-x)^b$$

$$I_x(a, 1) = x^a$$

When $a=b=1/2$, Excel uses the relation

$$I_x\left(\frac{1}{2}, \frac{1}{2}\right) = \frac{2}{\pi} \sin^{-1}(\sqrt{x})$$

When $a=b$ and x is small but positive, Excel uses the relation 26.5.14 in [13]:

$$I_x(a, a) = \frac{1}{2} I_{(2x-1)^2}\left(a, \frac{1}{2}\right)$$

Many different subroutines are used to approximate the function $I_x(a,b)$, each algorithm being more robust in a specific region of the parameters. The selection of the algorithm to use is based on Didonato and Morris [19] with some enhancements. Indeed, [19] provides a detailed analysis of the reasons to choose various approximations over different regions of the parameter space.

Excel has modified slightly the regions where each algorithm is used, while ensuring that the implementation is accurate. Excel also uses an approximation based on the algorithm discussed in [33], where the approximation is shown to be at least as accurate as the approach in [19] while at the same time being simpler to implement. Also, when a and b are large, Excel uses the `IBetaCF` function, which is similar to the asymptotic expansion given in [19].

Before choosing which algorithm to use, a check similar to the one described in [19] is made to see if the arguments need to be interchanged so as to avoid cancellation and improve the rate of convergence. The subroutines `IBetaBUP`, `IBeta_Series` and `BGRATSerAsm` are called with different arguments, depending on the parameters.

2.3.17 BGRATSerAsm

Declaration

`double` BGRATSerAsm (`double` a , `double` b , `double` x , `double` y)

Description

This function approximates the regularized incomplete beta function when a is large relative to b and x is close to 1. This region is considered to be problematic and asymptotic expansions are generally used. The asymptotic expansion in [19] has complicated coefficients that are too time-consuming to compute. Hence, Excel uses the substantially different asymptotic described in [33], which is much simpler and faster. The expansion is

$$I_x(a,b) = Q(b, -\gamma \log(x)) + \frac{x^\gamma}{\gamma B(a,b)} \sum_{n=0}^{\infty} \gamma^{-n} T_n(b,x)$$

where $\gamma = a + (b-1)/2$ and Q is the GammaQ function. Most of the error involved here is in the GammaQ function and in the first few terms of the series part.

Computing the coefficients $T_n(b,x)$ is rather involved and is done in the function `ComputeBGRITTerm` as follows:

$$\begin{aligned} T_n(b,x) &= \frac{d^n}{dt^n} \left[\left[2 \sinh\left(\frac{t}{2}\right) \right]^{b-1} - \sum_{m=0}^{\frac{n}{2}} c_m t^{2m+b-1} \right]_{t=-\log(x)} \\ &= \frac{d^n}{dt^n} \left[\sum_{m=\frac{n}{2}+1}^{\infty} c_m t^{2m+b-1} \right]_{t=-\log(x)} \end{aligned}$$

The coefficients c_m are the expansion coefficients of the power series of

$$\left[\frac{\sinh\left(\frac{t}{2}\right)}{\frac{t}{2}} \right]^{b-1}$$

and these can also be represented as generalized Bernoulli polynomials. One way to evaluate the coefficients c_m is to use the recursive procedure in [19]; however, this is relatively slow. Excel expands the power series and determines the exact form of the coefficients c_m polynomials in the parameter b .

2.3.18 IBeta_Series

Declaration

`double IBeta_Series (double a, double b, double x)`

Description

This function computes the regularized incomplete beta function using the power series 06.21.06.0001.01 described on the Wolfram functions website [37]:

$$I_x(a, b) = \frac{x^a}{B(a, b)} \left[\frac{1}{a} + \frac{(1-b)x}{1+a} + \frac{(1-b)(2-b)x^2}{2(2+a)} + \dots + \frac{(1-b)(2-b) \dots (n-b)x^n}{n!(n+a)} + \dots \right]$$

2.3.19 IBetaBUP

Declaration

`double IBetaBUP (double a, double b, double x, double y, int n)`

Description

The following equation, based on equation 26.5.16 in [13], is used to transform the original parameters to an easier-to-compute region of the parameter space:

$$I_x(a, b) = I_x(a + 1, b) + \frac{\Gamma(a + b)}{\Gamma(b)\Gamma(a + 1)} x^a (1 - x)^b$$

This equation can be generalized to the following, as described in [19]:

$$I_x(a, b) = I_x(a + n, b) + x^a (1 - x)^b \sum_{j=1}^n \frac{\Gamma(a + b + j - 1)}{\Gamma(b)\Gamma(a + j)}$$

for $n \geq 1$, thus allowing us to translate the parameter a by any positive integer n . [IBetaxDerivative](#) is used to calculate the summation portion of the formula above.

2.3.20 IBetaCF

Declaration

`double IBetaCF (double a, double b, double x, double y)`

Description

This function computes the regularized incomplete beta function using the continued fraction described in Didonato and Morris [19]:

$$I_x(a, b) = \frac{x^a y^b}{aB(a, b)} \left(\frac{1}{1 + \frac{d_1}{1 + \frac{d_2}{1 + \dots}}} \right)$$

Where

$$d_{2n} = \frac{n(b - n)}{(a + 2n - 1)(a + 2n)} x, \quad n > 0$$

And

$$d_{2n+1} = \frac{(a + n)(a + b + n)}{(a + 2n)(a + 2n + 1)} x, \quad n \geq 0$$

2.3.21 IBetaxDerivative

Declaration

`double` IBetaxDerivative (`double` a, `double` b, `double` x)

Description

This function, defined by

$$\text{IBetaxDerivative}(a, b, x) = \frac{x^{a-1}(1-x)^{b-1}}{B(a, b)}$$

approximates the derivative of the regularized incomplete beta function

$$I_x(a, b) = \frac{1}{B(a, b)} \int_0^x t^{a-1}(1-t)^{b-1} dt$$

with respect to z. When a,b>1, there is a danger of cancellation error and so Excel uses a modification of the algorithm described in [15]. Otherwise Excel calculates IBetaxDerivative directly from equation above.

2.3.22 LogBeta

Declaration

`double` LogBeta (`double` a, `double` b)

Description

This function approximates the logarithm of the (complete) beta function B(a,b), which is defined only for a,b≥0. When a=1 or b=1, the beta function is equal to 1/b or 1/a, respectively. Otherwise, Excel uses the fact that

$$\text{LogBeta}(a, b) = \text{LogGamma}(a) + \text{LogGamma}(b) - \text{LogGamma}(a + b)$$

where LogGamma is the logarithm of the gamma function.

2.3.23 LogGammaError

Declaration

```
double LogGammaError (double x)
```

Description

The logarithm of the gamma function has the following asymptotic series, which is series 6.1.41 in [13]:

$$\text{LogGamma}(x) = \left(x - \frac{1}{2}\right)\log(x) + \frac{1}{2}\log(2\pi) - x\left(\frac{1}{12x} - \frac{1}{360x^3} + \frac{1}{1260x^5} - \frac{1}{1680x^7} + \dots\right)$$

The LogGammaError function returns the infinite series part of this formula and is known as the LogGammaError function since this computes the error term in the asymptotic Stirling approximation to the LogGamma function.

This function is used in many other functions, primarily to approximate factorials accurately, which grow very quickly. Factorials appear in many statistical distributions involving binomial coefficients or ratios of factorials. Computing these factorials directly usually results in a severe loss of accuracy and so the use of the logarithm of the gamma function is preferred. However, this can also result in cancellation errors and so when computing binomial coefficients or other terms involving factorials, Excel uses this function along with the simple Stirling approximation indicated above (see [13]).

LogGammaError accepts only positive arguments and returns infinity (or the largest representable positive number) when $x < 0$. When x is large, Excel sums the first several terms of the series above, which results in very high accuracy. Otherwise, use of the series above can result in large errors, so Excel proceeds as follows: If x is integral, Excel returns the appropriate pre-computed value and if x is non-integral, Excel uses the [LogGammaErrorDiff](#) function to first bring x to an appropriate value and then use the series above.

2.3.24 LogGammaErrorDiff

Declaration

```
double LogGammaErrorDiff (double x)
```

Description

This function approximates the following quantity for small positive x :

$$\begin{aligned} \text{LogGammaErrorDiff} &= \text{LogGammaError}(x) - \text{LogGammaError}(x + 1) \\ &= x \log\left(1 + \frac{1}{x}\right) - 1 + \frac{1}{2}\log(1 + x) - \frac{1}{2}\log(x) \end{aligned}$$

Since this function is called only by the [LogGammaError](#) function for small x , the use of asserts or other checking conditions were deemed to be unnecessary.

2.3.25 LogXPlus1

Declaration

```
double LogXPlus1 (double x)
```

Description

This function approximates the quantity $\log(1+x)$ when x is small. Otherwise, Excel uses direct calculation.

2.3.26 LogXPlus1MinusX

Declaration

`double` LogXPlus1MinusX (`double` x)

Description

This function approximates the quantity $\log(x+1)-x$. When x is small, $\log(x+1)$ is close to x , which encourages cancellation error. Excel was able to obtain full machine accuracy by using a different algorithm, based on the following:

$$\begin{aligned} \log(x + 1) - x &= 2 \tanh^{-1}\left(\frac{x}{x + 2}\right) - x \\ &= \left(\frac{x}{x + 2}\right) \left[2 \left\{ \frac{1}{3} \left(\frac{x}{x + 2}\right)^2 + \frac{1}{5} \left(\frac{x}{x + 2}\right)^4 + \dots \right\} - x \right] \end{aligned}$$

2.3.27 TwoSum

Declaration

`double` TwoSum (`double` a , `double` b)

Description

TwoSum computes the round off error when adding two floating point numbers. Specifically, TwoSum performs the addition operation $a+b$, in a precision that is double the current working precision and returns the rounding error when the operation $a+b$, is computed in the current working precision. The function uses an algorithm of Donald Knuth in [25].

The approximation in TwoSum is an example of an error-free transformation for addition, which is an algorithm that computes the rounding error due to finite precision arithmetic, while remaining at the same finite precision. TwoSum is used by many other functions to obtain accurate intermediate results, for instance when approximating $\exp(a+b)$.

2.3.28 InverseERF

Declaration

`double` InverseERF (`double` p)

Description

This function approximates the inverse of the error function for $-1 \leq p \leq 1$. Since the inverse error function is odd, we need consider only positive values. When $p \leq 0.85$, Excel uses two new rational approximations: When $p^2 \leq 0.5$, Excel use a rational approximation to the function

$$\frac{1}{\sqrt{y}} \operatorname{ERF}^{-1}(\sqrt{y})$$

and when $p^2 > 0.5$, Excel uses a rational approximation to the function

$$\frac{1}{\sqrt{-\log(1 - \sqrt{y})}} \operatorname{ERF}^{-1}(\sqrt{y})$$

When $0.85 < p \leq 0.9375$, Excel uses a rational approximation of the function

$$\frac{1}{\sqrt{-\log(1 - y)}} \operatorname{ERF}^{-1}(y)$$

Finally, when $x > 0.9375$, Excel calls the [ComputeStdNormalTail](#) subroutine, which is accurate in this range.

2.3.29 InverseERFC

Declaration

```
double InverseERFC (double q)
```

Description

This function approximates the inverse of the complementary error function for $0 \leq q \leq 2$ by simply calling the inverse error function.

2.3.30 InverseGammaP, InverseGammaQ

Declaration

```
double InverseGammaP (double a, double p, double initialX)
double InverseGammaQ (double a, double q, double initialX)
```

Description

These functions compute the inverse of the GammaP function $P(x,a)$ and the GammaQ function $Q(x,a)$, for a fixed value of $a > 0$, specifically InverseGammaP computes the value of x given a and p in the equation $P(a,x)=p$ and InverseGammaQ computes the value of x given a and q in the equation $Q(a,x)=q$. Here $0 \leq p \leq 1$ and $0 \leq q \leq 1$.

Simple closed form solutions are available for the inverses when $a=0$ or $a=1$. When $a=0.5$, the error functions are used to find the inverses. The InverseGammaQ function checks to see if $a=0$, 1 or 0.5 and if not, then it just calls InverseGammaP. The InverseGammaP function uses a simple Newton iteration to find the inverse.

The initial estimate is crucial to the efficiency of Newton's method. The subroutine InverseGammaPInitApprox is called to provide an initial value. Excel uses a modification of the procedure proposed in [20].

Excel uses the first term of the power series as the initial approximation when p is small; otherwise Excel uses a polynomial approximation to the extended Wilson-Hilferty formula given in [27]. When both p and x are large, Excel modifies the

polynomial approximation to the extended Wilson-Hilferty formula as described in [20].

2.3.31 InverseIBeta

Declaration

`double` InverseIBeta (`double` a, `double` b, `double` p, `double` initialX)

Description

This function computes the inverse of the regularized incomplete beta function $I_x(a,b)$, that is, given a,b and p, the function returns the value of x in the equation $I_x(a,b)=p$. Here, a and b are positive and $0 \leq p \leq 1$.

A simple Newton method is used to find the root of this equation. The initial value is very important in Newton's method. It is determined by calling the subroutine InverseIBetaInitApprox, which uses an enhancement of the algorithms described in [39] and the approximation 26.5.22 in [13]. Excel found these approximations to be better than the algorithms described in [10] and [16].

2.3.32 InverseStandardNormal

Declaration

`double` InverseStandardNormal (`double` x)

Description

This function approximates the inverse of the standard normal distribution function $x=N_{0,1}(y)$, for $0 < x < 1$. When x is in the tail region, Excel uses the subroutine [ComputeStdNormalTail](#), which employs the algorithm in [18]. In all other cases, Excel calls the inverse error function using the relationship

$$\text{InverseStandardNormal}(x) = \sqrt{2} \text{ERF}^{-1}(2x - 1)$$

2.3.33 ComputeStdNormalTail

Declaration

`double` ComputeStdNormalTail (`double` x, `double` pcomplement)

Description

This computes the inverse of the standard normal distribution $x=N_{0,1}(y)$ in the tails. Excel uses Wichura's algorithm [47].

3 Consistent Function Library

3.1 Accurate and Consistent Function Names

To improve clarity around what Excel functions do and to have a consistent naming scheme across the function library, Excel 2010 offers new versions of existing functions with more accurate and descriptive names.

3.1.1 New Naming Scheme

Excel 2010 uses a new naming scheme for the Excel function library:

Naming Scheme	Description
<function>.<descriptor>	The descriptor differentiates one version of a function from another.
<distrib>.DIST	Functions with this syntax will be the <i>left-tailed</i> cumulative distribution function when the cumulative parameter is TRUE, and it will be the probability density function when the cumulative parameter is FALSE. Variations of this syntax are <distrib>.DIST.RT and <distrib>.DIST.2T which will be the <i>right-tailed</i> cumulative distribution function and the <i>two-tailed</i> cumulative distribution function, respectively.
<distrib>.INV	Functions with this syntax will be the inverse of the <i>left-tailed</i> cumulative distribution function. Variations of this syntax are <distrib>.INV.RT and <distrib>.INV.2T which will be the inverse of the <i>right-tailed</i> cumulative distribution function and the inverse of the <i>two-tailed</i> cumulative distribution function, respectively.
<function>.S	Functions with this syntax will act on a <i>sample</i> .
<function>.P	Functions with this syntax will act on a <i>population</i> .

3.1.2 Examples

Examples will best demonstrate the new naming scheme:

Example 1:

The existing BINOMDIST function in Excel 2007 and earlier calculates the left-tailed cumulative distribution function when the cumulative parameter is TRUE, and

calculates the probability density function when the cumulative parameter is FALSE. In Excel 2010, there is the BINOM.DIST function which has the exact same functionality as the BINOMDIST function.

Example 2:

In Excel 2007 and earlier there exists the CHIDIST function. In Excel 2010, there is the new CHISQ.DIST.RT function. This is a more appropriate name for the function since CHIDIST actually calculates the right-tailed cumulative distribution function of the chi-squared distribution.

Example 3:

The existing COVAR function in Excel 2007 and earlier calculates the covariance on a population. In Excel 2010, there is the COVARIANCE.P function. This differentiates it from the new COVARIANCE.S function which acts on a sample.

3.2 Function Details

3.2.1 CEILING.PRECISE

Excel 2010 offers a function that calculates the ceiling function that is consistent with user expectations. CEILING.PRECISE always rounds up to the nearest multiple of significance regardless of the sign of the number parameter. The absolute value of significance is always used in calculations.

Syntax: CEILING.PRECISE(number, [significance])

3.2.2 CONFIDENCE.NORM/CONFIDENCE.T

There are two functions which calculate the confidence interval in Excel 2010. CONFIDENCE.NORM calculates the confidence interval on a population, assuming it is normally distributed. CONFIDENCE.T calculates the confidence interval based on the Student's t distribution.

Syntax: CONFIDENCE.NORM(alpha,standard_dev,size)
 CONFIDENCE.T(alpha,standard_dev,size)

3.2.3 COVARIANCE.S/COVARIANCE.P

There are two versions of the covariance function in Excel 2010. COVARIANCE.S calculates the sample covariance, while COVARIANCE.P calculates the population covariance.

Syntax: COVARIANCE.S(array1,array2)
 COVARIANCE.P(array1,array2)

3.2.4 FLOOR.PRECISE

Excel 2010 offers a function that calculates the ceiling function that is consistent with user expectations. FLOOR.PRECISE always rounds down to the nearest multiple of

significance regardless of the sign of the number parameter. The absolute value of significance is always used in calculations.

Syntax: FLOOR.PRECISE(number, [significance])

3.2.5 MODE.SNGL/MODE.MULT

There are two functions which calculate the mode of a range of values in Excel 2010. Both functions return the most frequently occurring value in an array or range of data. MODE.SNGL returns only one mode value even if there are multiple modes while MODE.MULT returns more than one result if there are multiple modes. Because this function returns an array of values, it must be entered as an array formula.

Syntax: MODE.SNGL(number1,number2,...)

MODE.MULT(number1,number2,...)

3.2.6 PERCENTILE.INC/PERCENTILE.EXC and QUARTILE.INC/QUARTILE.EXC

There are two version of the percentile function and two versions of the quartile function in Excel 2010. PERCENTILE.INC and QUARTILE.INC both use algorithm #6 by Weibull and Gumbel in Hyndman and Fan² which allows a percentile value in the range of 0 to 1 inclusive. PERCENTILE.EXC and QUARTILE.EXC use Excel's current algorithm which allows a percentile value in the range of 0 to 1 exclusive.

Syntax: PERCENTILE.INC(array,k)

QUARTILE.INC(array,quart)

PERCENTILE.EXC(array,k)

QUARTILE.EXC(array,quart)

3.2.7 PERCENTRANK.INC/PERCENTRANK.EXC

In Excel 2010 there are two functions which return the rank of a value in a data set as a percentage of the data set. PERCENTRANK.INC returns percentage in the range of 0 to 100 inclusive while PERCENTRANK.EXC returns a percentage in the range of 0 to 100 exclusive.

Syntax: PERCENTRANK.INC(array,x,significance)

PERCENTRANK.EXC(array,x,significance)

3.2.8 RANK.EQ/RANK.AVG

There are two versions of the rank function in Excel 2010. In the presence of duplicate numbers in a range, RANK.EQ uses a downward rank method and gives

² Hyndman, R. J. and Fan, Y. (1996) Sample quantiles in statistical packages, *American Statistician*, 50, 361–365.

duplicate numbers the same rank, while RANK.AVG gives the average rank of the duplicate numbers.

Syntax: RANK.EQ(number,ref,order)
 RANK.AVG(number,ref,order)

3.2.9 STDEV.S/STDEV.P

There are two functions which calculate the standard deviation in Excel 2010. STDEV.S calculates the standard deviation based on a sample, while STDEV.P calculates the standard deviation based on a population.

Syntax: STDEV.S(number1[,number2,..])
 STDEV.P(number1[,number2,..])

3.2.10 VAR.S/VAR.P

There are two functions which calculate the variance in Excel 2010. VAR.S calculates the variance based on a sample, while VAR.P calculates the standard deviation based on a population.

Syntax: VAR.S(number1[,number2,..])
 VAR.P(number1[,number2,..])

4 References

- [1] E. W. Cheney, *Introduction to Approximation Theory*, 2nd edition, American Mathematical Society, 2000.
- [2] IEEE Standards Committee 754, *IEEE Standard for binary floating-point arithmetic, ANSI/IEEE Standard 754-1985*, Institute of Electrical and Electronics Engineers, Los Alamitos, CA, USA, 1985.
- [3] W. Gautschi, A computational procedure for incomplete gamma functions, *ACM Transactions on Mathematical Software*, 5, 466–481, 1979.
- [4] A. R. Didonato, A. Morris, Computation of the incomplete gamma function ratios and their inverse, *ACM Transactions on Mathematical Software*, 12, 377–393, 1989.
- [5] J. M. Blair, C. Edwards, J. Johnson, Rational chebyshev approximations for the inverse of the error function, *Mathematics of Computation*, 30, 827–830, 1976.
- [6] N. M. Temme, On the computation of the incomplete gamma functions for large values of the parameters, In: *Algorithms for Approximation*, J Mason, M Cox (eds.), vol. 10, Oxford University Press, New York, 479–489, 1987.
- [7] N. M. Temme, Asymptotic inversion of the incomplete gamma functions, *Mathematics of Computation*, 58, 755–764, 1992.
- [8] N. M. Temme, The asymptotic expansion of the incomplete gamma functions, *SIAM Journal on Mathematical Analysis*, 10, 757–766, 1979.
- [9] W. Gautschi, The incomplete gamma functions since Tricomi, In: *Tricomi's Ideas and Contemporary Applied Mathematics*, Atti dei Convegni Lincei, vol. 147, Accademia Nazionale dei Lincei, Roma, 203–237, 1998.
- [10] R. W. Abernathy, R. Smith, Applying series expansion to the inverse beta distribution to find percentile for the F distribution, *ACM Transactions on Mathematical Software*, 19, 474–480, 1993.
- [11] G. W. Hill, A. Davis, Generalized asymptotic expansions of the Cornish fisher type, *Annals of Mathematical Statistics*, 39, 1264–1273, 1968.
- [12] B. L. Shea, Algorithm AS 239, Chi Squared and incomplete gamma integral, *Applied Statistics*, 37, 466–473, 1988.
- [13] M. Abramowitz, I. A Stegun, *Handbook of Mathematical Functions*, 10th edition, Dover Publications, New York, 1972.
- [14] P. D. Haines, A closed form approximation for calculating the percentage points of the F and t distributions, *Applied Statistics*, 37, 95–100, 1988.
- [15] C. Loader, Fast and accurate computation of binomial probabilities, Working Paper, 2002.

- [16] N. Temme, Asymptotic inversion of the incomplete beta function, *Journal of Computational and Applied Mathematics*, 41, 145–157, 1992.
- [17] L. Shenton, Inequalities for the normal integral including a new continued fraction, *Biometrika*, 41, 177–189, 1954.
- [18] M. Wichura, Algorithm AS 241, The percentage points of the normal distribution, *Applied Statistics*, 37, 477–484, 1987.
- [19] A. Didonato, A. Morris, Algorithm 708 Significant digit computation of the incomplete beta function ratio, *ACM Transactions on Mathematical Software*, 18, 360–373, 1992.
- [20] D. Best, D. Roberts, Algorithm AS 91, The percentage points of the chi square distribution, *Applied Statistics*, 24, 385–388, 1975.
- [21] R. B Paris, A uniform asymptotic expansion for the incomplete gamma function, *Journal of Computational and Applied Mathematics*, 148, 323–339, 2002.
- [22] N. Temme, A set of algorithms for the incomplete gamma functions, *Probability in the Engineering and the Informational Sciences*, 8, 291–307, 1994.
- [23] W. Cody, Rational chebyshev approximations for the error function, *Mathematics of Computation*, 23, 631–637, 1969.
- [24] B. Brown, L. Levy, Certification of algorithm 708, significant-digit computation of the incomplete beta, *ACM Transactions on Mathematical Software*, 20, 393–397, 1994.
- [25] Donald Knuth, *The Art of Computer Programming, Seminumerical Algorithms*, vol. 2, 3rd edition, Addison-Wesley, Reading, MA, US, 1998.
- [26] W. Cody, Algorithm 715, SPECFUN-A portable FORTRAN package of special function routines and test drivers, *ACM Transactions on Mathematical Software*, 19, 22–32, 1993.
- [27] J. Zar, Approximations for the percentage points of the chi squared distribution, *Applied Statistics*, 27, 280–290, 1978.
- [28] Ping Tang, Table driven implementation of the expm1 function IEEE floating point arithmetic, *ACM Transactions on Mathematical Software*, 18, 211–222, 1992.
- [29] G. Pugh, *An Analysis of the Lanczos Gamma Approximation*, PhD dissertation, University of British Columbia, Vancouver, Canada, 2004.
- [30] Ping Tang, Table driven implementation of the logarithm function in IEEE floating point arithmetic, *ACM Transactions on Mathematical Software*, 16, 378–400, 1990.
- [31] W. H Press, B. P Flannery, S. A Teukolsky, W. T Vetterling, *Numerical Recipes in C*, 2nd edition, Cambridge University Press, Cambridge, UK, 1992.
- [32] Paul Godfrey, A note on the computation of the convergent lanczos complex gamma approximation, <http://my.fit.edu/~gabdo/gamma.txt>, 2001.

- [33] B. Doman, An asymptotic expansion for the incomplete beta function, *Mathematics of Computation*, 65, 1283 -1288, 1996.
- [34] Stephen Moshier, *Methods and Programs for Mathematical Functions*, Ellis Horwood, 1998.
- [35] R Core Development Team, *R, A language and environment for statistical computing*, R Foundation for Computing, Vienna, Austria, www.R-project.org, 2007.
- [36] Frontline Systems, Inc., *Risk Solver 7.0 User Guide*, Frontline Systems, Inc., Incline Village, NV, US, 2007.
- [37] Wolfram Functions Site, www.functions.wolfram.com, 2007.
- [38] Netlib Repository, www.netlib.org.
- [39] G. W. Cran, K. Martin, G. Thomas, Remark AS R 19 and Algorithm AS 109, *Applied Statistics*, 26, 111–114, 1977.
- [40] Masden, K., A root finding algorithm based on Newton's method, *BIT*, 13, 1973, 71–75.
- [41] Mekwi, W. R., *Iterative methods for roots of polynomials*, University of Oxford, MSc Thesis, 2001.
- [42] McNamee, J. M., A comparison of methods for accelerating convergence of Newton's method for multiple polynomial roots, *SIGNUM Newsletter*, 1998.
- [43] Zeng, Z., Algorithm 835: A Matlab package for computing polynomial roots and multiplicities, *ACM TOMS*, 30, 2004, 318–336.
- [44] Goldberg, D., *What every computer scientist should know about floating point arithmetic*, *ACM Computing Surveys*, 1991.
- [45] Alefeld, G.E., Potra, F.A. and Shi, Y., Algorithm 748: Enclosing zeros of continuous function, *ACM TOMS*, 21, 1995, 327–344.
- [46] Wu, Trong, An accurate computation of the hypergeometric distribution function, *ACM TOMS*, 19 (1), 1993, 33–43.
- [47] Wichura, M. J., Algorithm AS 241: The Percentage Points of the Normal Distribution. *Applied Statistics*, 37, 1988, 477–484.