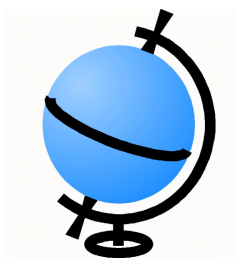


Equator

A Function Calculator



User Guide

Equator

A Function Calculator

Keith B. Oldham

*Trent University
Peterborough, Ontario, Canada*

Jan C. Myland

*Trent University
Peterborough, Ontario, Canada*

Jerome Spanier

*University of California
Irvine, California, U.S.A.*



Springer

New York Berlin Heidelberg London Paris Tokyo

Copyright © 2008 K.B. Oldham, J.C. Myland and J. Spanier. All rights reserved. Printed in U.S.A.

Contents

About <i>Equator</i>	1
Getting started	2
Operating <i>Equator</i>	4
<i>Equator</i> functions and keywords	22
Legalities	37
Envoi	38

About *Equator*

The *Equator* software was conceived as an integral part of the book *An Atlas of Functions*^{*}, though its utility is not limited to users of the book. The *Atlas* is available in both print and electronic formats and full details can be found at www.springer.com. *Equator* is offered for sale as a stand-alone unit in part to benefit those whose access to the *Atlas* is through a portal to the electronic version.

Equator is a software package designed to generate numerical values of over 200 mathematical functions used by engineers, scientists and others. Among the functions made available by *Equator* are such simple ones as cosines and binomial coefficients as well as more elaborate examples such as Bessel functions and elliptic integrals. Consult the table at the back of this booklet for a comprehensive listing of the functions to which *Equator* caters, together with the corresponding keywords recognized by *Equator*.

In addition to its primary goal of providing function values, *Equator* will perform a number of subsidiary tasks.

The notation in *Equator* is identical to that in *An Atlas of Functions*. The algorithms used by *Equator* are not published, but the formulas on which they are based are identified in the *Atlas*.

Equator is easy to use and its operation will soon become as intuitive as using your hand calculator. We hope that you will find it useful.

^{*} Keith B. Oldham, Jan C. Myland and Jerome Spanier, *An Atlas of Functions, Second Edition*, Springer, New York, 2008.

Getting started

Equator must be installed from the CD onto the hard drive of a personal computer running at least Windows XP. The present version is not designed to operate on Macintosh computers or on Unix/Linux systems.

You may need “administrator privileges” in order to install *Equator*, *A Function Calculator*. Memory and processor requirements adequate for running Windows XP will satisfy *Equator*'s needs. During its installation, *Equator* will check if Microsoft's *.Net Framework 2.0* is already installed on your computer. If not, *Equator* will automatically download and install *.Net Framework 2.0* for you.

The following simple steps will enable you to install *Equator*:


- Insert the CD into your computer's disk drive. If the installation program does not immediately start, the “autorun” feature may be disabled on your computer. If so, double-click the CD drive icon under **My Computer**; then double-click the **setup.exe** file.
- If it is absent from your computer, you will be asked to install *.Net Framework 2.0* and you will need to be connected to the internet to do so. Click **Accept** when the *.Net* license agreement appears. The required files will be downloaded and installed. This may take several minutes.
- An **Application Install - Security Warning** screen may appear. Just click the **Install** button.
- Follow the prompts to complete the installation of *Equator*. You will be required to accept a licensing agreement.
- A shortcut to *Equator* will have been placed in the **Start** menu and

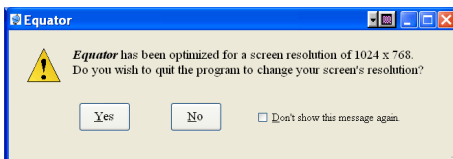
an *Equator* icon will have appeared on your desktop.

- The first time you run *Equator* you will be prompted to input your name, address, and licensing code. The code will be found with the CD that you purchased.

Should you ever need to uninstall *Equator*, double-click **Add or Remove Programs** on the **Windows Control Panel**. Select **Equator** from the list and click **Remove**.

Operating *Equator*

Double-click the Equator icon  on your desktop. If your computer monitor is not set to 1024×768 , you will see the following screen

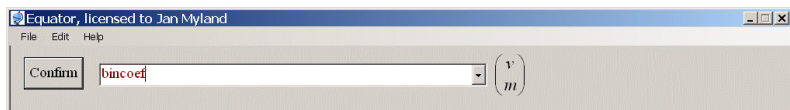


You may, or may not, wish to adjust the resolution. Click the small ☐ checkbox to avoid being reminded of this next time.

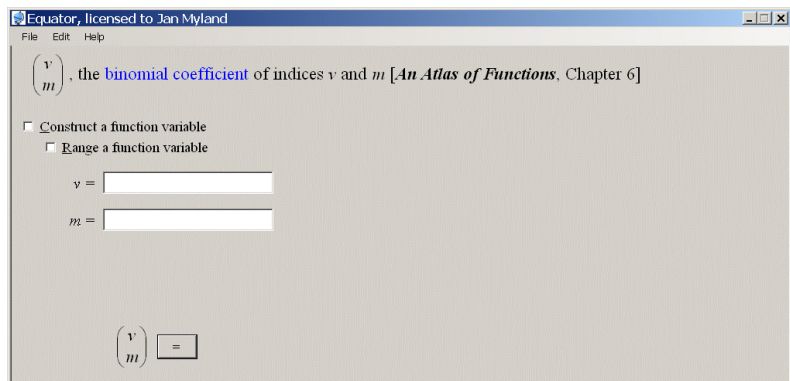
Every function calculable by *Equator* has a **name** and a **keyword** [see the table at the back of this booklet for a complete list]. *Equator*'s opening screen,



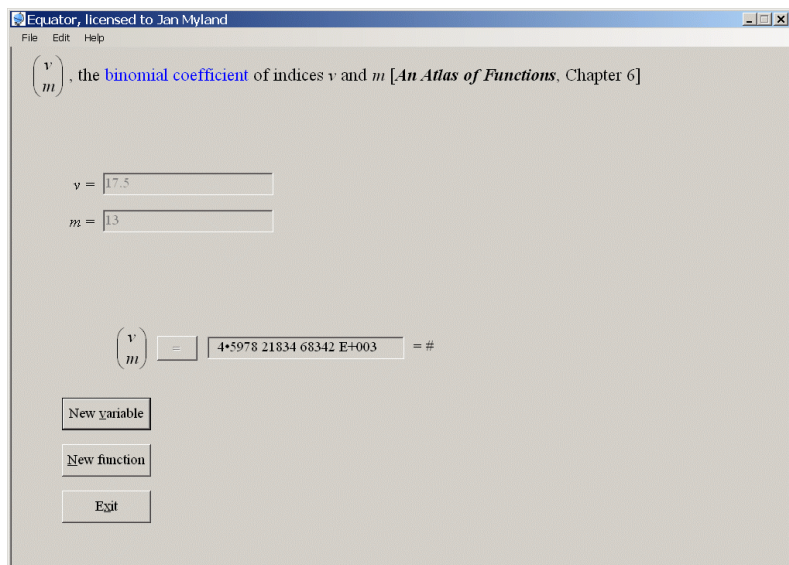
invites you to type either the **name** or the **keyword** into the header box. Alternatively, you may scroll down through the comprehensive alphabetized list to locate the sought **name** or **keyword**, then select and click it. The experienced user will use the typing option and will type the keyword, once familiar with it. As you type, *Equator* will try to anticipate your choice and will also display the corresponding mathematical symbol. For example, if wanting a binomial coefficient you type the keyword "bincoef", the screen will show:



Once the sought function's **name** or **keyword** appears, click the **Confirm** button. In the binomial coefficient example, this brings up the following starting screen:



Because the binomial coefficient is bivariate, there are two input boxes but, depending on the function, there can be as many as four or as few as zero input boxes. For now, ignore the small square checkboxes. Recognize that you are being asked to type numbers into the $v =$ and $m =$ boxes. You might, for example, enter "17.5" and "13" for these variables. Then, on clicking the $=$ button, you will find that *Equator* responds immediately with the answer screen:




At any time prior to clicking , you can return to the start screen by pressing the **Esc** key.


With the calculation complete, there are now three buttons that you might click. One of these allows you to proceed to calculate the value of a different binomial coefficient. Another permits the choice of a new function. The third exits *Equator*.

The basics of *Equator* have now been covered, but there are several additional optional features.

Memories and constants

Notice the “=#” message following the answer in the screenshot above. This indicates that the answer has been stored by *Equator* in

a #-memory, in case you might need to use it in some subsequent “chain” calculation. If, to continue with the previous example, you wished to find the logarithm of the binomial coefficient $\binom{17.5}{13}$, you need only summon the logarithm routine and type “#” into the argument box. Clicking  then generates the logarithm of the binomial coefficient.

The #-memory is ephemeral. Numbers so stored are retained only until the next time  is clicked. Sometimes, however, you may want to use a calculated answer more than once. To preserve an answer, type **ctrl-M**. This stores the answer into a more permanent M-memory. To reuse it, simply type “M” or “m” wherever the stored value is needed. It will remain stored until you overwrite it, or end the *Equator* session.

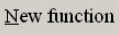
The values of seven mathematical constants and many of the most widely used physical constants are encoded in *Equator*. See the section on *Data input* on page 14.

Variable construction

Knowing the value of some input quantity t , one often needs to find the value, not of the function $f(t)$ of t itself, but of some modified variable, such as $f(\pi t)$ or $f(1+t^2)$. *Equator* caters to this need by providing a facility to construct a desired input variable in accordance with the formula

$$x = wt^p + k$$

The default values of the multiplier w , power p , and the constant k are 1, 1, and 0, but these parameters may be altered by the user.

For example, you might want to compute $\log_2(3^{-\pi})$. Click  and then choose the function **logarithm to any base** (or the keyword **loganybase**), which computes $\log_b(x)$. Enter 2 into the

β box. Next click the checkbox ☐ Construct a function variable and also, to signify that it is argument x that is being constructed rather than the base β , click the \circ button to the left of $x =$. The screen now appears as:

Equator, licensed to Jan Myland

File Edit Help

$\log_{\beta}(x)$, the logarithm to any base, with base β and argument x [*An Atlas of Functions*, Section 25:14]

☒ Construct a function variable

☐ Range a function variable

$\beta =$

$x =$ $= w t^p + k$

$w =$

$t =$

$p =$

$k =$

$\log_{\beta}(x)$

Enter “3” into the t box and “-pi” into the p box, leaving k and w with their default values. Then, on clicking , the screen appears as:

Equator, licensed to Jan Myland

File Edit Help

$\log_{\beta}(x)$, the logarithm to any base, with base β and argument x [*An Atlas of Functions*, Section 25:14]

☒ Construct a function variable

$\beta =$

$x =$ $= w t^p + k$

$w =$

$t =$

$p =$

$k =$

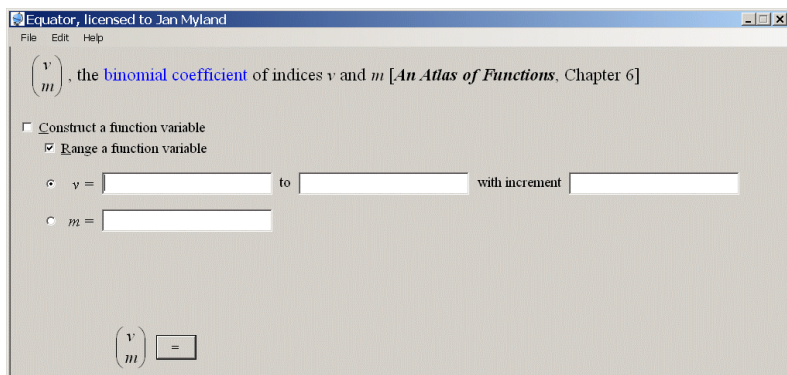
$\log_{\beta}(x)$ $= \#$

Notice that, as well as the answer being generated, the calculated value of the argument x is shown, “grayed out” in the appropriate variable box.

No more than one variable may be constructed in this way.

Variable ranging


You might wish to determine, not just a single value of, say, a binomial coefficient, but a whole range of values. *Equator* provides for this need with its “variable ranging” feature. Test this out by again choosing the binomial coefficient function (keyword **bincoef**) and then clicking the checkbox ☐ **Range a function variable** . The screen

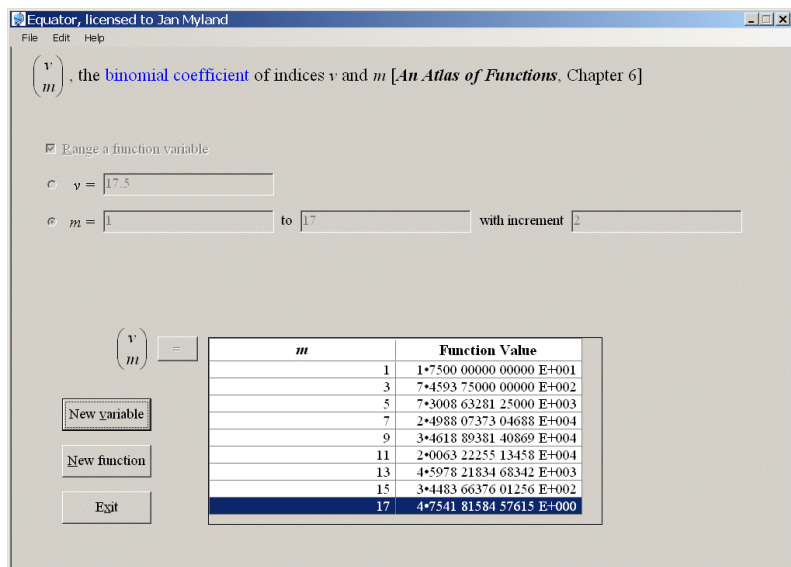


appears. Notice that the ☒ button opposite the first variable, v , is already selected, on the assumption that it is this variable, rather than m , that you wish to range. If this is not your wish, click the button opposite m , converting it from ☐ to ☒ and allowing you to insert values into the appropriate boxes.

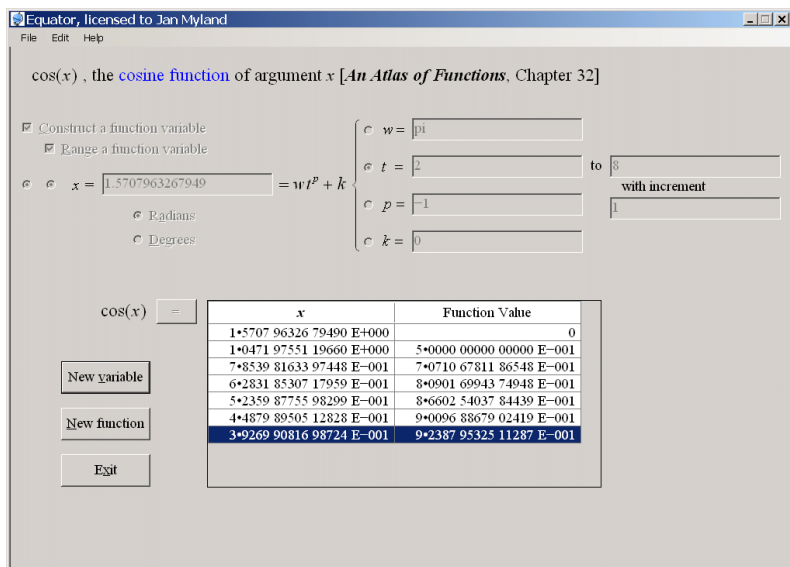
Equator will always evaluate the function at the value of the variable shown in the “to” box, even if this value is not in the chosen

sequence. For example, if you ask for output from 1 to 10 in steps of 2, you will be given $f(1)$, $f(3)$, $f(5)$, $f(9)$ and $f(10)$ values.

If you choose $v = 17.5$ and decide to range m from 1 to 17 in steps of 2 then, after clicking , the screen will be



You can combine the construction and ranging features of *Equator* by ranging on any one of the w , t , p , or k construction tools. Simply click both ☐ checkboxes and the ☐ button alongside whichever one of the tools you choose to range. Imagine, for example, that you wish to find the cosine of several submultiples of π ; perhaps $\cos(\pi/2)$, $\cos(\pi/3)$, $\cos(\pi/4)$, ..., $\cos(\pi/8)$. The next screenshot shows how this may be accomplished and what the answers are.



You cannot construct one variable and range another.

Other uses of ranging

Equator has no plotting capability, but should you wish to graph the output of a ranged calculation, you can easily transfer the data to graphing software, such as *Excel* or *SigmaPlot*®. *Equator*'s **Edit** menu provides a **Copy results to clipboard** facility which is useful for data transfer, not only for graphing, but also for creating hard copy or for another purpose.

Every time you use the ranging feature, *Equator* maintains a running sum of all the values it enters into the rightmost data column. Normally no use is made of this sequential addition and the user is unaware of its occurrence. However, if you wish to see the

sum, type **ctrl-S** and the sum will appear as an additional final entry in the rightmost column. This sum is simply

$$\text{sum} = f(x_1) + f(x_2) + f(x_3) + \cdots + f(x_N)$$

where x_1 is the first argument, and $f(x_N)$ is the last. This facility can be used, for instance, as a means of summing certain finite series. Alternatively, if you type **ctrl-A**, the average, sum/N , appears.

Likewise, *Equator* automatically compiles another kind of sum each time a ranging calculation is performed. The sum in this case is called the trapezoidal area and it is

$$\begin{aligned} \text{trapezoidal} \\ \text{area} &= h \left[\frac{1}{2} f(x_1) + f(x_2) + f(x_3) + \cdots + \frac{1}{2} f(x_{N-1}) \right] \\ &\quad + \frac{x_N - x_{N-1}}{2} [f(x_{N-1}) + f(x_N)] \end{aligned}$$

where h is the increment. To access this value, type **ctrl-T**. The trapezoidal area is output at the bottom of the rightmost column. It provides an approximation to the definite integral

$$\int_{x_1}^{x_N} f(t) dt \approx \text{trapezoidal area}$$

How good this approximation is depends on the properties of the function, as well as on N , the number of data you have specified.

These three facilities are inoperative if the ranged data consists of more than one column, or if any of the data are unavailable or out of range. The value of the sum, average or trapezoidal area is computed by unsophisticated summation; no steps are taken to ensure that all reported digits are significant.

Data input

You may enter a numerical value into *Equator* as an integer or a decimal number in either fixed or floating point (scientific) notation. Thus -765, 0.00573, and 12.987E-78 are all acceptable inputs, but not E-78 without a preceding number. E may be replaced by e, but not by 10. The minus sign, -, must precede a negative input, but + is optional for a positive number. Up to 15 digits may be input; any extra will be ignored. With decimal notation, *Equator* assumes that all digits beyond those input are zeros; thus 17.666 is treated as 17.666 00000 00000. The magnitude of input numbers may range between E-308 and E307.

As alternatives to numbers, the following “number substitutes” may be input:

- the # symbol, representing the value last calculated by *Equator*. If the calculation involved ranging, it is the result of the *final* calculation that is stored in #.
- the M or m symbol, representing a number previously placed in the M-memory.
- any one of the first seven keywords listed in the table overleaf, representing a mathematical constant.
- any one of the twenty keywords listed in the lower section of the table, representing a fundamental physical constant, a conversion factor or an accepted standard. For the most part, these constants are known to less than 15-digit precision, but *Equator* takes no cognizance of this limited precision in calculations involving these constants.
- fractions, such as 17/369 or -pi/8.4E-7. The format must be two numbers (or number substitutes) separated by a solidus, “/”. *Equator* converts the fraction to a decimal number before utilizing it in

calculations, but it does remember that a fraction was input, because this is important in some computations.

Symbol and name	Keyword	Value
π , Archimedes's constant	pi	3.1415 92653 58979
G , Catalan's constant	catalan	0.91596 55941 77219
e , base of natural logarithms	ebase	2.7182 81828 45905
γ , Euler's constant	euler	0.57721 56649 01533
g , Gauss's constant	gauss	0.83462 68416 74073
Z , Apéry's constant	apery	1.2020 56903 15959
ν , golden section	golden	1.6180 33988 74990
c , velocity of light / m s^{-1}	lightc	299 792 458
h , Planck's constant / J s	Planckh	6.6260 690 E-34
k , Boltzmann's constant / J K^{-1}	Boltzmannk	1.3806 504 E-23
L , Avogadro's constant / mol^{-1}	AvogadroL	6.0221 418 E23
F , Faraday's constant / C mol^{-1}	FaradayF	9.6485 340 E4
R , gas constant / $\text{J K}^{-1} \text{mol}^{-1}$	gasR	8.3144 72
ϵ_0 , permittivity of space / F m^{-1}	epsilon0	8.8541 87817 62039 E-12

Symbol and name	Keyword	Value
μ_0 , permeability of space / N A ⁻²	mu0	1.2566 37061 43592 E-6
q_e , electron charge / C	electronq	-1.6021 76487 E-19
m_e , electron mass / kg	electronm	9.1093 8215 E-31
m_p , proton mass / kg	protonm	1.6726 2164 E-27
m_n , neutron mass / kg	neutronm	1.6749 27201 E-27
G , gravitational constant / N m ² kg ⁻²	gravityG	6.67428 E-011
R_∞ , Rydberg constant / m ⁻¹	RydbergR	1.0973 73156 8527 E7
α , fine structure constant	alpha	7.2973 52537 6 E-3
u , atomic mass unit / kg	amu	1.6605 38782 E-27
Da, dalton / kg	amu	1.6605 38782 E-27
eV, electron volt / J	eV	1.6021 76487 E-19
g , standard gravitational acceleration / m s ⁻²	earthg	9.8066 5
T_{std} , standard laboratory temperature / K	standardT	298.15
P_{std} , standard atmospheric pressure / Pa	standardP	101 325

Equator makes no general provision for the input of complex or imaginary numbers, but the [complex number raised to a real power](#), the [square-root function](#), and the [exponential function of complex](#)

[argument](#) routines do accept real and/or imaginary input.

Equator uses the period as the decimal separator. This is the standard in many geographical regions. If, however, you are working in a country where the comma customarily fills the separator role, then (unless that setting in your computer has been changed), you must use a comma when inputting decimal numbers and *Equator* will respond in that system. Otherwise, *Equator* will use the period as the decimal separator and you should too. If you use commas *for any purpose*, when the decimal separator is the period, they are ignored. Conversely, periods will be ignored whenever the decimal separator is the comma. For visual convenience, you may wish to insert spaces within a number; that's okay: *Equator* will ignore them.

Data Output

Answers that are integers smaller in magnitude than $1\text{E}16$ are reported as integers. Function values whose magnitudes lie in the range 0.1 to 1000 are output in fixed-point decimal format. Otherwise a standard floating point notation, exemplified by

$$-1.2345\ 67890\ 12345\text{E}+123$$




is adopted. *Equator* can rarely output nonzero numbers outside the ranges $-1\text{E}-308$ to $-1\text{E}307$ and $1\text{E}-308$ to $1\text{E}307$.


A few of *Equator*'s routines (the [power function](#), [cubic zeros](#), etc.) provide dual outputs: the real and imaginary parts of the complex answer. Some other routines (for example, the [zeros, and associated values, of the Bessel functions](#)) provide two real answers. In these cases, it is the penultimate output (often the real part) that is retained in the #-memory.

Equator strives to generate answers with “15-digit precision”, by which we mean 14 exact digits, with the fifteenth having some uncertainty. This is not always possible. Generally, however, *Equator* is able to detect when precision is likely to have been lost during its calculations and it then curtails the output, such that all the reported digits (of which there may be as few as 3) are significant; that is, only the final reported digit is ever uncertain. If, for any reason, you wish to know what the original 15-digit answer was, just type “#”: the uncurtailed answer will usually be displayed.

If *Equator* cannot confidently provide a function value, even one with only three significant digits, it will report “unavailable” or give some explanation of its failure.

Other Equator features


As do many computer programs, *Equator* allows mouse-less operation. Thus, instead of clicking on the  button, you can type **alt-C**. Likewise, you may navigate between the various input boxes by use of the keyboard **Tab** key. The  or  buttons may be similarly replaced by the **Enter** key.

Equator allows you to input arguments to the trigonometric and **Gudermannian** functions in radians (the default) or degree measure. Simply click the appropriate  button. Likewise you have the choice of radian or degree output from the inverse trigonometric and **inverse Gudermannian** functions and the **elliptic amplitude** routines.

The **rational approximation** routine (keyword **rational**) provides a means of approximating a decimal number by a quotient of two integers. Thus it finds the numerator and denominator in such relationships as

$$\pi \approx \frac{355}{113}$$

The **nearest binary approximant** routine (keyword **bin**) as described on page 21, is another approximation feature of *Equator*.

Equator usually gives an answer almost immediately. Rarely, however, the operation may be slow. The  message at the top of the screen will confirm that calculation is proceeding. If you wish to abort a computation, press the **Esc** key.

Arithmetic processing

So that you will have little need to supplement your *Equator* calculations by using a regular calculator, *Equator* has routines that carry out simple arithmetic operations. Routines for addition, subtraction, multiplication, division, square root and exponentiation exist and have the keywords **+**, **-**, *****, **/**, **sqrt**, and **power**. As well, the construction feature may be used for these purposes. However, *Equator* has another powerful tool to supplement function computation.

The “construction” feature, described earlier, is a valuable way of tailoring your data prior to using them in evaluating the function of interest. There is often a requirement to tailor *Equator*’s output similarly, to suit your particular need. For example, you might have just calculated $\arctan(2x/\pi)$, when what you actually need is $\sqrt{x} \arctan(2x/\pi) - \pi$. To aid in such a supplementary task, *Equator* provides an **arithmetic function** (keyword **arith**). This is a quadrivariate function defined by

$$\text{arith}(W, T, P, K) = WT^P + K$$

where each of W , T , P , and K may be any number or number

substitute. To use the arithmetic function in the example cited, one may choose the four parameters as follows

$$\text{arith}(\#, x, 1/2, -\pi) = \sqrt{x} \arctan(2x/\pi) - \pi$$

However the arithmetic function is more powerful than that example suggests because any one of the four parameters may itself be constructed, opening the possibility of using as many as seven parameters in formulating the output. Any one of those seven may be ranged in the normal way.

Though the arithmetic function is used most often as a sequel to a prior calculation (via the # or M memories), that is not its only application. For instance the quantity $(\pi x + 1)/2x$ may be calculated as $\text{arith}(\pi/2, 1, 1, 0.5/x)$, or in other ways. It may also be used to construct more elaborate input variables than the unaided “construction feature” can handle. As you become familiar with *Equator*, you will find the arithmetic function invaluable.

Accuracy

A whole book could be, and perhaps one day will be, written about the measures that *Equator* adopts to preserve accuracy in calculating function values. Here we merely mention three sources of inaccuracy and methods by which some of these hazards may be mitigated. First note that there are two ways in which inaccuracies may be characterized. One is in terms of the absolute error

$$\text{absolute error} = \hat{f}(x) - f(x)$$

where $f(x)$ is the true value of the function at the argument x and $\hat{f}(x)$ is the calculated value. The second is the relative error

$$\text{relative error} = \frac{\hat{f}(x) - f(x)}{f(x)} \approx \frac{\hat{f}(x) - f(x)}{\hat{f}(x)}$$

A “good” algorithm is one that generates a small absolute error. However, it is the latter measure, the relative error, that determines how many digits can legitimately be reported as a function value. To report 15 digits requires that the relative error be on the order of $1\text{E}-15$. It is the relative error that is usually thought of as reflecting “how accurate” an answer is.

Unless some special procedure can be exploited, *there is inevitably a large inaccuracy associated with computing the value of a function at an argument close to one of its zeros*. The truth of that statement is clear on inspection of previous equation. Whenever x is close to a zero of the function, the denominator will be very small, which places extremely severe demands on the accuracy of computation of $\hat{f}(x)$ if the relative error is to be acceptably small. Of course, you may not even know that your input variable(s) correspond to a function value close to a zero. A small numerical value of the answer will provide a clue that this is the case and explain why *Equator* returns an answer with severely curtailed precision. An “inadequate significance” message may be returned if your argument is extremely close to a zero.

A second issue relates to number systems. Like most computer programs, *Equator* operates with binary numbers. On the other hand, the user inputs a decimal number. Hence, an early step in the operation of *Equator* is the conversion of the user’s decimal number into a binary number. Now, some decimal numbers, such as 73.244 16503 90625, are also exact binary numbers (in this example 1010110.110000000001), whereas another decimal number, such as 73.244 39024 39025, which is superficially similar, is not. The first number converts exactly to a binary number, whereas the second number will be converted by *Equator* into the nearest binary number. This obligatory approximation that *Equator* makes may or

may not engender a serious error in the output. It depends on how “steep” the function is at the argument in question; that is, the inaccuracy depends on the magnitude of $\partial f / \partial x$. Sometimes this derivative can be very large indeed (especially close to a discontinuity). *If the input variable is not an exactly binary number, inevitable inaccuracies may be present in the output, especially if the function is steep.* These inaccuracies will arise, of course, even if *Equator* returns an exact answer – it is the exact answer for a *different* argument. If you suspect that your answers may risk contamination from this cause, and if your particular problem allows, it will generally help if you use only inputs that are exactly binary. For this purpose, *Equator* has a [nearest binary approximant](#) routine (keyword **bin**), described in Section 8:14. This algorithm outputs, in decimal, the nearest exactly binary number, of no more than 15 decimal digits, to the input number. Of course, an alternative is to restrict yourself to input numbers, such as 73, 73.75, or 73.03125, that you recognize on sight as exactly binary.

Precision is lost whenever two numbers of like sign are subtracted, and *Equator* takes cognizance of this loss by outputting only digits that are significant. Numerous such subtractions are especially destructive of precision and hence *there is the danger of severe loss of precision whenever a computation incorporates the modulo operation, or its special case, extraction of the fractional-part*, as a necessary part of the routine. A case in point that arises frequently is the calculation of such functions as $\sin(\pi x)$ when x is large. To evade this problem, *Equator* provides a [reperiodized sine function](#) routine and a [reperiodized cosine function](#) (keywords **sinpi** and **cospi**) that internally multiply the x by π and provide accurate $\sin(\pi x)$ and $\cos(\pi x)$ values, no matter how large x might be.

Equator Functions and Keywords

In addition to recognizing the full **names** of functions and sometimes their synonyms, *Equator* recognizes the following **keywords**, which are sequences of up to ten characters, mnemonically mirroring either the symbol or the name of the function. There follows a comprehensive listing of all *Equator*'s routines and the corresponding keywords. The final column lists the relevant chapter (or appendix) of the book, *An Atlas of Functions*, and often the section too. Please refer to that book (see the footnote on page 1) for further information on the various functions.

Name and symbol	Keyword	Chap or Section
addition , +	+	C:10
Airy Ai function , $Ai(x)$	Ai	56
Airy Bi function , $Bi(x)$	Bi	56
Apery's constant , Z	apery	1:7
Archimedes's constant , π	pi	1:7
arithmetic function , $\text{arith}(W, T, P, K)$	arith	C:10
associated Laguerre polynomial , $L_n^{(m)}(x)$	assocLpoly	23:12
associated Legendre function of the first kind , $P_v^{(m)}(x)$	assocP	59:12
associated Legendre function of the second kind , $Q_v^{(m)}(x)$	assocQ	59:12

Name and symbol	Keyword	Chap or Section
associated value of extremum of Bessel function, $J(j_n^{(k)})$	extremeJ	52:7
atomic mass unit, u	amu	A:3
auxiliary Airy fai function, $\text{fai}(x)$	fai	56:6
auxiliary Airy gai function, $\text{gai}(x)$	gai	56:6
auxiliary cosine integral, $\text{gi}(x)$	gi	38:13
auxiliary cylinder fc function, $\text{fc}_v(x)$	fc	54:14
auxiliary cylinder gc function, $\text{gc}_v(x)$	gc	54:14
auxiliary Fresnel cosine integral, $\text{Fres}(x)$	Fres	39:13
auxiliary Fresnel sine integral, $\text{Gres}(x)$	Gres	39:13
auxiliary sine integral, $\text{fi}(x)$	fi	38:13
Avogadro's (Löschmidt's) constant, L	AvogadroL	A:5
base of natural logarithms, e	ebase	1:7
Bateman G function, $G(v)$	G	44:13
Bateman's confluent function, $\kappa_v(x)$	kappa	48:13
Bernoulli number, B_n	Bnum	4
Bernoulli polynomial, $B_n(x)$	Bpoly	19
Bessel function, $J_n(x)$ or $J_v(x)$	J	52 or 53
(complete) beta function, $B(v, \mu)$	Beta	43:13
beta number, $\beta(v)$	betanum	3

Name and symbol	Keyword	Chap or Section
binomial coefficient, $\binom{v}{m}$	bincoef	6
bivariate eta function, $\eta(v,u)$	eta	64:13
Boltzmann's constant, k	Boltzmannk	A:5
Catalan's constant, G	catalan	1:7
Chebyshev gamma coefficient, $\gamma_j^{(n)}$	Chebygamma	22:5
Chebyshev polynomial of the first kind, $T_n(x)$	Tpoly	22
Chebyshev polynomial of the second kind, $U_n(x)$	Upoly	22
Chebyshev tau coefficient, $\tau_k^{(n)}$	Chebytau	22:6
Clausen's integral, Clausen(x)	Clausen	32:14
common mean, mc(x,y)	mc	61:14
complete beta function, $B(v,\mu)$	Beta	43:13
complete elliptic integral of the first kind, $K(k)$	EllipticK	61
complete elliptic integral of the second kind, $E(k)$	EllipticE	61
complete elliptic integral of the third kind, $\Pi(v,k)$	EllipticPi	61:12
complete gamma function, $\Gamma(v)$	Gamma	43
complex number raised to a real power, $(x+iy)^n$ or $(x+iy)^v$	compower	10:11 or 12:8

Name and symbol	Keyword	Chap or Section
cosecant function, $\csc(x)$	csc	33
cosine function, $\cos(x)$	cos	32
cosine integral, $\text{Ci}(x)$	Ci	38
cotangent function, $\cot(x)$	cot	34
cotangent root, $\rho_n(b)$	rho	34:7
cubic function, x^3+ax^2+bx+c	cubic	16
cubic zeros, $r_3(a,b,c,n)$	r3	16:7
cumulative function for a Boltzmann distribution, $F_{\text{Boltzmann}}(\mu, x)$	FBoltzmann	27:14
cumulative function for a Laplace distribution, $F_{\text{Laplace}}(\mu, \sigma, x)$	FLaplace	27:14
cumulative function for a logistic distribution, $F_{\text{logistic}}(\mu, \sigma, x)$	Flogistic	27:14
cumulative function for a lognormal distribution, $F_{\text{lognormal}}(\mu, \sigma, x)$	Flognormal	27:14
cumulative function for a Lorentz distribution, $F_{\text{Lorentz}}(\mu, \sigma, x)$	FLorentz	27:14
cumulative function for a Maxwell distribution, $F_{\text{Maxwell}}(\mu, x)$	FMaxwell	27:14
cumulative function for a normal distribution, $F_{\text{normal}}(\mu, \sigma, x)$	Fnormal	27:14
cumulative function for a Rayleigh distribution, $F_{\text{Rayleigh}}(\mu, x)$	FRayleigh	27:14

Name and symbol	Keyword	Chap or Section
Dawson's integral, daw(x)	daw	42
Debye function, $\int_0^x \frac{t^n dt}{\exp(t) - 1}$	Debye	3:15
decadic logarithm, $\log_{10}(x)$	log10	25:14
digamma function, $\psi(v)$	digamma	44
dilogarithm, diln(x)	diln	25:12
discrete Chebyshev polynomial, $t_n^{(J)}(x)$	discCheby	22:13
division, \div	/	C:10
double factorial function, $n!!$!!	2:13
electron charge, q_e	electronq	A:5
electron mass, m_e	electronm	A:5
electron volt, eV	eV	A:3
elliptic amplitude, am(k, x)	am	63
elliptic nome, q(k)	q	61:15
entire cosine integral, Cin(x)	Cin	38
entire exponential integral, Ein(x)	Ein	37
entire hyperbolic cosine integral, Chin(x)	Chin	38
entire incomplete gamma function, $\gamma n(v, x)$	gamentire	45

Name and symbol	Keyword	Chap or Section
error function, $\operatorname{erf}(x)$	erf	40
error function complement, $\operatorname{erfc}(x)$	erfc	40
eta number, $\eta(v)$	etanum	3
Euler number, E_n	Enum	5
Euler polynomial, $E_n(x)$	Epoly	20
Euler's constant, γ	euler	1:7
exponential error function complement product, $\exp(x)\operatorname{erfc}(\sqrt{x})$	experfc	41
exponential function, $\exp(x)$	exp	26
exponential function of complex argument, $\exp(x+iy)$	complexp	26:11
exponential integral, $\operatorname{Ei}(x)$	Ei	37
exponential polynomial, $e(x)$	epoly	26:12
exponential theta-four function, $\theta_4(v, t)$	theta4	27:13
exponential theta-one function, $\theta_1(v, t)$	theta1	27:13
exponential theta-three function, $\theta_3(v, t)$	theta3	27:13
exponential theta-two function, $\theta_2(v, t)$	theta2	27:13

Name and symbol	Keyword	Chap or Section
extrema, and their (associated) values, of the Bessel function, $j_n^{(k)}$ and $J(j_n^{(k)})$	Jextrema	52:7
factorial function, $n!$!	2
Faraday's constant, F	FaradayF	A:5
Fibonacci number, $\text{Fib}(n)$	Fibnum	23:14
Fibonacci polynomial, $\text{Fib}_n(x)$	Fibpoly	23:14
fine structure constant, α	alpha	A:5
fractional-part function, $\text{Fp}(x)$	Fp	8
fractional-value function, $\text{frac}(x)$	frac	8
Fresnel cosine integral, $C(x)$	C	39
Fresnel sine integral, $S(x)$	S	39
(complete) gamma function, $\Gamma(v)$	Gamma	43
gas constant, R	gasR	A:5
Gauss hypergeometric function, $F(a,b,c,x)$	F	60
Gauss's constant, g	gauss	1:7
Gegenbauer polynomial, $C_n^{(\lambda)}(x)$	Cpoly	22:12
generalized mean, $m(x,y,n)$	m	61:14
golden section, v	golden	23:14
gravitational constant, G	gravityG	A:5
Gudermannian function, $\text{gd}(x)$	gd	33:15

Name and symbol	Keyword	Chap or Section
Hermite polynomial, $H_n(x)$	Hpoly	24
Hurwitz function, $\zeta(v,u)$	Hurwitz	64
hyperbolic cosecant function, $\operatorname{csch}(x)$	csch	29
hyperbolic cosine function, $\cosh(x)$	cosh	28
hyperbolic cosine integral, $\operatorname{Chi}(x)$	Chi	38
hyperbolic cotangent function, $\coth(x)$	coth	30
hyperbolic secant function, $\operatorname{sech}(x)$	sech	29
hyperbolic sine function, $\sinh(x)$	sinh	28
hyperbolic sine integral, $\operatorname{Shi}(x)$	Shi	38
hyperbolic tangent function, $\tanh(x)$	tanh	30
incomplete elliptic integral of the first kind, $F(k,\varphi)$	ellipF	62
incomplete elliptic integral of the second kind, $E(k,\varphi)$	ellipE	62
incomplete elliptic integral of the third kind, $\Pi(v,k,\varphi)$	ellipPi	62:12
incomplete beta function, $B(v,\mu,x)$	incompBeta	58
integer-part function, $\operatorname{Ip}(x)$	Ip	8
integer-value function, $\operatorname{Int}(x)$	Int	8
inverse cosecant function, $\operatorname{arccsc}(x)$	arccsc	35
inverse cosine function, $\arccos(x)$	arccos	35
inverse cotangent function, $\operatorname{arccot}(x)$	arccot	35

Name and symbol	Keyword	Chap or Section
inverse error function, $\operatorname{inverf}(x)$	inverf	40
inverse Gudermannian function, $\operatorname{invgd}(x)$	invgd	33:15
inverse hyperbolic cosecant function, $\operatorname{arcsch}(x)$	arcsch	31
inverse hyperbolic cosine function, $\operatorname{arcosh}(x)$	arcosh	31
inverse hyperbolic cotangent function, $\operatorname{arcoth}(x)$	arcoth	31
inverse hyperbolic secant function, $\operatorname{arsech}(x)$	arsech	31
inverse hyperbolic sine function, $\operatorname{arsinh}(x)$	arsinh	31
inverse hyperbolic tangent function, $\operatorname{artanh}(x)$	artanh	31
inverse secant function, $\operatorname{arcsec}(x)$	arcsec	35
inverse sine function, $\operatorname{arcsin}(x)$	arcsin	35
inverse tangent function, $\operatorname{arctan}(x)$	arctan	35
Jacobi polynomial, $P_n^{(\nu, \mu)}(x)$	Jacobipoly	22:12
Jacobian elliptic cd function, $\operatorname{cd}(k, x)$	cd	63
Jacobian elliptic cn function, $\operatorname{cn}(k, x)$	cn	63
Jacobian elliptic cs function, $\operatorname{cs}(k, x)$	cs	63
Jacobian elliptic dc function, $\operatorname{dc}(k, x)$	dc	63

Name and symbol	Keyword	Chap or Section
Jacobian elliptic dn function, $\text{dn}(k,x)$	dn	63
Jacobian elliptic ds function, $\text{ds}(k,x)$	ds	63
Jacobian elliptic nc function, $\text{nc}(k,x)$	nc	63
Jacobian elliptic nd function, $\text{nd}(k,x)$	nd	63
Jacobian elliptic ns function, $\text{ns}(k,x)$	ns	63
Jacobian elliptic sc function, $\text{sc}(k,x)$	sc	63
Jacobian elliptic sd function, $\text{sd}(k,x)$	sd	63
Jacobian elliptic sn function, $\text{sn}(k,x)$	sn	63
Kelvin bei function, $\text{bei}(x)$	bei	55
Kelvin ber function, $\text{ber}(x)$	ber	55
Kelvin kei function, $\text{kei}(x)$	kei	55
Kelvin ker function, $\text{ker}(x)$	ker	55
Kummer function, $M(a,c,x)$	M	47
Laguerre polynomial, $L_n(x)$	Lpoly	23
lambda number, $\lambda(v)$	lambdanum	3
Langevin function, $\coth(x) - (1/x)$	Langevin	30:14
Legendre function of the first kind, $P_\nu(x)$	P	59
Legendre function of the second kind, $Q_\nu(x)$	Q	59
Legendre polynomial, $P_n(x)$	Ppoly	21

Name and symbol	Keyword	Chap or Section
Lerch function, $\Phi(x, v, u)$	Lerch	64:12
logarithm to any base, $\log_{\beta}(x)$	loganybase	25:14
logarithm to base 10 of the factorial function, $\log_{10}(n!)$	log10!	2:8
logarithm to base 10 of the gamma function, $\log_{10}\{\Gamma(v)\}$	log10Gamma	43:8
logarithmic factorial function, $\ln(n!)$	ln!	2:8
logarithmic function, $\ln(x)$	ln	25
logarithmic gamma function, $\ln\{\Gamma(v)\}$	lnGamma	43:8
logarithmic integral, $\text{li}(x)$	li	25:13
lower incomplete gamma function, $\gamma(v, x)$	gamlower	45
Macdonald function, $K_v(x)$	K	51
Mittag-Leffler function, $E_{v, \mu}(x)$	Mittag	45:13
modified Struve function, $\mathbb{L}_v(x)$	l	57
modified (hyperbolic) Bessel function, $I_n(x)$ or $I_v(x)$	I	49 or 50
modified spherical Bessel function, $i_n(x)$	i	28:13
modulo function, $v(\text{mod } m)$	mod	8:12
multiplication, \times	*	C:10

Name and symbol	Keyword	Chap or Section
n-fold integral of the error function complement, $i^n \text{erfc}(x)$	inerfc	40:13
nearest binary approximant, $\text{bin}(x)$	bin	8:14
Neumann function, $Y_\nu(x)$	Y	54
neutron mass, m_n	neutronm	A:5
Neville's c theta function, $\mathfrak{G}_c(k, x)$	theta-c	61:15
Neville's d theta function, $\mathfrak{G}_d(k, x)$	theta-d	61:15
Neville's n theta function, $\mathfrak{G}_n(k, x)$	theta-n	61:15
Neville's s theta function, $\mathfrak{G}_s(k, x)$	theta-s	61:15
normally distributed random variates, $\text{normal}(\mu, \sigma, J, s)$	normal	40:14
parabolic cylinder function, $D_\nu(x)$	D	46
permeability of free space, m_0	mu0	A:5
permittivity of free space, ϵ_0	epsilon0	A:5
Planck's constant, h	Planckh	A:5
Pochhammer polynomial, $(x)_n$	Poch	18
polygamma function, $\psi^{(n)}(v)$	polygamma	44
power function, x^n or x^ν	power	10 or 12
probability function for a Boltzmann distribution, $P_{\text{Boltzmann}}(\mu, x)$	PBoltzmann	27:14
probability function for a Laplace distribution, $P_{\text{Laplace}}(\mu, \sigma, x)$	PLaplace	27:14

Name and symbol	Keyword	Chap or Section
probability function for a logistic distribution, $P_{\text{logistic}}(\mu, \sigma, x)$	Plogistic	27:14
probability function for a lognormal distribution, $P_{\text{lognormal}}(\mu, \sigma, x)$	Plognormal	27:14
probability function for a Lorentz distribution, $P_{\text{Lorentz}}(\mu, \sigma, x)$	PLorentz	27:14
probability function for a Maxwell distribution, $P_{\text{Maxwell}}(\mu, x)$	PMaxwell	27:14
probability function for a normal distribution, $P_{\text{normal}}(\mu, \sigma, x)$	Pnormal	27:14
probability function for a Rayleigh distribution, $P_{\text{Rayleigh}}(\mu, x)$	PRayleigh	27:14
proton mass, m_p	protonm	A:5
quadratic function, $ax^2 + bx + c$	quadratic	15
quadratic zeros, $r_2(a, b, c, n)$	r2	15:7
quartic zeros, $r_4(a_3, a_2, a_1, a_0)$	r4	16:12
rational approximants, n/d	rational	8:13
reperiodized cosine function, $\cos(\pi x)$	cospi	32:8
reperiodized sine function, $\sin(\pi x)$	sinpi	32:8
Rydberg constant, R_∞	RydbergR	A:5
sampling function, $\text{sinc}(x)$	sinc	32:13
secant function, $\sec(x)$	sec	33

Name and symbol	Keyword	Chap or Section
sine function, $\sin(x)$	sin	32
sine integral, $\text{Si}(x)$	Si	38
spherical Bessel function, $j_n(x)$	j	32:13
spherical Macdonald function, $k_n(x)$	k	26:13
spherical Neumann function, $y_n(x)$	y	32:13
square-root function, $\sqrt{x+iy}$	sqrt	11
standard atmospheric pressure, P_{std}	standardP	A:6
standard gravitational acceleration, g	earthg	A:6
standard laboratory temperature, T_{std}	standardT	A:6
standard random numbers, $\text{random}(J,s)$	random	40:14
Stirling number of the first kind, $S_n^{(m)}$	Snum	18:6
Stirling number of the second kind, $\sigma_n^{(m)}$	sigmanum	2:14
Struve function, $h_\nu(x)$	h	57
subtraction, $-$	-	C:10
tangent function, $\tan(x)$	tan	34
tangent root, $r_n(b)$	r	34:7
tetragamma function, $\psi^{(2)}(\nu)$	tetragamma	44
Tricomi function, $U(a,c,x)$	U	48
trigamma function, $\psi^{(1)}(\nu)$	trigamma	44
trilogarithm, $\text{triln}(x)$	triln	25:12

Name and symbol	Keyword	Chap or Section
upper incomplete gamma function, $\Gamma(v, x)$	gamupper	45
velocity of light, c	lightc	A:5
Whittaker M function, $M_{v,\mu}(x)$	WhitM	48:13
Whittaker W function, $W_{v,\mu}(x)$	WhitW	48:13
zeros, and their associated values, of the Bessel function, $j_n^{(k)}$ and $J'(j_n^{(k)})$	Jzero	52:7
zeta number, $\zeta(v)$	zetanum	3

Legalities

By installing the *Equator* software, you are agreeing to be bound by End User License Agreement which you will find on the CD. The following is a precis of that agreement:

This software product is protected by copyright and all rights are reserved by the authors and Springer. You are licensed to use this software on a single computer. Copying the software to another medium or format for use on a single computer does not violate the U.S. Copyright Law. Copying the software for any other purpose is a violation of the U.S. Copyright Law.

This software product is provided as is without warranty of any kind, either express or implied, including, but not limited to, the implied warranty of merchantability and fitness for a particular purpose. Neither the authors, nor Springer, nor their dealers or distributors assume any liability for any alleged or actual damages arising from the use of or the inability to use this software. (Some jurisdictions do not allow the exclusion of implied warranties, so the exclusion may not apply to you.)

Envoi

Great care has been exercised in the creation of *Equator*. However, it is too much to hope that errors are absent. The authors would appreciate your notifying them of any errors that you detect or difficulties you may encounter. We would also welcome any suggestions as to additional facilities that might be included in some future edition of *Equator*. We can be contacted by email at: koldham@trentu.ca, jmyland@trentu.ca, or jspanier@uci.edu.

The authors wish to express their sincere thanks to Michelle Johnston for her sedulous assistance in creating some of the algorithms used by *Equator* and for suggesting its name. We also appreciate the help of Sten Englom. Many at Springer have contributed greatly to the success of this project.