

Package ‘mathr’

October 10, 2026

Version 0.1.5

Title Scientific Computation Using R

Description A collection of undergraduate level mathematical routines for quantitative work, covering calculus, distribution functions, random variate generation, linear algebra, differential equations and optimization, sized for one semester. The package is best studied with its source open: every routine is short and written to be read.

Depends R (>= 3.0.0)

Copyright 2018-2026, Kyun-Seop Bae

License GPL-3

Encoding UTF-8

NeedsCompilation no

URL <https://github.com/ksbae/mathr>

BugReports <https://github.com/ksbae/mathr/issues>

Author Kyun-Seop Bae [aut, cre]

Maintainer Kyun-Seop Bae <k@acr.kr>

Repository CRAN

Date/Publication 2026-10-10 06:30:02 UTC

Contents

betai	4
Bin2Dec	5
ChkFx	6
Chol	7
Choose	8
conv	9
conv0	9
Dbeta	10
Dbinom	11
dc	12

Dchisq	13
Dec2Bin	13
deconv	14
deconv0	15
DENORM	16
Deriv0	17
Deriv1	17
Deriv2	18
Df	19
Dgamma	20
Dlnorm	21
Dnorm	21
Dpois	22
Dt	23
ellipRange	24
ellipse	25
erf	26
erfc	27
erfccheb	28
evJacobi	29
EXP	29
GAMMA	30
gammln	31
gammq	32
gammq	33
GenD	34
GetP	34
GQuad8	35
Grad	36
Hessian	37
invbetai	37
inverfc	38
invgammq	39
LDLT	40
LGAMMA	41
LOG	42
MachEps	43
MachEps2	43
MachEps3	44
MachEps4	45
mlr	45
NashSVD	47
NORM	47
Optim0	48
OuterProd	49
Pbeta	50
Pbinom	51
Pchisq	51

Pf	52
Pgamma	53
Plnorm	54
Pnorm	54
PolyNom3	55
Ppois	56
Pt	57
Qbeta	58
Qbinom	58
Qchisq	59
Qf	60
Qgamma	61
Qlnorm	61
Qnorm	62
Qpois	63
Qt	64
Rbeta	65
Rexp	66
Rgamma	66
Rgamma0	67
RK4	68
Rmvn	69
Rnorm	70
romb	71
Round	72
run.p	73
Run.test	74
runifLC	74
sft	75
simps13	76
simps38	77
SQRT	78
SymDet	78
SymInv	79
SymSol	80
tableFactorial	81
trapez0	81
trapez1	82
VMmin	83

betai	<i>Incomplete beta function</i>
-------	---------------------------------

Description

Incomplete beta function with parameter a and b

Usage

```
betai(a, b, x)
```

Arguments

a	shape constant, distribution parameter
b	shape constant, distribution parameter
x	to which it integrates

Details

This calls `betacf` and `gammln`. Ref: `gsl_sf_beta_inc` of the GNU Scientific Library, `specfunc/beta_inc.c`. This function works a value at a time: a, b and x must each be a single number. Use `sapply` for a vector.

Value

integrated value

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```
betai(2, 3, 0.4)  
pbeta(0.4, 2, 3)
```

Bin2Dec

*Binary to Decimal Format***Description**

Transforms a binary IEEE 754 format value stored in an array to a decimal value.

Usage

```
Bin2Dec(b)
```

Arguments

b a binary value stored in an array with IEEE 754 format

Details

IEEE 754 defines how a computer represents a floating point number in binary format. This transforms binary format to decimal value.

Value

A decimal value with attribute "Expression"

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

IEEE 754

Examples

```
Bin2Dec(rep(0, 32))                      # +0.0
Bin2Dec(c(1, rep(0, 31)))                # -0.0
Bin2Dec(c(0, rep(1, 8), rep(0, 23)))    # +Inf
Bin2Dec(c(1, rep(1, 8), rep(0, 23)))    # -Inf
Bin2Dec(c(0, rep(1, 8), rep(1, 23)))    # NaN
Bin2Dec(c(1, rep(1, 8), rep(1, 23)))    # NaN
Bin2Dec(c(1, rep(1, 30), 0))            # NaN

Bin2Dec(c(0, 0,1,1,1,1,1,1,1,1,1, 0,0,0,0, rep(0,48))) # +1
Bin2Dec(c(1, 0,1,1,1,1,1,1,1,1,1, 0,0,0,0, rep(0,48))) # -1
Bin2Dec(c(0, 0,1,1,1,1,1,1,1,1,1, 1,0,0,0, rep(0,48))) # 1.5
Bin2Dec(c(0, 1,0,0,0,0,0,0,0,0,0, 0,0,0,0, rep(0,48))) # 2
Bin2Dec(c(0, 1,0,0,0,0,0,0,0,0,0,1, 1,0,1,0, rep(0,48))) # 6.5
```

ChkFx

*Check function input and output types***Description**

Check the input and output of a function be a scalar, a vector, or a matrix.

Usage

```
ChkFx(func, x)
```

Arguments

func	function to be checked
x	given input

Details

This is to check the types (scalar, vector, matrix) of a function input and output.

Value

CaseNo	Case number
minInput	minimal input type
minOutput	minimal output type
Given	type of given value
Repeat	repeated evaluation of the function
Gradient	type of gradient output

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx0 = function(x) {
  if (length(x) > 1) return(NULL)
  return(x*x)
}

fx1 = function(x) x*x # independent each other (elementwise operation only)
fx2 = function(x) sum(x)
fx3 = function(x) c(sum(x), mean(x)) # c(mean(x), sd(x))

fx4 = function(x) {
  if (is.vector(x)) { Result = sum(x)
  } else if (is.matrix(x)) { Result = rowSums(x)
  }
```

```
    } else { Result = NULL }
    return(Result)
}

fx5 = function(x) {
  if (is.vector(x)) { Result = c(sum(x), mean(x))
  } else if (is.matrix(x)) { Result = cbind(rowSums(x), rowMeans(x))
  } else { Result = NULL }
  return(Result)
}

ChkFx(fx1, 3)
ChkFx(fx1, 1:3)
ChkFx(fx2, 3)
ChkFx(fx2, 1:3)
ChkFx(fx3, 3)
ChkFx(fx3, 1:3)
ChkFx(fx4, 1:3)
ChkFx(fx4, matrix(1:9, nrow=3))
ChkFx(fx5, 1:3)
ChkFx(fx5, matrix(1:9, nrow=3))
```

Chol

Cholesky Decomposition

Description

Cholesky decomposition of a symmetric matrix

Usage

Chol(M)

Arguments

M a symmetric matrix of real values

Details

Input should be a symmetric matrix of real values. This is one-to-one transformation.

Value

the lower triangular matrix L such that $M = L L'$

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
M <- matrix(c(4, 2, 2, 3), 2, 2)
L <- Chol(M)
L
L %*% t(L)           # back to M
```

Choose

Combination function

Description

Another simple implementation of combination function like choose

Usage

```
Choose(n, r)
```

Arguments

n	total count
r	count to choose

Details

This is a simple version of the combination function that does not overflow easily. This function works a value at a time: n and r must each be a single number. Use sapply for a vector.

Value

combination of choosing r cases from n cases.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Choose(10, 5)
Choose(100, 50)
Choose(1000, 500)
Choose(2000, 1000)
```

conv	<i>Linear convolution using fft</i>
------	-------------------------------------

Description

Convolute two given sequences and return the real part only.

Usage

```
conv(x, y)
```

Arguments

x	Input sequence
y	Response sequence to unit impulse

Details

One is input and the other is unit response sequence. The lengths of the two sequences should be the same. The order is not important.

Value

Real part of convolution result.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
x <- c(1, 0.5, 0.25, 0, 0, 0)
y <- c(1, 2, 3, 0, 0, 0)
conv(x, y)                # linear convolution
round(deconv(conv(x, y), x), 10) # deconv undoes it
```

conv0	<i>Circular convolution using fft</i>
-------	---------------------------------------

Description

Convolute two given sequences.

Usage

```
conv0(x, y)
```

Arguments

x	Input sequence
y	Response sequence to unit impulse

Details

One is input and the other is unit response sequence. The lengths of the two sequences should be the same. The order is not important.

Value

convolution result.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
x <- c(1, 0.5, 0.25, 0.125)
y <- c(1, 2, 3, 4)
Re(conv0(x, y))           # circular convolution, by fft
round(Re(deconv0(conv0(x, y), x)), 10) # deconv0 undoes it
```

Dbeta

Density of beta distribution

Description

Calculate the density of a beta distribution

Usage

```
Dbeta(x, alph, bet)
```

Arguments

x	x value for a beta distribution
alph	the shape parameter alpha of a beta distribution
bet	the shape parameter beta of a beta distribution

Details

It calculates the density for x value of a beta distribution. This function works a value at a time: x, alph and bet must each be a single number. Use sapply for a vector.

Value

density for x

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Dbeta(0.3, 2, 3)
dbeta(0.3, 2, 3)
```

Dbinom

Density of binomial distribution

Description

Calculate the mass for k of a binomial distribution

Usage

```
Dbinom(k, n, pe)
```

Arguments

k	count k of wanted event for a binomial distribution
n	total number of trials n of a binomial distribution
pe	the probability of wanted event for a binomial distribution

Details

It calculates the probability for count k of a binomial distribution. This function works a value at a time: k, n and pe must each be a single number. Use `sapply` for a vector.

Value

probability for k

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Dbinom(3, 10, 0.3)
dbinom(3, 10, 0.3)
```

dc

*Deconvolution of two sequences at different time points***Description**

Deconvolute two given sequences measured at different time points.

Usage

```
dc(t1, c1, t2, c2)
```

Arguments

t1	time points of sequence 1
c1	measurement values of sequence 1, to be deconvoluted, Response values
t2	time points of sequence 2
c2	measurement values of sequence 2, input sequence or unit response sequence

Details

Two measurement sequences with two different time point sequences. The order matters.

Value

Deconvolution result: input sequence or unit response sequence.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
t1 <- 0:5
c1 <- c(0, 1, 3, 4, 3, 1)      # response
t2 <- 0:5
c2 <- c(0, 1, 1, 0, 0, 0)      # unit response
dc(t1, c1, t2, c2)
```

Dchisq	<i>Density of chi-square distribution</i>
--------	---

Description

Calculate the density of a chi-square distribution

Usage

```
Dchisq(x2, nu)
```

Arguments

x2	chi-square value for a chi-square distribution
nu	degrees of freedom nu of a chi-square distribution

Details

It calculates the density for chi-square value of a chi-square distribution. This function works a value at a time: x2 and nu must each be a single number. Use `sapply` for a vector.

Value

density for x2

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Dchisq(3, 5)
dchisq(3, 5)
```

Dec2Bin	<i>Decimal to Binary Format Vector</i>
---------	--

Description

Transforms a decimal value to a binary IEEE 754 format value stored in an array.

Usage

```
Dec2Bin(x, Double=TRUE)
```

Arguments

x	a decimal value
Double	set this FALSE if you want single precision.

Details

IEEE 754 defines how a computer represents a floating point number in binary format. This transforms a decimal value to a binary format array. This function works a value at a time: x and Double must each be a single value. Use sapply for a vector.

Value

An array containing the binary value.

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

IEEE 754

Examples

```
Dec2Bin(1)
Dec2Bin(-1)
Dec2Bin(1.5)
Dec2Bin(2)
Dec2Bin(6.5)

Dec2Bin(1, FALSE)
Dec2Bin(-1, FALSE)
Dec2Bin(1.5, FALSE)
Dec2Bin(2, FALSE)
Dec2Bin(6.5, FALSE)
```

deconv

Linear deconvolution using fft

Description

Deconvolute two given sequences, and return the real part only.

Usage

```
deconv(z, x)
```

Arguments

z	Response sequence to be deconvoluted
x	Input or Unit response sequence

Details

The lengths of the two sequences should be the same. The order matters.

Value

Real part of deconvolution result.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
# conv truncates the linear convolution back to the input length, so
# pad the tails with zeros to keep the whole of it. Then deconv
# recovers the input exactly.
x <- c(1, 0.5, 0.25, 0, 0, 0)
y <- c(1, 2, 3, 0, 0, 0)
z <- conv(x, y)
z
round(deconv(z, x), 10)          # back to y
```

deconv0

Circular deconvolution using fft

Description

Deconvolute two given sequences.

Usage

```
deconv0(z, x)
```

Arguments

z	Response sequence to be deconvoluted
x	Input or Unit response sequence

Details

The lengths of the two sequences should be the same. The order matters.

Value

Deconvolution result.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
# deconv0 is the inverse of conv0: both work on the circular
# convolution, so the round trip is exact.
x <- c(1, 0.5, 0.25, 0.125)
y <- c(1, 2, 3, 4)
z <- conv0(x, y)
round(Re(deconv0(z, x)), 10)      # back to y
```

DENORM

Denormalize

Description

From normalized number to decimal number

Usage

```
DENORM(Norm)
```

Arguments

Norm Normalized number, a vector of length 2.

Details

Ref. Plauger PJ The Standard C Library. 1992.

Value

Norm[1] * 2^Norm[2]

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
DENORM(c(0.625, 4))      # 0.625 * 2^4
DENORM(NORM(pi))
```

`Deriv0`*First derivative, version 0*

Description

Get first derivative of scalar-input scalar-valued function using Richardson extrapolation 4th order.

Usage

```
Deriv0(fx, x)
```

Arguments

<code>fx</code>	scalar valued function
<code>x</code>	x point at which derivative is calculated

Details

This simplest implementation of 4th order Richardson extrapolation for derivative calculation uses a 4x4 matrix.

Value

derivative value

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx = function(x) x*x
Deriv0(fx, 1) # 2
```

`Deriv1`*First derivative, version 1*

Description

Get first derivative of scalar-input scalar-valued function using Richardson extrapolation 4th order.

Usage

```
Deriv1(fx, x)
```

Arguments

fx	scalar valued function
x	x point at which derivative is calculated

Details

This implementation of 4th order Richardson extrapolation for derivative calculation, uses a vector of length 4.

Value

derivative value

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx = function(x) x*x
Deriv1(fx, 1) # 2
```

Deriv2

First derivative, version 2

Description

Get first derivative of vector-input (multi-variable) scalar-valued function using Richardson extrapolation 4th order.

Usage

```
Deriv2(fx, x)
```

Arguments

fx	vector-input (multi-variable) scalar-valued function
x	vector x at which gradients are calculated

Details

This implementation of 4th order Richardson extrapolation for derivative calculation, uses 4 vectors.

Value

gradient values

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx2 = function(x) sum(x*x)
Deriv2(fx2, c(1,1)) # 2, 2
```

Df

Density of F distribution

Description

Calculate the density of a F distribution

Usage

```
Df(f, nu1, nu2)
```

Arguments

f	f value for an F distribution
nu1	numerator degrees of freedom nu1 of an F distribution
nu2	denominator degrees of freedom nu2 of an F distribution

Details

It calculates the density for f value of an F distribution. This function works a value at a time: f, nu1 and nu2 must each be a single number. Use `sapply` for a vector.

Value

density for f

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Df(2, 4, 9)
df(2, 4, 9)
```

Dgamma

Density of gamma distribution

Description

Calculate the density of a gamma distribution

Usage

```
Dgamma(x, alph, bet=1)
```

Arguments

x	x value for a gamma distribution
alph	the shape parameter alpha of a gamma distribution
bet	the rate parameter beta of a gamma distribution

Details

It calculates the density for x value of a gamma distribution. This function works a value at a time: x, alph and bet must each be a single number. Use `sapply` for a vector.

Value

density for x

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Dgamma(2, 3)          # bet defaults to 1
Dgamma(2, 3, 2)        # bet is a rate
dgamma(2, 3, rate=2)
```

Dlnorm	<i>Density of log-normal distribution</i>
--------	---

Description

Calculate the density of a log-normal distribution

Usage

```
Dlnorm(x, mu=0, sig=1)
```

Arguments

x	x value for a log-normal distribution
mu	the mean mu of the logarithm for a log-normal distribution
sig	the standard deviation sigma of the logarithm for a log-normal distribution

Details

It calculates the density for x value of a log-normal distribution. x and sig must each be a single number, not a vector. Use `sapply` over x for several values.

Value

density for x

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Dlnorm(1)
dlnorm(1)
```

Dnorm	<i>Density of normal distribution</i>
-------	---------------------------------------

Description

Calculate the density of a normal distribution

Usage

```
Dnorm(x, mu=0, sig=1)
```

Arguments

x	x value for a normal distribution
mu	the mean mu of a normal distribution
sig	the standard deviation sigma of a normal distribution

Details

It calculates the density for x value of a normal distribution. sig must be a single number, not a vector.

Value

density for x

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Dnorm(0)
Dnorm(1.96, mu=0, sig=1)
dnorm(1.96)      # base R, for comparison
```

Dpois	<i>Density of Poisson distribution</i>
-------	--

Description

Calculate the mass for n of a Poisson distribution

Usage

```
Dpois(n, lam)
```

Arguments

n	count n for a Poisson distribution
lam	mean lambda of a Poisson distribution

Details

It calculates the mass for n of a Poisson distribution. This function works a value at a time: n and lam must each be a single number. Use sapply for a vector.

Value

mass (probability) for n

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Dpois(3, 4)
dpois(3, 4)
```

Dt	<i>Density of t distribution</i>
----	----------------------------------

Description

Calculate the density of a t distribution

Usage

```
Dt(t, nu, mu=0, sig=1)
```

Arguments

t	t value for a t distribution
nu	degrees of freedom
mu	the location parameter mu of a t distribution
sig	the scale parameter sigma of a t distribution

Details

It calculates the density for t value of a t distribution. nu and sig must each be a single number, not a vector.

Value

density for t

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Dt(1.5, 7)
dt(1.5, 7)
```

ellipRange	<i>Ranges of an Ellipse</i>
------------	-----------------------------

Description

Ranges of an ellipse

Usage

```
ellipRange(center=c(0, 0), radius=c(2, 1), alpha=0)
```

Arguments

center	coordinate of center point
radius	length of long and short axes
alpha	rotation angle of long axis

Details

This returns ranges of x and y axes.

Value

Column vectors of x and y ranges

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
ellipRange()  
ellipRange(c(0, 0), c(3, 2), pi/4)  
ellipRange(c(1, 1), c(3, 2), pi/4)  
ellipRange(c(1, 1), c(3, 2), pi/6)
```

`ellipse`*Ellipse*

Description

Plot ellipse and return coordinates.

Usage

```
ellipse(center=c(0, 0), radius=c(2, 1), alpha=0, npoints=100, add=FALSE, ...)
```

Arguments

<code>center</code>	coordinate of center point
<code>radius</code>	length of long and short axes
<code>alpha</code>	rotation angle of long axis
<code>npoints</code>	number of points used to plot
<code>add</code>	whether to plot on an active device
<code>...</code>	arguments to be passed to lines or plot

Details

This plots an ellipse and returns coordinates of points used to plot. `add` must be a single value, not a vector.

Value

coordinates of points used to plot.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
ellipse(asp=1)

ellipse(c(0, 0), c(3, 2), pi/4, asp=1)

ellipse(c(1, 1), c(3, 2), pi/4, asp=1)
points(1, 1, pch="+")
abline(v=1, h=1, lty=3)
```

erf	<i>Error function</i>
-----	-----------------------

Description

Error function.

Usage

`erf(x)`

Arguments

`x` a real value

Details

This calls `gammp`.

Ref: $\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$, the relation the GNU Scientific Library uses between `gsl_sf_erf` and `gsl_sf_gamma_inc_P`. This function works a value at a time: `x` must be a single number. Use `sapply` for a vector.

Value

Integration value $\frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz
M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```
erf(1)
2*pnorm(sqrt(2)) - 1           # base R, for comparison
```

erfccheb*Complementary error function, kept for compatibility*

Description

Retained for backward compatibility. Forwards to erfc.

Usage

erfccheb(z)

Arguments

z a real value

Details

Retained for backward compatibility. Forwards to erfc, which now uses the incomplete gamma relation over the whole range. This function works a value at a time: z must be a single number. Use sapply for a vector.

Value

complementary error function value, the same as erfc(z)

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz
M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```
erfccheb(1)                      # same as erfc(1)
```

evJacobi	<i>Spectral Decomposition of a Real Symmetric Matrix by Jacobi Algorithm</i>
----------	--

Description

Eigen values and vectors of a Real Symmetric Matrix by Jacobi Algorithm

Usage

```
evJacobi(A)
```

Arguments

A a symmetric matrix of real values

Details

Input should be a symmetric matrix of real values.

Value

values	eigen values
vectors	eigen vectors

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
A <- matrix(c(2, 1, 1, 2), 2, 2)
r <- evJacobi(A)
r$values
eigen(A)$values
```

EXP	<i>Exponential</i>
-----	--------------------

Description

Exponential of a real value.

Usage

```
EXP(x)
```

Arguments

x a single real number to be exponentiated

Details

Current CPUs support this function at the instruction level, so this implementation is for study rather than for speed. Ref. Plauger PJ The Standard C Library. 1992. This function works a value at a time: x must be a single number. Use `sapply` for a vector.

Value

Exponential value of a real value.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
EXP(1)
exp(1)
EXP(-3)
sapply(c(1, 2, 3), EXP)  # one value at a time
```

GAMMA

Gamma Function

Description

This gamma function is a generalization of factorial working with real numbers. This function uses Lanczos' approximation and is fairly short.

Usage

```
GAMMA(z)
```

Arguments

z real number

Details

If z is a positive integer, $GAMMA(z) = (z - 1)!$. If z is zero or a negative integer, it returns +Inf. Ref: Lanczos g=7 coefficients as used by the GNU Scientific Library, `gsl_sf_lngamma`. This function works a value at a time: z must be a single number. Use `sapply` for a vector.

Value

Gamma function real value

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
GAMMA(0)
GAMMA(1) # 0! = 1
GAMMA(2) # 1! = 1
GAMMA(171) # 170! = 7.257416e+306
format(GAMMA(0.5), digits=22) # = sqrt(pi) = 1.772453850905516
format(gamma(0.5), digits=22)
format(sqrt(pi), digits=22)
format(GAMMA(1.5), digits=22)
format(gamma(1.5), digits=22)
```

gammln

Log gamma function

Description

Log gamma function

Usage

gammln(xx)

Arguments

xx a real positive number to be gammln-ed

Details

Ref: Lanczos g=7 coefficients as used by the GNU Scientific Library, gsl_sf_lngamma. This function works a value at a time: xx must be a single number. Use sapply for a vector.

Value

Log gamma

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
 Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz
 M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```

gamm1n(5)
lgamma(5)           # log(4!) = log(24)

```

gammp	<i>Regularized lower incomplete gamma function $P(a, x)$</i>
-------	---

Description

Incomplete gamma function with parameter a

Usage

```
gammp(a, x)
```

Arguments

a	shape constant, distribution parameter
x	to which it integrates

Details

This calls gser or gcf. Ref: Cephes igam/igamc, as redistributed in ALGLIB specialfunctions.cs. This function works a value at a time: a and x must each be a single number. Use sapply for a vector.

Value

integrated value

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
 Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz
 M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```

gammp(2, 3)           # P(2, 3)
pgamma(3, 2)

```

gammq*Regularized upper incomplete gamma function $Q(a, x)$*

Description

Incomplete gamma function, $Q(a, x) = 1 - P(a, x)$

Usage

`gammq(a, x)`

Arguments

<code>a</code>	shape constant, distribution parameter
<code>x</code>	to which it integrates

Details

This calls `gser` or `gcf`. Ref: Cephes `igam/igamc`, as redistributed in ALGLIB `specialfunctions.cs`. This function works a value at a time: `a` and `x` must each be a single number. Use `sapply` for a vector.

Value

integrated value

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```
gammq(2, 3)           # Q(2, 3) = 1 - P(2, 3)
pgamma(3, 2, lower.tail=FALSE)
```

GenD

Get Hessian and Gradient

Description

This is the combination of Grad and Hessian

Usage

```
GenD(func, x)
```

Arguments

func	vector-input (multi-variable) scalar-valued function
x	x point at which gradients are calculated

Details

Input is the same to Grad and Hessian functions.

Value

gr	gradient vector
Hessian	Hessian matrix

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
GenD(function(x) sum(x^2), c(1, 2))    # gradient, hessian and more
GenD(function(x) sum(x^3), 2)
```

GetP

Get the Direction Vector

Description

Get the direction vector for the new line search during the minimization.

Usage

```
GetP(Hv, g)
```

Arguments

Hv	vector of lower triangular part of hessian
g	gradient

Details

The result is the same as `solve(H, g)`.

Value

Direction vector for a new search

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
# Hv is the lower triangle of the hessian, column by column
GetP(c(4, 1, 3), c(1, 2))
solve(matrix(c(4, 1, 1, 3), 2, 2), c(1, 2))
```

GQuad8

Integration using Gaussian quadrature

Description

Integrates with given fx, a, and b

Usage

```
GQuad8(fx, a, b)
```

Arguments

fx	real valued scalar function
a	from
b	to

Details

This gives approximate integration value using Gaussian quadrature. It uses 8 points approximation.

Value

Integration value from a to b with function.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx = function(x) exp(-x/100)
# exact value: 100 * (1 - exp(-0.24)) = 21.33721
GQuad8(fx, 0, 24)
```

Grad

Gradient Calculation

Description

Get gradient using Richardson extrapolation 4th order.

Usage

```
Grad(func, x)
```

Arguments

func	vector-input (multi-variable) scalar-valued function
x	x point at which gradients are calculated

Details

This is a simplified implementation of 4th order Richardson extrapolation for gradient calculation in numDeriv package.

Value

gradient value

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx2 = function(x) sum(x*x)
Grad(fx2, c(1, 1)) # 2, 2
```

Hessian

*Hessian Calculation***Description**

Get hessian using Richardson extrapolation 4th order.

Usage

```
Hessian(fx, x)
```

Arguments

fx	vector-input (multi-variable) scalar-valued function
x	x point at which hessian is calculated

Details

This is a simplified implementation of 4th order Richardson extrapolation for hessian calculation. See numDeriv package for more detail.

Value

hessian value

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx2 = function(x) sum(x*x)
Hessian(fx2, c(1, 1)) # matrix(c(2, 0, 0, 2), nrow=2)
```

invbetai

*Inverse of incomplete beta function***Description**

Inverse of incomplete beta function with parameter a and b

Usage

```
invbetai(p, a, b)
```

Arguments

p	integration value between 0 and 1
a	shape constant, distribution parameter
b	shape constant, distribution parameter

Details

This calls `gammln` and `betai`. Ref: bracketed Newton steps on `betai`, following `gsl_cdf_beta_Pinv` of the GNU Scientific Library. This function works a value at a time: p, a and b must each be a single number. Use `sapply` for a vector.

Value

inverse of incomplete beta function

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
 Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz
 M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```
invbetai(0.5, 2, 3)
qbeta(0.5, 2, 3)
```

inverfc

Inverse of complementary error function

Description

Inverse of complementary error function.

Usage

```
inverfc(p)
```

Arguments

p	a positive real integration value usually between 0 and 2
---	---

Details

It goes through the normal quantile, since $\text{erfc}(x) = 2 \Phi(-x \sqrt{2})$.

Ref: Cephes ndtri, as redistributed in ALGLIB specialfunctions.cs (invnormalcdf). This function works a value at a time: p must be a single number. Use sapply for a vector.

Value

inverse of complementary error function value, $p = \text{erfc}(x)$

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
 Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz
 M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```
inverfc(0.05)
erfc(inverfc(0.05))      # back to 0.05
inverfc(1)               # 0
```

invgammp	<i>Inverse of gammp function</i>
----------	----------------------------------

Description

Inverse of gammp function

Usage

```
invgammp(p, a)
```

Arguments

p	integration value between 0 and 1
a	shape constant, distribution parameter

Details

This calls gammln, gammp, gammq and inverfc. Ref: Wilson and Hilferty (1931) starting value refined by bracketed Newton steps, following gsl_cdf_gamma_Pinv of the GNU Scientific Library. This function works a value at a time: p and a must each be a single number. Use sapply for a vector.

Value

inverse of incomplete gamma function

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Galassi M, et al. GNU Scientific Library Reference Manual. 3rd ed. Network Theory, 2009.
 Moshier SL. Methods and Programs for Mathematical Functions. Ellis Horwood, 1989. Abramowitz
 M, Stegun IA. Handbook of Mathematical Functions. Dover, 1972.

Examples

```
invgammp(0.5, 3)
qgamma(0.5, 3)
```

 LDLT

LDL' Factorization

Description

LDL' factorization of a symmetric matrix

Usage

LDLT(SymMat)

Arguments

SymMat a symmetric matrix of real values

Details

Input should be a symmetric matrix of real values.

Value

L lower triangular matrix
 D the diagonal of D, as a vector

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
M <- matrix(c(4, 2, 2, 3), 2, 2)
r <- LDLT(M)
r
r$L %%% diag(r$D) %%% t(r$L) # back to M
```

 LGAMMA

Natural Log Gamma Function

Description

This log gamma function is for larger real values, beyond the range of GAMMA. This function uses Lanczos' approximation and is fairly short.

Usage

```
LGAMMA(z)
```

Arguments

z real number

Details

If $\text{abs}(z)$ is less than or equal to 171, it uses GAMMA function. Ref: Lanczos $g=7$ coefficients as used by the GNU Scientific Library, `gsl_sf_lngamma`. This function works a value at a time: z must be a single number. Use `sapply` for a vector.

Value

natural log value of Gamma function

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
LGAMMA(0)
LGAMMA(1) # log(0!) = 0
LGAMMA(2) # log(1!) = 0
LGAMMA(171) # log(170!) = 706.5731
LGAMMA(-171) # Inf for all negative integers, as in base lgamma()
format(LGAMMA(0.5), digits=22) # = log(sqrt(pi)) = 0.5723649429247
format(lgamma(0.5), digits=22)
format(log(sqrt(pi)), digits=22)
format(LGAMMA(1.5), digits=22)
format(lgamma(1.5), digits=22)
```

LOG	<i>Logarithm natural or common</i>
-----	------------------------------------

Description

Natural or common log of a real positive value.

Usage

LOG(x, DecFlag=FALSE)

Arguments

x	a real positive number to be log-ed
DecFlag	FALSE for natural log, TRUE for common log

Details

Current CPUs support this function at the instruction level, so this implementation is for study rather than for speed. Ref. Plauger PJ The Standard C Library. 1992. This function works a value at a time: x and DecFlag must each be a single value. Use sapply for a vector.

Value

log value of base e or 10.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
LOG(10)
log(10)
LOG(10, DecFlag=TRUE)      # base 10
```

MachEps	<i>Get Machine Epsilon</i>
---------	----------------------------

Description

Calculate machine epsilon.

Usage

MachEps()

Arguments

none

Details

Machine epsilon + 1 is different from 1. But half of it added to one cannot be differentiated from one.

Value

Machine epsilon

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

MachEps()

MachEps2	<i>Get Epsilon a little more general than Machine Epsilon</i>
----------	---

Description

Calculate epsilon using x and Start value.

Usage

MachEps2(x = 1, Start = 1)

Arguments

x	value to be used at the equation of $x + \text{eps} > x$
Start	Starting epsilon value. Epsilon is halved by each iteration.

Details

This finds the smallest epsilon which satisfies $x + \text{eps} > x$ by halving epsilon for each iteration. This function works a value at a time: `x` and `Start` must each be a single number. Use `sapply` for a vector.

Value

epsilon

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
MachEps2(100, 1)
(eps2 = MachEps2(1, 0.8))
1 + eps2 == 1 # FALSE and less than .Machine$double.eps
```

MachEps3

Get Meticulous Positive Epsilon

Description

Calculate meticulous positive epsilon using bisectional search.

Usage

```
MachEps3(MaxIter = 1000)
```

Arguments

`MaxIter` maximum iteration number

Details

This finds really the smallest epsilon by bisectional search. This function works a value at a time: `MaxIter` must be a single number. Use `sapply` for a vector.

Value

epsilon with "iteration" count attribute

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
MachEps3() # Smallest epsilon in this environment. Smaller than .Machine$double.eps
```

MachEps4*Get Meticulous Negative Epsilon*

Description

Calculate meticulous negative epsilon using bisectional search.

Usage

```
MachEps4(MaxIter = 1000)
```

Arguments

MaxIter maximum iteration number

Details

This finds really the smallest epsilon by bisectional search. This function works a value at a time: MaxIter must be a single number. Use supply for a vector.

Value

epsilon with "iteration" count attribute

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
MachEps4() # different not just sign, but also magnitude with MachEps3()
```

mlr*Multiple Linear Regression*

Description

Multiple linear regression.

Usage

```
mlr(y, x.raw, standardize=0, Plot=FALSE)
```

Arguments

y	dependent part, a vector
x.raw	independent part, a matrix
standardize	standardize method, 0 means no standardize
Plot	plot or not

Details

It shows various outputs. standardize and Plot must each be a single value, not a vector.

Value

Variable	variable names
Estimate	estimates
SE	standard error
T	t value
p-value	p-value
Residual	residuals
R-Student	externally studentized residuals
hat	hat value
Cook's D	Cook's D
COV-Ratio	covariance ratio
DFFITS	DFFITS
DFBETAs	DFBETAs

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
set.seed(1)
n <- 30
X <- data.frame(a = rnorm(n, 10, 2), b = rnorm(n, -5, 3))
y <- 1 + 2*X$a - X$b + rnorm(n)
fit <- mlr(y, X)
fit[[1]] # estimates, SE, t, p
coef(summary(lm(y ~ a + b, data = X))) # base R, for comparison
mlr(y, X, standardize = 1)[[1]] # centred by column
```

NashSVD*Singular Value Decomposition by Nash*

Description

Singular value decomposition by Nash

Usage

NashSVD(M)

Arguments

M a matrix of real values

Details

This can be used for a least square problem.

Value

Decomposed result

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
M <- matrix(c(1, 2, 3, 4, 5, 6), 3, 2)
r <- NashSVD(M)
r$d                                      # singular values
svd(M)$d
```

NORM*Normalize*

Description

From a decimal number to a normalized number

Usage

NORM(x)

Arguments

x a decimal number to be normalized

Details

It returns (f,n) for $x = f * 2^n$ where ($0.5 \leq f \leq 1$). Returns a vector of length 2. For a negative x, f is negative and its magnitude is as for -x. x must be finite. Ref. Plauger PJ The Standard C Library. 1992. This function works a value at a time: x must be a single number. Use sapply for a vector.

Value

a vector c(f, n) with $x = f * 2^n$

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
NORM(10)                      # 10 = 0.625 * 2^4
DENORM(NORM(10))
```

Optim0

Prototype of optimization function

Description

Shows how a quasi-Newton type optimization algorithm works.

Usage

```
Optim0(x0, func, TypF=1, GradTol=.Machine$double.eps^(1/3),
       StepTol=.Machine$double.eps^(2/3),
       FnTol=.Machine$double.eps^(1/3), ItnLimit=100)
```

Arguments

x0	initial value
func	function to be minimized
TypF	typical function value, a scalar, to measure approximate magnitude
GradTol	gradient tolerance
StepTol	x direction or step tolerance
FnTol	y or function tolerance
ItnLimit	iteration count limit

Details

If the changes of between steps are less than the tolerances, it stops. GradTol, StepTol, FnTol and ItnLimit must each be a single number, not a vector.

Value

par	final x vector values
value	function value at the final x vector
FnCount	function evaluation count
convergence	convergence is met or not
grad	gradient at the final x vector
hessian	hessian at the final x vector
RelGrad	relative gradient
RelStep	relative step
DeIF	delta F, function value
f	function values
x	x vector values

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Optim0(c(-1.2, 1), function(x) 100*(x[2] - x[1]^2)^2 + (1 - x[1])^2)
```

OuterProd

Outer product of two three-dimensional vectors

Description

Calculate the outer product of two three-dimensional vectors

Usage

```
OuterProd(a, b)
```

Arguments

a	the first vector
b	the second vector

Details

Outer product is not commutative.

Value

Outer product of two three-dimensional vectors

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
# the vector cross product of two vectors in three dimensions
OuterProd(c(1, 0, 0), c(0, 1, 0))
OuterProd(c(1, 2, 3), c(4, 5, 6))
```

Pbeta

Cumulative probability of beta distribution

Description

Calculate the cumulative probability for x of a beta distribution

Usage

```
Pbeta(x, alph, bet)
```

Arguments

x	x value for a beta distribution
alph	the shape parameter alpha of a beta distribution
bet	the shape parameter beta of a beta distribution

Details

It calculates the cumulative probability for x of a beta distribution. This function works a value at a time: x, alph and bet must each be a single number. Use sapply for a vector.

Value

cumulative probability till x

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Pbeta(0.3, 2, 3)
pbeta(0.3, 2, 3)
```

Pbinom	<i>Cumulative probability of binomial distribution</i>
--------	--

Description

Calculate the cumulative probability $P(X \leq k)$ of a binomial distribution

Usage

```
Pbinom(k, n, pe)
```

Arguments

k	count k of wanted event for a binomial distribution
n	total number of trials n of a binomial distribution
pe	the probability of wanted event for a binomial distribution

Details

It calculates the cumulative probability for k of a binomial distribution. This function works a value at a time: k, n and pe must each be a single number. Use `sapply` for a vector.

Value

cumulative probability till k

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Pbinom(3, 20, 0.3)      # P(X <= 3)
pbinom(3, 20, 0.3)
```

Pchisq	<i>Cumulative probability of chi-square distribution</i>
--------	--

Description

Calculate the cumulative probability for x2 of a chi-square distribution

Usage

```
Pchisq(x2, nu)
```

Arguments

x2	chi-square value for a chi-square distribution
nu	degrees of freedom nu of a chi-square distribution

Details

It calculates the cumulative probability for x2 of a chi-square distribution. This function works a value at a time: x2 and nu must each be a single number. Use `sapply` for a vector.

Value

cumulative probability till x2

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Pchisq(3.84, 1)          # about 0.95
pchisq(3.84, 1)
```

Pf	<i>Cumulative probability of F distribution</i>
----	---

Description

Calculate the cumulative probability for f of an F distribution

Usage

```
Pf(f, nu1, nu2)
```

Arguments

f	f value for an F distribution
nu1	numerator degrees of freedom nu1 of an F distribution
nu2	denominator degrees of freedom nu2 of an F distribution

Details

It calculates the cumulative probability for f of an F distribution. This function works a value at a time: f, nu1 and nu2 must each be a single number. Use `sapply` for a vector.

Value

cumulative probability till f

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Pf(2, 4, 9)
pf(2, 4, 9)
```

Pgamma

Cumulative probability of gamma distribution

Description

Calculate the cumulative probability for x of a gamma distribution

Usage

```
Pgamma(x, alph, bet=1)
```

Arguments

x	x value for a gamma distribution
alph	the shape parameter alpha of a gamma distribution
bet	the rate parameter beta of a gamma distribution

Details

It calculates the cumulative probability for x of a gamma distribution. This function works a value at a time: x, alph and bet must each be a single number. Use sapply for a vector.

Value

cumulative probability till x

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Pgamma(2, 3)
Pgamma(2, 3, 2)      # bet is a rate
pgamma(2, 3, rate=2)
```

Plnorm

Cumulative probability of log-normal distribution

Description

Calculate the cumulative probability for x of a log-normal distribution

Usage

```
Plnorm(x, mu=0, sig=1)
```

Arguments

x	x value for a log-normal distribution
mu	the mean mu of the logarithm for a log-normal distribution
sig	the standard deviation sigma of the logarithm for a log-normal distribution

Details

It calculates the cumulative probability for x of a log-normal distribution. This function works a value at a time: x, mu and sig must each be a single number. Use `sapply` for a vector.

Value

cumulative probability till x

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Plnorm(1)
plnorm(1)
```

Pnorm*Cumulative probability of normal distribution*

Description

Calculate the cumulative probability for x of a normal distribution

Usage

```
Pnorm(x, mu=0, sig=1)
```

Arguments

x	x value for a normal distribution
mu	the mean mu of a normal distribution
sig	the standard deviation sigma of a normal distribution

Details

It calculates the cumulative probability for x of a normal distribution. This function works a value at a time: x, mu and sig must each be a single number. Use `sapply` for a vector.

Value

cumulative probability till x

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Pnorm(1.96)
pnorm(1.96)
Pnorm(100, mu=110, sig=15) # IQ below 100
```

PolyNom3

Approximation of pnorm using a polynomial equation

Description

It approximates pnorm using a polynomial equation.

Usage

```
PolyNom3(x)
```

Arguments

x	Z value
---	---------

Details

It calculates the approximation of `pnorm(x)`. This function works a value at a time: x must be a single number. Use `sapply` for a vector.

Value

Probability of $Z < x$

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
format(pnorm(2), digits=22)
format(PolyNom3(2), digits=22) # Compare with the above
```

Ppois

Cumulative probability of Poisson distribution

Description

Calculate the cumulative probability $P(X \leq n)$ of a Poisson distribution

Usage

```
Ppois(n, lam)
```

Arguments

n	count n for a Poisson distribution
lam	mean lambda of a Poisson distribution

Details

It calculates the cumulative probability for n of a Poisson distribution. This function works a value at a time: n and lam must each be a single number. Use sapply for a vector.

Value

cumulative probability till n

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Ppois(0, 4)          # P(X <= 0) = exp(-4)
Ppois(3, 4)
ppois(3, 4)
```

Pt	<i>Cumulative probability of t distribution</i>
----	---

Description

Calculate the cumulative probability for t of a t distribution

Usage

```
Pt(t, nu, mu=0, sig=1)
```

Arguments

t	t value for a t distribution
nu	degrees of freedom
mu	the location parameter mu of a t distribution
sig	the scale parameter sigma of a t distribution

Details

It calculates the cumulative probability for t of a t distribution. This function works a value at a time: t, nu, mu and sig must each be a single number. Use `sapply` for a vector.

Value

cumulative probability till t

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Pt(1.5, 7)
pt(1.5, 7)
```

Qbeta

Reverse of Pbeta function

Description

Calculate the x value for the cumulative probability p of a beta distribution

Usage

```
Qbeta(p, alph, bet)
```

Arguments

p	cumulative probability p for a beta distribution
alph	the shape parameter alpha of a beta distribution
bet	the shape parameter beta of a beta distribution

Details

It calculates the x value for the cumulative probability p of a beta distribution. This function works a value at a time: p, alph and bet must each be a single number. Use sapply for a vector.

Value

x of Pbeta(x, alph, bet)

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qbeta(0.5, 2, 3)
qbeta(0.5, 2, 3)
```

Qbinom

Reverse of Pbinom function

Description

Calculate the k for the cumulative probability p of a binomial distribution

Usage

```
Qbinom(p, n, pe)
```

Arguments

p	cumulative probability of wanted event for a binomial distribution
n	total number of trials n of a binomial distribution
pe	the probability of wanted event for a binomial distribution

Details

It returns the smallest k with $P_{\text{binom}}(k, n, pe) \geq p$, the quantile convention of `stats::qbinom`.

Ties are the one place this can part from base R. Feeding a cumulative probability straight back in as p asks which side of that value the answer falls on, and `Pbinom` is built from the incomplete beta function rather than the saddle point algorithm base R uses, so the two agree only to about 1e-12 there. A tolerance of that size is applied before the comparison. For any p not sitting exactly on a cumulative probability the two agree exactly. This function works a value at a time: p, n and pe must each be a single number. Use `sapply` for a vector.

Value

the smallest k whose cumulative probability reaches p

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qbinom(0.5, 20, 0.3)      # smallest k with P(X <= k) >= p
qbinom(0.5, 20, 0.3)
```

Qchisq

Reverse of Pchisq function

Description

Calculate the chi-square value for the cumulative probability p of a chi-square distribution

Usage

```
Qchisq(p, nu)
```

Arguments

p	cumulative probability p for a chi-square distribution
nu	degrees of freedom nu of a chi-square distribution

Details

It calculates the chi-square value for the cumulative probability p of a chi-square distribution. This function works a value at a time: p and nu must each be a single number. Use `sapply` for a vector.

Value

x2 of Pchisq(x2, nu)

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qchisq(0.95, 1)      # about 3.84
qchisq(0.95, 1)
```

Qf

Reverse of Pf function

Description

Calculate the f for the cumulative probability p of an F distribution

Usage

```
Qf(p, nu1, nu2)
```

Arguments

p	cumulative probability for f of an F distribution
nu1	numerator degrees of freedom nu1 of an F distribution
nu2	denominator degrees of freedom nu2 of an F distribution

Details

It calculates the f for the cumulative probability p of an F distribution. This function works a value at a time: p, nu1 and nu2 must each be a single number. Use sapply for a vector.

Value

f of Pf(f, nu1, nu2)

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qf(0.95, 4, 9)
qf(0.95, 4, 9)
```

Qgamma

Reverse of Pgamma function

Description

Calculate the x value for the cumulative probability p of a gamma distribution

Usage

```
Qgamma(p, alph, bet=1)
```

Arguments

p	cumulative probability p for a gamma distribution
alph	the shape parameter alpha of a gamma distribution
bet	the rate parameter beta of a gamma distribution

Details

It calculates the x value for the cumulative probability p of a gamma distribution. This function works a value at a time: p, alph and bet must each be a single number. Use sapply for a vector.

Value

x of Pgamma(x, alph, bet)

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qgamma(0.5, 3)
qgamma(0.5, 3)
```

Qlnorm

Reverse of Plnorm function

Description

Calculate the x value for the cumulative probability p of a log-normal distribution

Usage

```
Qlnorm(p, mu=0, sig=1)
```

Arguments

p	cumulative probability p for a log-normal distribution
mu	the mean mu of the logarithm for a log-normal distribution
sig	the standard deviation sigma of the logarithm for a log-normal distribution

Details

It calculates the x value for the cumulative probability p of a log-normal distribution. p and sig must each be a single number, not a vector. Use `sapply` over p for several values.

Value

x of `Plnorm(x, mu, sig)`

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qlnorm(0.5)
qlnorm(0.5)
```

Qnorm

Reverse of Pnorm function

Description

Calculate the x value for the cumulative probability p of a normal distribution

Usage

```
Qnorm(p, mu=0, sig=1)
```

Arguments

p	cumulative probability p for a normal distribution
mu	the mean mu of a normal distribution
sig	the standard deviation sigma of a normal distribution

Details

It calculates the x value for the cumulative probability p of a normal distribution. p and sig must each be a single number, not a vector. Use `sapply` over p for several values.

Value

x of Pnorm(x, mu, sig)

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qnorm(0.975)
qnorm(0.975)
Qnorm(0.5, mu=110, sig=15)
```

Qpois	<i>Reverse of Ppois function</i>
-------	----------------------------------

Description

Calculate the n for the cumulative probability p of a Poisson distribution

Usage

```
Qpois(p, lam)
```

Arguments

p	cumulative probability of a Poisson distribution
lam	mean lambda of a Poisson distribution

Details

It returns the smallest n with $Ppois(n, lam) \geq p$, the quantile convention of `stats::qpois`.

Ties are the one place this can part from base R. Feeding a cumulative probability straight back in as p asks which side of that value the answer falls on, and Ppois is built from the incomplete gamma function rather than the saddle point algorithm base R uses, so the two agree only to about 1e-12 there. A tolerance of that size is applied before the comparison. For any p not sitting exactly on a cumulative probability the two agree exactly. This function works a value at a time: p and lam must each be a single number. Use `sapply` for a vector.

Value

the smallest n whose cumulative probability reaches p

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qpois(0.5, 4)          # smallest n with P(X <= n) >= p
qpois(0.5, 4)
```

Qt

*Reverse of Pt function***Description**

Calculate the t value for the cumulative probability p of a t distribution

Usage

```
Qt(p, nu, mu=0, sig=1)
```

Arguments

p	cumulative probability p for a t distribution
nu	degrees of freedom
mu	the location parameter mu of a t distribution
sig	the scale parameter sigma of a t distribution

Details

It calculates the t value for the cumulative probability p of a t distribution. p and nu must each be a single number, not a vector. Use `sapply` over p for several values.

Value

t of `Pt(t, nu, mu, sig)`

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Qt(0.975, 7)
qt(0.975, 7)
```

Rbeta	<i>Random deviate from beta distribution</i>
-------	--

Description

Generate random numbers from a beta distribution

Usage

```
Rbeta(n, alph, bet)
```

Arguments

n	count, how many do you want
alph	shape parameter alpha of a beta distribution
bet	shape parameter beta of a beta distribution

Details

It generates random numbers from random numbers of other distributions. alph and bet must each be a single number, not a vector.

Value

a vector of random numbers

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
set.seed(1)
Rbeta(5, 2, 5)
round(mean(Rbeta(10000, 2, 5)), 3)      # near 2/(2+5)
```

Rexp

Random deviate from exponential distribution

Description

Generate random numbers from an exponential distribution

Usage

```
Rexp(n, alpha)
```

Arguments

n	count, how many do you want
alpha	rate parameter alpha of an exponential distribution

Details

It uses inverse-transformation method.

Value

a vector of random numbers

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
set.seed(1)
Rexp(5, 2)
round(mean(Rexp(10000, 2)), 3)           # alpha is a rate, near 0.5
```

Rgamma*Random deviate from gamma distribution*

Description

Generate random numbers from a gamma distribution

Usage

```
Rgamma(n, alph, bet)
```

Arguments

n	count, how many do you want
alph	shape parameter alpha of a gamma distribution
bet	scale parameter beta of a gamma distribution

Details

It uses acceptance-rejection method. alph must be a single number, not a vector.

Value

a vector of random numbers

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Banks J. Handbook of simulation (1998) pp152-153

Examples

```
set.seed(1)
Rgamma(5, 3, 2)
# bet is a scale here, so the mean is alph * bet
round(mean(Rgamma(10000, 3, 2)), 2)      # near 6
```

Rgamma0

Random deviate from gamma distribution, simple method

Description

Generate random numbers from a gamma distribution by a simple method

Usage

```
Rgamma0(n, alph, bet)
```

Arguments

n	count, how many do you want
alph	shape parameter alpha of a gamma distribution, at least 1
bet	scale parameter beta of a gamma distribution

Details

It uses acceptance-rejection method. alph must be a single number, not a vector.

Value

a vector of random numbers

Author(s)

Kyun-Seop Bae <k@acr.kr>

References

Banks J. Handbook of simulation (1998) pp152-153

Examples

```
set.seed(1)
Rgamma0(5, 3, 2)
round(mean(Rgamma0(10000, 3, 2)), 2)      # near 6
# alph must be at least 1; use Rgamma for smaller shapes
```

 RK4

Runge-Kutta 4th Order Routine for Numerical Integration

Description

Advance State vector by tau time with Deriv

Usage

```
RK4(State, tau, Deriv)
```

Arguments

State	state vector
tau	time amount to advance
Deriv	Derivative functions with the input of State vector

Details

Deriv should accept State vector as an input.

Value

A new state vector after time advancement by tau

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
# harmonic oscillator: x'' = -x, state is c(x, x')
Deriv <- function(s) c(s[2], -s[1])
RK4(c(1, 0), 0.1, Deriv)
c(cos(0.1), -sin(0.1))          # exact
```

Rmvn

*Random deviate from multivariate normal distribution***Description**

Generate random numbers from a multivariate normal distribution

Usage

```
Rmvn(n, Mu, Cov)
```

Arguments

n	count, how many do you want
Mu	mean vector of multivariate normal distribution
Cov	variance-covariance matrix of multivariate normal distribution

Details

It uses Cholesky decomposition. n must be a single number, not a vector.

Value

a matrix of random vectors

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
set.seed(1)
Cov <- matrix(c(1, 0.7, 0.7, 1), 2, 2)
Rmvn(4, c(0, 0), Cov)
round(cor(Rmvn(5000, c(0, 0), Cov))[1, 2], 2) # near 0.7
```

Rnorm	<i>Random deviate from normal distribution</i>
-------	--

Description

Generate random numbers from a normal distribution

Usage

```
Rnorm(n, mu=0, sigma=1)
```

Arguments

n	count, how many do you want
mu	mean of normal distribution
sigma	standard deviation of normal distribution

Details

It is more efficient with even n. n must be a single number, not a vector.

Value

a vector of random numbers

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
set.seed(1)
Rnorm(5)
set.seed(1)
round(mean(Rnorm(10000, mu=1, sigma=2)), 2) # near 1
```

romb*Integration using Romberg integration*

Description

Integrates with given fx, a, b, and N

Usage

```
romb(fx, a, b, N=4)
```

Arguments

fx	real valued scalar function
a	from
b	to
N	degree; the larger, the more precise

Details

This gives approximate integration value using Romberg integration. N is the number of rows of the Romberg table; N = 1 is the trapezoidal rule over the whole interval. N must be a single number, not a vector.

Value

Integration value from a to b with function.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx = function(x) exp(-x/100)
# exact value: 100 * (1 - exp(-0.24)) = 21.33721
romb(fx, 0, 24, 10)
```

Round	<i>Ordinary Rounding Function</i>
-------	-----------------------------------

Description

Rounding function in R is round and its behaviour is different from other common software. This is an ordinary rounding function.

Usage

```
Round(x, digits=0)
```

Arguments

x	real number to be rounded
digits	integer indicating the number of decimal places. 0 means $10^0=1$, 2 means $10^{-2}=0.01$, -1 means $10^1 = 10$ etc.

Details

Ordinary rounding function (round half away from zero). For more details, see Wikipedia.

Value

rounded value

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
Round(1:10 - 0.5)
Round(450, -2)
Round(1.235, 2)
```

run.p	<i>Cumulative probability of run count for run test</i>
-------	---

Description

Cumulative probability of the run count for the run test

Usage

```
run.p(m, n, r)
```

Arguments

m	count of fewer species (minimum value = 0)
n	count of more frequent species (minimum value = 1)
r	count of run (minimum value = 1)

Details

It calculates $P(\text{Run count} \leq r \mid m, n)$, the cumulative probability of the run count that `Run.test` turns into a p-value. This function works a value at a time: `m`, `n` and `r` must each be a single number. Use `sapply` for a vector.

Value

probability of run count to be less than or equal to `r` with `m` and `n`. $P(\text{Run count} \leq r \mid m, n)$

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
run.p(5, 5, 4)           # P(run count <= 4 | m=5, n=5)
run.p(5, 5, 10)
```

Run.test	<i>Run test on a sequence</i>
----------	-------------------------------

Description

p-value for run test

Usage

```
Run.test(RES)
```

Arguments

RES sequence vector to be tested, usually residual values

Details

It calculates p-value for run test. Zeros are omitted first. If no nonzero value is left, there are no runs to count and it returns NA.

Value

p-value, if it is larger than 0.5, it returns 1 - p.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
set.seed(1)
Run.test(rnorm(30))           # random, so a large p-value
Run.test(rep(c(1, -1), 15))  # perfectly alternating
```

runifLC	<i>Random number of uniform distribution [0, 1) by linear congruential method</i>
---------	---

Description

Random number generation distributed as Uniform[0, 1) by linear congruential method

Usage

```
runifLC(n = 1, Seed = 123457, a = 16807, m = 2147483647, k = 0, b = 10)
```

Arguments

n	count of random numbers to generate
Seed	Seed number. If abs(Seed) is less than 1, Sys.time() is the seed.
a	multiplier
m	modulus. It should be a prime number.
k	increment
b	count of initial values to discard

Details

$R[i] = X[i]/(m + 1)$. $X[i] = (a * X[i - 1] + k) \bmod m$. Then, period is $m - 1$. n, Seed, a and b must each be a single number, not a vector. Use sapply over n for several values.

Value

Uniform[0, 1) random numbers of n

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
runifLC(8)
runifLC(8, 1)
runifLC(8, 2)
runifLC(8, 0) # random seed
```

sft

Slow Fourier Transformation

Description

Fourier transformation using the definition, therefore slow.

Usage

```
sft(y)
```

Arguments

y	a sequence of fixed time interval
---	-----------------------------------

Details

This is slow but easy to understand.

Value

Fourier transformation result

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
sft(c(1, 2, 3, 4))      # slow Fourier transform
fft(c(1, 2, 3, 4))      # base R, for comparison
```

simps13

Integration using Simpson's 1/3 formula

Description

Integrates with given fx, a, b, and n

Usage

```
simps13(fx, a, b, n)
```

Arguments

fx	real valued scalar function
a	from
b	to
n	split count

Details

This gives approximate integration value using Simpson's 1/3 formula. Argument n should be an even number. n must be a single number, not a vector.

Value

Integration value from a to b with function.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx = function(x) exp(-x/100)
# exact value: 100 * (1 - exp(-0.24)) = 21.33721
simps13(fx, 0, 24, 10)  # n must be even
simps13(fx, 0, 24, 30)
```

simps38*Integration using Simpson's 3/8 formula*

Description

Integrates with given fx, a, b, and n

Usage

```
simps38(fx, a, b, n)
```

Arguments

fx	real valued scalar function
a	from
b	to
n	split count

Details

This gives approximate integration value using Simpson's 3/8 formula. Argument n should be a multiple of 3. n must be a single number, not a vector.

Value

Integration value from a to b with function.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx = function(x) exp(-x/100)
# exact value: 100 * (1 - exp(-0.24)) = 21.33721
simps38(fx, 0, 24, 12)  # n must be a multiple of 3
simps38(fx, 0, 24, 30)
```

SQRT*Square-rooting*

Description

Square-rooting of a real positive number.

Usage

SQRT(x)

Arguments

x a real positive number to be square-rooted

Details

Current CPUs support this function at the instruction level, so this implementation is for study rather than for speed. Ref. Plauger PJ The Standard C Library. 1992. This function works a value at a time: x must be a single number. Use `sapply` for a vector.

Value

square-rooted value

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
SQRT(2)
sqrt(2)
```

SymDet*Determinant of a Symmetric Matrix*

Description

Determinant of a symmetric matrix

Usage

SymDet(SymMat)

Arguments

SymMat a symmetric matrix of real values

Details

Input should be a symmetric matrix of real values.

Value

Determinant of the input

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
M <- matrix(c(4, 1, 1, 3), 2, 2)
SymDet(M)
det(M)
```

SymInv

Inverse of a Symmetric Matrix

Description

Inverse of a symmetric matrix

Usage

SymInv(SymMat)

Arguments

SymMat a symmetric matrix of real values

Details

Input should be a symmetric matrix of real values.

Value

Inverse of the input

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
M <- matrix(c(4, 1, 1, 3), 2, 2)
SymInv(M)
solve(M)
```

SymSol

*Solution of a Symmetric Matrix***Description**

Solution of a symmetric matrix linear equation

Usage

```
SymSol(H, g)
```

Arguments

H	a symmetric matrix of real values, such as hessian
g	a response vector, such as gradient

Details

Input should be a symmetric matrix of real values and a real vector

Value

Solution of the linear system with a real symmetric coefficient matrix

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
M <- matrix(c(4, 1, 1, 3), 2, 2)
SymSol(M, c(1, 2))
solve(M, c(1, 2))
```

tableFactorial	<i>Factorial using Table</i>
----------------	------------------------------

Description

This function returns factorial value from the stored table.

Usage

```
tableFactorial(n)
```

Arguments

n non-negative integer to be calculated

Details

This function returns factorial value from the table known to be precise. This function works a value at a time: n must be a single number. Use sapply for a vector.

Value

factorial value

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
tableFactorial(0)
tableFactorial(170)
tableFactorial(171)
```

trapez0	<i>Integration using trapezoidal rule on two sequences</i>
---------	--

Description

Integrates with given two sequences: x_i and y_i

Usage

```
trapez0(x, y)
```

Arguments

x	sequence of x
y	sequence of y

Details

This is the simplest implementation of the trapezoidal rule. The lengths of the two sequences should be the same.

Value

Integration value with given sequences.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx = function(x) exp(-x/100)
x = c(0, 0.5, 1, 2, 3, 4, 6, 8, 12, 24)
y = fx(x)
trapez0(x,y)
```

trapez1

Integration using trapezoidal rule on a function

Description

Integrates with given fx, a, b, and n

Usage

```
trapez1(fx, a, b, n)
```

Arguments

fx	real valued scalar function
a	from
b	to
n	split count

Details

This gives approximate integration value using trapezoidal rule. n must be a single number, not a vector.

Value

Integration value from a to b with function.

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
fx = function(x) exp(-x/100)
# exact value: 100 * (1 - exp(-0.24)) = 21.33721
trapez1(fx, 0, 24, 10)
trapez1(fx, 0, 24, 1000)
```

VMmin	<i>Simplified version of optimization function</i>
-------	--

Description

Shows how a quasi-Newton type optimization algorithm works.

Usage

```
VMmin(x0, func, MaxIter=9999, Tol=1e-4)
```

Arguments

x0	initial value
func	function to be minimized
MaxIter	iteration count limit
Tol	tolerance

Details

If the changes of between steps are less than the tolerances, it stops. MaxIter and Tol must each be a single number, not a vector.

Value

par	final x vector values
value	function value at the final x vector
FnCount	function evaluation count
GrCount	gradient evaluation count
convergence	convergence is met or not
grad	gradient at the final x vector

Author(s)

Kyun-Seop Bae <k@acr.kr>

Examples

```
VMmin(c(-1.2, 1), function(x) 100*(x[2] - x[1]^2)^2 + (1 - x[1])^2)
VMmin(0, function(x) sum((x - 2.5)^2) + 1) # one dimension
```

Index

betai, [4](#)
Bin2Dec, [5](#)

ChkFx, [6](#)
Chol, [7](#)
Choose, [8](#)
conv, [9](#)
conv0, [9](#)

Dbeta, [10](#)
Dbinom, [11](#)
dc, [12](#)
Dchisq, [13](#)
Dec2Bin, [13](#)
deconv, [14](#)
deconv0, [15](#)
DENORM, [16](#)
Deriv0, [17](#)
Deriv1, [17](#)
Deriv2, [18](#)
Df, [19](#)
Dgamma, [20](#)
Dlnorm, [21](#)
Dnorm, [21](#)
Dpois, [22](#)
Dt, [23](#)

ellipRange, [24](#)
ellipse, [25](#)
erf, [26](#)
erfc, [27](#)
erfccheb, [28](#)
evJacobi, [29](#)
EXP, [29](#)

GAMMA, [30](#)
gammln, [31](#)
gamp, [32](#)
gammq, [33](#)
GenD, [34](#)

GetP, [34](#)
GQuad8, [35](#)
Grad, [36](#)

Hessian, [37](#)

invbetai, [37](#)
inverfc, [38](#)
invgammp, [39](#)

LDLT, [40](#)
LGAMMA, [41](#)
LOG, [42](#)

MachEps, [43](#)
MachEps2, [43](#)
MachEps3, [44](#)
MachEps4, [45](#)
mlr, [45](#)

NashSVD, [47](#)
NORM, [47](#)

Optim0, [48](#)
OuterProd, [49](#)

Pbeta, [50](#)
Pbinom, [51](#)
Pchisq, [51](#)
Pf, [52](#)
Pgamma, [53](#)
Plnorm, [54](#)
Pnorm, [54](#)
PolyNom3, [55](#)
Ppois, [56](#)
Pt, [57](#)

Qbeta, [58](#)
Qbinom, [58](#)
Qchisq, [59](#)
Qf, [60](#)

Qgamma, [61](#)
Qlnorm, [61](#)
Qnorm, [62](#)
Qpois, [63](#)
Qt, [64](#)

Rbeta, [65](#)
Rexp, [66](#)
Rgamma, [66](#)
Rgamma0, [67](#)
RK4, [68](#)
Rmvn, [69](#)
Rnorm, [70](#)
romb, [71](#)
Round, [72](#)
run.p, [73](#)
Run.test, [74](#)
runifLC, [74](#)

sft, [75](#)
simps13, [76](#)
simps38, [77](#)
SQRT, [78](#)
SymDet, [78](#)
SymInv, [79](#)
SymSol, [80](#)

tableFactorial, [81](#)
trapez0, [81](#)
trapez1, [82](#)

VMmin, [83](#)