

# Package ‘NMcalc’

January 8, 2024

**Title** Basic Calculations for PK/PD Modeling

**Version** 0.0.3

**Maintainer** Philip Delff <philip@delff.dk>

**Description** Essentials for PK/PD (pharmacokinetics/pharmacodynamics) such as area under the curve, (geometric) coefficient of variation, and other calculations that are not part of base R. This is not a noncompartmental analysis (NCA) package.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**Imports** stats, data.table

**Suggests** ggplot2

**NeedsCompilation** no

**Author** Philip Delff [aut, cre]

**Repository** CRAN

**Date/Publication** 2024-01-08 21:20:06 UTC

## R topics documented:

CV . . . . .	2
invlogit . . . . .	2
logit . . . . .	3
means . . . . .	3
quantbin . . . . .	4
seqlog . . . . .	6
signif2 . . . . .	6
trapez . . . . .	7
<b>Index</b>	<b>8</b>

CV *Calculate coefficient of variation of data*

---

**Description**

Calculate coefficient of variation of data

**Usage**

```
CV(x, log = FALSE)
```

**Arguments**

x	The data
log	If TRUE, the geometric coefficient of variation is calculated. This is $\sqrt{\text{exp}(\text{var}(\log(x)) - 1)}$ .

**Details**

This function is intended to be used on data. For a log-normal  $\text{THETA1} * \text{EXP}(\text{ETA}(1))$  'Nonmem' parameter, do  $\text{CV} = \sqrt{\text{exp}(\text{OMEGA}[1,1]) - 1}$ .

**Value**

A numeric

**Examples**

```
set.seed(139)
x1 <- rnorm(1000, mean=5)
CV(x1)
CV(x1, log=TRUE)
x2 <- exp(x1)
CV(x2)
CV(x2, log=TRUE)
```

---

invlogit *Inverse logit function*

---

**Description**

Inverse logit function

**Usage**

```
invlogit(x)
```

**Arguments**

x                    a number to transform

**See Also**

logit

---

logit	<i>Logit function</i>
-------	-----------------------

---

**Description**

Logit function

**Usage**

```
logit(x)
```

**Arguments**

x                    a number to transform

**See Also**

invlogit

---

means	<i>calculate arithmetic or geometric mean and confidence intervals</i>
-------	--

---

**Description**

calculate arithmetic or geometric mean and confidence intervals

**Usage**

```
means(  
  x,  
  type = "arithmetic",  
  na.rm = FALSE,  
  z.rm = FALSE,  
  ci = FALSE,  
  dist.ci = "t",  
  p.ci = 0.95,  
  colnames = c("est", "ll", "ul"),  
  format = "df"  
)
```

**Arguments**

x	vector to calculate the geometric mean of
type	type of mean or median. Default is arithmetic, geometric and median are available as well. Only first letters needed, so say "geo" or even "g" is enough.
na.rm	Remove NA's before doing calculations?
z.rm	removes zeros before calculation? Default is FALSE. Can only be TRUE if type="geometric".
ci	if TRUE, a data.frame including point estimate and confidence interval returned. If FALSE, a numeric representing the mean value returned.
dist.ci	The distribution to use for the confidence interval. Default and only supported is "t". If type=geometric, this is applied after transformation to gaussian.
p.ci	probability covered by confidence interval. Default is 0.95
colnames	If ci, this defines the column names of the resulting data frame. Default is c("est","ll","ul").
format	The format of the result. Possible values are df and num.

**Value**

If ci=FALSE, a numeric. If ci=TRUE, a data.frame.

**Examples**

```
x <- 1:100
means(x, type="arithmetic", ci=TRUE)
means(x, type="geometric", ci=TRUE)
means(x, type="median", ci=TRUE)
library(data.table)
## CRAN requires examples to run on a single thread
data.table::setDTthreads(1)
data.table(x=x)[,append(means(x,ci=TRUE),list(N=.N))]
```

---

quantbin

*Bin observations by quantiles. Label by bin number or by interval.*

---

**Description**

This is simple stuff, but I can never remember the exact quantile and findInterval/cut commands to use. quantbin finds quantiles using quantile and then assigns bins using either findInterval or cut.

**Usage**

```
quantbin(x, nbins, label = "num", ...)
```

**Arguments**

x	The observations
nbins	Number of bins to use
label	label="num" gives a numeric bin number (findInterval). label="interval" gives a character representation of the interval (cut).
...	additional arguments passed to quantile.

**Details**

quantbin uses stats::quantile for quantile estimation. Except for x and probs, all parameters can be controlled using na.rm and ... arguments. See ?stats::quantile for details.

na.rm na.rm=TRUE is needed for quantile to be able to estimate the distribution if x contains NA's. Notice, if na.rm=T, an NA element in x will still result in an NA element in return. If na.rm=F and there are NA's in x, all elements will be NA in result (quantiles cannot be determined, nor can the binning of x by those quantiles).

If data is not continuous, this method may not lead to balanced distributions.

**Value**

If label="num", integers. If label="interval", factors.

**Examples**

```
set.seed(134)
library(data.table)
## CRAN requires examples to run on a single thread
data.table::setDTthreads(1)
dt1 <- data.table(x=rnorm(n=1000))
dt1[,bin:=quantbin(x,nbins=4,label="num")]
dt1[,int:=quantbin(x,nbins=4,label="interval")]
## perfect - flat distribution
dt1[,.N,keyby=(bin,int)]

dt2 <- data.table(x=c(rnorm(n=100000),NA))
dt2[,bin:=quantbin(x,nbins=4,label="num",na.rm=TRUE)]
dt2[,int:=quantbin(x,nbins=4,label="interval",na.rm=TRUE)]
## perfect - flat distribution
dt2[,.N,keyby=(bin,int)]
unique(dt2[,.(bin,int)][order(bin)])

## we may not get a flat distribution in case of discrete observations
dt3 <- data.table(x=c(sample(1:3,100,replace=TRUE)))
dt3[,bin:=quantbin(x,nbins=2,label="num",na.rm=TRUE)]
dt3[,int:=quantbin(x,nbins=2,label="interval",na.rm=TRUE)]
## Not a flat distribution
dt3[,.N,keyby=(x,bin,int)]
```

---

seqlog	<i>Log-scale equidistant sequences</i>
--------	--

---

**Description**

Useful for generating sequences to be plotted on log scale. This is really simple - seq is run on from and to after log transformation, then the exponential is reported.

**Usage**

```
seqlog(from, to, length.out)
```

**Arguments**

from	start of sequence
to	end of sequence
length.out	length of sequence

**Value**

A numeric vector.

**Examples**

```
df <- data.frame(x=seqlog(1,100,100))
df <- transform(df, y=x/(10+x))
## Not run:
library(ggplot2)
## the points are equidistant on the log x scale
ggplot(df,aes(x,y))+geom_point()+scale_x_log10()

## End(Not run)
```

---

signif2	<i>round to fixed number of significant digits</i>
---------	--

---

**Description**

Even if theoretically correct, the built-in 'R' functions 'round' and 'signif' can be confusing (see examples). 'signif2' is a simple solution that can be used for reporting results consistently.

**Usage**

```
signif2(x, digits = 1, add, ...)
```

**Arguments**

x	a numeric vector.
digits	number of significant digits to round to. Must be an integer larger than 0.
add	pad with zeros where digits>nchar(x[i]). Currently not used.
...	additional arguments passed to formatC.

**Value**

A character vector.

**Examples**

```
x <- c(1.24e-4,1.1334e6,1.1,22.00000,10.00,1)
data.frame(x,s.3=signif(x,3),sc.3=as.character(signif(x,3)),s2.3=signif2(x,3))
signif2(c(.2,11.84),2)
## digits has no effect when x==0
signif2(0,1)
signif2(0,3)
```

---

trapez

*trapezoidal area under the curve on linear scale*


---

**Description**

This is a numerical integration of y with respect to x by the trapezoidal method on linear scale.

**Usage**

```
trapez(x, y, cum = FALSE, na.rm = FALSE)
```

**Arguments**

x	The vector to integrate y with respect to (typically TIME to get area under the curve).
y	The variable to integrate.
cum	Return the cumulative trapezoidal area under the curve? If false (default) a single number is returned. If true, a vector is returned. Notice, the vector is one element shorter than x and y.
na.rm	Remove indexes in x and y wherever x or y are NA.

**Value**

a numeric

# Index

CV, [2](#)

invlogit, [2](#)

logit, [3](#)

means, [3](#)

quantbin, [4](#)

seqlog, [6](#)

signif2, [6](#)

trapez, [7](#)