

Package: catseyes (via r-universe)

June 5, 2026

Type Package

Title Create Catseye Plots Illustrating the Normal Distribution of the Means

Version 0.2.5

Description Provides the tools to produce catseye plots, principally by `catseyesplot()` function which calls R's standard `plot()` function internally, or alternatively by the `catseyes()` function to overlay the catseye plot onto an existing R plot window. Catseye plots illustrate the normal distribution of the mean (picture a normal bell curve reflected over its base and rotated 90 degrees), with a shaded confidence interval; they are an intuitive way of illustrating and comparing normally distributed estimates, and are arguably a superior alternative to standard confidence intervals, since they show the full distribution rather than fixed quantile bounds. The `catseyesplot` and `catseyes` functions require pre-calculated means and standard errors (or standard deviations), provided as numeric vectors; this allows the flexibility of obtaining this information from a variety of sources, such as direct calculation or prediction from a model. Catseye plots, as illustrations of the normal distribution of the means, are described in Cumming (2013 & 2014). Cumming, G. (2013). The new statistics: Why and how. *Psychological Science*, 27, 7-29. <[doi:10.1177/0956797613504966](https://doi.org/10.1177/0956797613504966)> pmid:24220629.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Suggests emmeans

NeedsCompilation no

Author Clark Andersen [cre, aut]

Maintainer Clark Andersen <crandersen@mdanderson.org>

Repository <https://clarkandersen.r-universe.dev>

Date/Publication 2020-05-08 15:40:02 UTC

RemoteUrl <https://github.com/cran/catseyes>

RemoteRef HEAD

RemoteSha 110383a1f401806224b57ef571a3200beeeb88a1

Contents

catseyes	2
catseyesplot	4

Index	8
--------------	----------

catseyes	<i>catseyes</i>
----------	-----------------

Description

The `catseyes()` function is used to plot catseye interval(s) onto an existing basic R plot background. Catseye plots illustrate the normal distribution of the mean (picture a normal bell curve reflected over its base and rotated 90 degrees), with a shaded confidence interval; they are an intuitive way of illustrating and comparing normally distributed estimates, and are arguably a superior alternative to standard confidence intervals, since they show the full distribution rather than fixed quantile bounds. The `catseyes()` function requires pre-calculated means and standard errors (or standard deviations), provided as numeric vectors; this allows the flexibility of obtaining this information from a variety of sources, such as direct calculation or prediction from a model – see examples below. NOTE: The drawn vertical range of the outline spans 99.8% of the distribution of the mean.

Usage

```
catseyes(
  x,
  ymean,
  yse,
  dx = 0.1,
  conf = 0.95,
  se.only = TRUE,
  col = "black",
  shade = rgb(0.05, 0.05, 0.05, 0.2),
  lwd = 1,
  plot.mean.line = FALSE,
  fTransform = NULL
)
```

Arguments

x	numeric horizontal position(s); if factor, will be converted to integer in factor level order
ymean	numeric mean(s)
yse	numeric standard error(s); may use standard deviation(s) for population level plots
dx	specifies the width (in x direction) of the catseye interval(s)
conf	specifies the confidence of the confidence interval (conf=.95 for alpha=.05)
se.only	boolean, if TRUE (default) will shade only +/- 1 standard error about the mean, overriding conf, otherwise if FALSE will shade the confidence interval (per conf) about the mean
col	specifies the color of the outline of the catseye, as well as the interval point & line, if shown
shade	specifies the color of the shaded confidence region
lwd	sets the line width of the interval and outline
plot.mean.line	boolean, draws a horizontal line at the position of the mean if TRUE
fTransform	Optional function to transform catseye plot from normal distribution (as with analyzing log-transformed data, see example under catseyesplot)

Author(s)

Clark R. Andersen <crandersen@mdanderson.org>

References

Cumming, G. (2014). The new statistics: Why and how. *Psychological Science*, 27, 7-29. <doi:10.1177/0956797613504966>
 pmid:24220629
<http://www.psychologicalscience.org/index.php/publications/observer/2014/march-14/theres-life-beyond-05.html>

Examples

```
#Show catseye plots for 4 groups with means of c(-3,2,-1,6)
# and standard errors of c(1,2,4,3)
plot(NULL,xlim=c(.5,4.5),ylim=c(-10,10),xlab="",ylab="",main="4 Groups",xaxt="n")
axis(1,at=1:4,labels = c("Group1","Group2","Group3","Group4"))
catseyes(1:4,ymean=c(-3,2,-1,6),yse=c(1,2,4,3))
#Optionally, add points and lines (usually lines only when joining time sequence)
lines(1:4,c(-3,2,-1,6),type="b")
```

`catseyesplot`*catseyesplot*

Description

The `catseyesplot()` function plots catseye intervals as a basic R `plot()` window in one step. Can be called with standard plot parameters to further customize the resulting figure. If `xlim` & `ylim` are not specified, these will be generated internally per the provided `x`, `ymean`, and `yse`. Catseye plots illustrate the normal distribution of the mean (picture a normal bell curve reflected over its base and rotated 90 degrees), with a shaded confidence interval; they are an intuitive way of illustrating and comparing normally distributed estimates, and are arguably a superior alternative to standard confidence intervals, since they show the full distribution rather than fixed quantile bounds. The `catseyesplot()` function requires pre-calculated means and standard errors (or standard deviations), provided as numeric vectors; this allows the flexibility of obtaining this information from a variety of sources, such as direct calculation or prediction from a model – see examples below. NOTE: The drawn vertical range of the outline spans 99.8% of the distribution of the mean.

Usage

```
catseyesplot(  
  x,  
  ymean,  
  yse,  
  dx = 0.1,  
  conf = 0.95,  
  se.only = TRUE,  
  col = "black",  
  shade = rgb(0.05, 0.05, 0.05, 0.2),  
  lwd = 1,  
  plot.mean.line = FALSE,  
  fTransform = NULL,  
  labels = FALSE,  
  xlim = NULL,  
  ylim = NULL,  
  x_scatter = NULL,  
  y_scatter = NULL,  
  jitter_scatter = FALSE,  
  dx_scatter = 0.05,  
  pch_scatter = 1,  
  col_scatter = 1,  
  cex_scatter = 1,  
  ...  
)
```

Arguments

`x` numeric horizontal position(s); if factor, will be converted to integer in factor level order

ymean	numeric mean(s)
yse	numeric standard error(s); may use standard deviation(s) for population level plots
dx	specifies the width (in x direction) of the catseye interval(s)
conf	specifies the confidence of the confidence interval (conf=.95 for alpha=.05)
se.only	boolean, if TRUE (default) will shade only +/- 1 standard error about the mean, overriding conf, otherwise if FALSE will shade the confidence interval (per conf) about the mean
col	specifies the color of the outline of the catseye, as well as the interval point & line, if shown
shade	specifies the color of the shaded confidence region
lwd	sets the line width of the interval and outline
plot.mean.line	boolean, draws a horizontal line at the position of the mean if TRUE
fTransform	Optional function to transform catseye plot from normal distribution (as with analyzing log-transformed data, see example)
labels	Optional, may be logical (if TRUE, uses x) or a character vector
xlim	x limits of the plot, as with plot.default
ylim	y limits of the plot, as with plot.default
x_scatter	numeric x values of corresponding raw data for scatterplot; factors will convert to integer sequence of levels
y_scatter	numeric y values of corresponding raw data for scatterplot
jitter_scatter	boolean, if TRUE x_scatter will be randomly jittered by jitter function, with amount=jitter_scatter
dx_scatter	numeric value specifying amount of jittering used if jitter_scatter is TRUE
pch_scatter	pch characters of points in scatterplot; if non-null, must be single value or vector corresponding to x, otherwise selected automatically
col_scatter	color of points in scatterplot; if non-null, must be single value or vector corresponding to x, otherwise selected automatically
cex_scatter	numeric scaling factor of points in scatterplot
...	standard arguments to be passed to the plot function

Value

Returns a list containing xlim and ylim used in the plot

Author(s)

Clark R. Andersen <crandersen@mdanderson.org>

References

Cumming, G. (2014). The new statistics: Why and how. *Psychological Science*, 27, 7-29. <doi:10.1177/0956797613504966> pmid:24220629
<http://www.psychologicalscience.org/index.php/publications/observer/2014/march-14/theres-life-beyond-05.html>

Examples

```

#Show catseye plots for 4 groups with means of c(-3,2,-1,6)
# and standard errors of c(1,2,4,3)
catseyesplot(1:4,ymean=c(-3,2,-1,6),yse=c(1,2,4,3),xlab="",ylab="",main="4 Groups",xaxt="n")
axis(1,at=1:4,labels = c("Group1","Group2","Group3","Group4"))
#Optionally, add points and lines (usually lines only when joining time sequence)
lines(1:4,c(-3,2,-1,6),type="b")

#Using the labels option
catseyesplot(1:4,ymean=c(-3,2,-1,6),yse=c(1,2,4,3),xlab="",ylab="",labels =
  c("Group A","Group B","Group C","Group D"))
catseyesplot(1:4,ymean=c(-3,2,-1,6),yse=c(1,2,4,3),xlab="",ylab="",labels = TRUE)

#Demonstration of inclusion of scatterplots
datTest=data.frame(x=c(rep(1,10),rep(2,10),rep(3,10)),y=rnorm(10,30))
datTest$y[datTest$x==2]=datTest$y[datTest$x==2]+7
datTest$y[datTest$x==3]=datTest$y[datTest$x==3]+5
means=c(mean(datTest$y[datTest$x==1]),mean(datTest$y[datTest$x==2]),
  mean(datTest$y[datTest$x==3]))
ses=c(sd(datTest$y[datTest$x==1]),sd(datTest$y[datTest$x==2]),
  sd(datTest$y[datTest$x==3]))/sqrt(10)

catseyesplot(1:3,ymean=means,yse=ses,xlab="Group",ylab="",x_scatter = datTest$x,
  y_scatter = datTest$y)
catseyesplot(1:3,ymean=means,yse=ses,xlab="Group",ylab="",x_scatter = datTest$x,
  y_scatter = datTest$y,jitter_scatter = TRUE,xaxt="n")
axis(1,at=1:3,labels = c("Group1","Group2","Group3"))

#Demonstration of plotting of factor estimates by direct prediction from lm model
datTest$x=factor(datTest$x)
lm1=lm(y~x,data=datTest)
newdata=data.frame(x=c("1","2","3"))
pred_lm=predict(lm1,se.fit = TRUE,newdata=newdata,type="response")
catseyesplot(1:3,ymean=pred_lm$fit,yse=pred_lm$se.fit,xlab="Group",ylab="",
  plot.mean.line = TRUE,labels=TRUE,
  x_scatter = datTest$x,y_scatter = datTest$y,jitter_scatter = TRUE,xaxt="n")

#Demonstration of plotting of factor estimates from emmeans package
require(emmeans)
emmeans1=emmeans(lm1,~x)
#Assess differences between levels of x
pairs(emmeans1,adjust="tukey")
preds=confint(emmeans1)
catseyesplot(1:3,ymean=preds$emmean,yse=preds$SE,xlab="Group",ylab="",
  plot.mean.line = TRUE,labels=TRUE,
  x_scatter = datTest$x,y_scatter = datTest$y,jitter_scatter = TRUE,xaxt="n")
#Plot with variable x positions
catseyesplot(c(1,3.5,5),ymean=pred_lm$fit,yse=pred_lm$se.fit,xlab="Group",
  plot.mean.line = TRUE,labels=TRUE,
  ylab="",x_scatter = datTest$x,y_scatter = datTest$y,jitter_scatter = TRUE,xaxt="n")

#Demonstrate use of transformation function fTransform

```

```

#Create skewed y
set.seed(3142)
datTest=data.frame(x=c(rep(1,10),rep(2,10),rep(3,10)),y=rnorm(30,mean=0))
datTest$y[datTest$x==2]=datTest$y[datTest$x==2]+1
datTest$y[datTest$x==3]=datTest$y[datTest$x==3]+.5
datTest$y=exp(datTest$y)#Create skewed y
datTest$log_y=log(datTest$y+1)#Transform skewed y to normal distribution for analysis
qqnorm(datTest$y)
qqnorm(datTest$log_y)
plot(datTest$x,datTest$y)
plot(datTest$x,datTest$log_y)
means=c(mean(datTest$log_y[datTest$x==1]),mean(datTest$log_y[datTest$x==2]),
        mean(datTest$log_y[datTest$x==3]))
ses=c(sd(datTest$log_y[datTest$x==1]),sd(datTest$log_y[datTest$x==2]),
      sd(datTest$log_y[datTest$x==3]))/sqrt(10)
#Plot on log scale
catseyesplot(1:3,ymean=means,yse=ses,xlab="Group",ylab="",x_scatter = datTest$x,
            y_scatter = datTest$log_y,jitter_scatter = TRUE,xaxt="n",yaxt="n")
axis(1,at=1:3,labels = c("Group1","Group2","Group3"))
axis(2,at=log(c(0,1,2,4,8,16)+1),labels = c(0,1,2,4,8,16))
#Show catseye plot on original (skewed) scale
#Define function to invert data from log_y scale to y scale
fInvertLog<-function(y_vals) {exp(y_vals)-1}
catseyesplot(1:3,ymean=means,yse=ses,xlab="Group",ylab="",x_scatter = datTest$x,
            y_scatter = datTest$y,jitter_scatter = TRUE,xaxt="n",fTransform=fInvertLog)
axis(1,at=1:3,labels = c("Group1","Group2","Group3"))

#Logistic regression example (2 groups)
set.seed(3333)
datBin=data.frame(Group=factor(c(rep("A",15),rep("B",15))),
                  Y=c(rbinom(15,1,.8),rbinom(15,1,.5)))
sum(datBin$Y[datBin$Group=="A"])/sum(datBin$Group=="A")
sum(datBin$Y[datBin$Group=="B"])/sum(datBin$Group=="B")
glm1=glm(Y~Group-1,family = binomial,data=datBin)
summary(glm1)
(smr=coefficients(summary(glm1)))
#Plot Results on logit=log(odds) Scale
catseyesplot(1:2,smr[,1],smr[,2],xaxt="n",ylab="log(odds)",xlab="Group")
axis(1,at=c(1,2),labels = c("A","B"))
#Plot Results on Probability Scale
fInvLogit<-function(yy) {exp(yy)/(1+exp(yy))}
catseyesplot(1:2,smr[,1],smr[,2],xaxt="n",ylab="Probability",xlab="Group",
            fTransform = fInvLogit,ylim=c(0,1))
axis(1,at=c(1,2),labels = c("A","B"))

```

Index

catseyes, [2](#)
catseyesplot, [4](#)