# Package 'basicTrendline'

July 26, 2018

**Version** 2.0.3

**Date** 2018-07-26

**Title** Add Trendline and Confidence Interval of Basic Regression Models to Plot

**Maintainer** Weiping Mei <meiweipingg@163.com>

**Description** Plot, draw regression line and confidence interval, and show regression equation, R-square and P-value, as simple as possible, by using different models (``line2P'', ``line3P'', ``log2P'', ``exp2P'', ``exp3P'', ``power2P'', ``power3P'') built in the 'trendline()' function.

**Depends** R (>= 2.1.0)

**Imports** graphics, stats, scales, investr

**BugReports** https://github.com/PhDMeiwp/basicTrendline/issues

**License** GPL-3

**URL** https://github.com/PhDMeiwp/basicTrendline

**LazyData** true

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Weiping Mei [aut, cre, cph] (<https://orcid.org/0000-0001-6400-9862>),
Guangchuang Yu [aut] (<https://orcid.org/0000-0002-6485-8781>),
Jiangshan Lai [ctb],
Qiang Rao [ctb],
Yu Umezawa [ctb]

**Repository** CRAN

**Date/Publication** 2018-07-26 11:50:05 UTC

## R topics documented:

---

SSexp2P *Self-Starting Nls 'exp2P' Regression Model*

---

## Description

This selfStart model evaluates the power regression function (formula as: y=a*exp(b*x)). It has an initial attribute that will evaluate initial estimates of the parameters 'a' and 'b' for a given set of data.

## Usage

```
SSexp2P(predictor, a, b)
```

## Arguments

| | |
|---|---|
| predictor | a numeric vector of values at which to evaluate the model. |
| a, b | The numeric parameters responsing to the exp2P model. |

## Author(s)

Weiping Mei <meiweipingg@163.com>

## See Also

trendline, SSexp3P, SSpower3P, nls, selfStart

## Examples

```
library(basicTrendline)
x<-1:5
y<-c(2,4,8,20,25)
xy<-data.frame(x,y)
getInitial(y ~ SSexp2P(x,a,b), data = xy)
## Initial values are in fact the converged values

fitexp2P <- nls(y~SSexp2P(x,a,b), data=xy)
summary(fitexp2P)
```

SSexp3P *Self-Starting Nls 'exp3P' Regression Model*

## Description

This selfStart model evaluates the exponential regression function (formula as: y=a*exp(b*x)+c). It has an initial attribute that will evaluate initial estimates of the parameters a, b, and c for a given set of data.

## Usage

```
SSexp3P(predictor, a, b, c)
```

## Arguments

| | |
|---|---|
| predictor | a numeric vector of values at which to evaluate the model. |
| a, b, c | Three numeric parameters responsing to the exp3P model. |

## Author(s)

Weiping Mei <meiweipingg@163.com>

## See Also

trendline, SSexp3P, SSpower3P, nls, selfStart

## Examples

```
library(basicTrendline)
x<-1:5
y<-c(2,4,8,16,28)
xy<-data.frame(x,y)
getInitial(y ~ SSexp3P(x,a,b,c), data = xy)
## Initial values are in fact the converged values

fitexp3P <- nls(y~SSexp3P(x,a,b,c), data=xy)
summary(fitexp3P)
```

SSpower2P *Self-Starting Nls 'power2P' Regression Model*

## Description

This selfStart model evaluates the power regression function (formula as: y=a*x^b). It has an initial attribute that will evaluate initial estimates of the parameters 'a' and 'b' for a given set of data.

## Usage

```
SSpower2P(predictor, a, b)
```

## Arguments

| | |
|---|---|
| predictor | a numeric vector of values at which to evaluate the model. |
| a, b | The numeric parameters responsing to the exp2P model. |

## Author(s)

Weiping Mei <meiweipingg@163.com>

## See Also

trendline, SSexp3P, SSpower3P, nls, selfStart

## Examples

```
library(basicTrendline)
x<-1:5
y<-c(2,4,8,20,25)
xy<-data.frame(x,y)
getInitial(y ~ SSpower2P(x,a,b), data = xy)
## Initial values are in fact the converged values

fitpower2P <- nls(y~SSpower2P(x,a,b), data=xy)
summary(fitpower2P)
```

SSpower3P *Self-Starting Nls 'power3P' Regression Model*

## Description

This selfStart model evaluates the power regression function (formula as: y=a*x^b+c). It has an initial attribute that will evaluate initial estimates of the parameters a, b, and c for a given set of data.

## Usage

```
SSpower3P(predictor, a, b, c)
```

## Arguments

| | |
|---|---|
| predictor | a numeric vector of values at which to evaluate the model. |
| a, b, c | Three numeric parameters responsing to the exp3P model. |

## Author(s)

Weiping Mei <meiweipingg@163.com>

## See Also

trendline, SSexp3P, SSpower3P, nls, selfStart

## Examples

```
library(basicTrendline)
x<-1:5
y<-c(2,4,8,20,25)
xy<-data.frame(x,y)
getInitial(y ~ SSpower3P(x,a,b,c), data = xy)
## Initial values are in fact the converged values

fitpower3P <- nls(y~SSpower3P(x,a,b,c), data=xy)
summary(fitpower3P)
```

`trendline` *Add Trendline and Show Equation to Plot*

## Description

Plot, draw regression line and confidence interval, and show regression equation, R-square and P-value, as simple as possible, by using different models built in the 'trendline()' function. The function includes the following models in the latest version: "line2P" (formula as: y=a*x+b), "line3P" (y=a*x^2+b*x+c), "log2P" (y=a*ln(x)+b), "exp2P" (y=a*exp(b*x)),"exp3P" (y=a*exp(b*x)+c), "power2P" (y=a*x^b), and "power3P" (y=a*x^b+c). Besides, the summarized result of each fitted model is also output by default.

## Usage

```
trendline(x, y, model = "line2P", Pvalue.corrected = TRUE,
  linecolor = "blue", lty = 1, lwd = 1, show.equation = TRUE,
  show.Rpvalue = TRUE, Rname = 1, Pname = 0, xname = "x", yname = "y",
  yhat = FALSE, summary = TRUE, ePos.x = NULL, ePos.y = NULL,
  text.col = "black", eDigit = 5, eSize = 1, CI.fill = TRUE,
  CI.level = 0.95, CI.color = "grey", CI.alpha = 1, CI.lty = 1,
  CI.lwd = 1, las = 1, xlab = NULL, ylab = NULL, ...)
```

## Arguments

| | |
|---|---|
| `x, y` | the x and y arguments provide the x and y coordinates for the plot. Any reasonable way of defining the coordinates is acceptable. |
| `model` | select which model to fit. Default is "line2P". The "model" should be one of c("line2P", "line3P", "log2P", "exp2P", "exp3P", "power2P", "power3P"), their formulas are as follows:<br>"line2P": y=a*x+b<br>"line3P": y=a*x^2+b*x+c<br>"log2P": y=a*ln(x)+b<br>"exp2P": y=a*exp(b*x)<br>"exp3P": y=a*exp(b*x)+c<br>"power2P": y=a*x^b<br>"power3P": y=a*x^b+c |
| `Pvalue.corrected` | if P-value corrected or not, the value is one of c("TRUE", "FALSE"). |
| `linecolor` | color of regression line. |
| `lty` | line type. lty can be specified using either text c("blank","solid","dashed","dotted","dotdash","longdash","<br>or number c(0, 1, 2, 3, 4, 5, 6). Note that lty = "solid" is identical to lty=1. |
| `lwd` | line width. Default is 1. |
| `show.equation` | whether to show the regression equation, the value is one of c("TRUE", "FALSE"). |
| `show.Rpvalue` | whether to show the R-square and P-value, the value is one of c("TRUE", "FALSE"). |

| | |
|---|---|
| Rname | to specify the character of R-square, the value is one of c(0, 1), corresponding to c(r^2, R^2). |
| Pname | to specify the character of P-value, the value is one of c(0, 1), corresponding to c(p, P). |
| xname | to specify the character of "x" in equation, see Examples [case 5]. |
| yname | to specify the character of "y" in equation, see Examples [case 5]. |
| yhat | whether to add a hat symbol (^) on the top of "y" in equation. Default is FALSE. |
| summary | summarizing the model fits. Default is TRUE. |
| ePos.x, ePos.y | equation position. Default as ePos.x = "topleft". If no need to show equation, set ePos.x = NA. It's same as those in legend. |
| text.col | the color used for the equation text. |
| eDigit | the numbers of digits for equation parameters. Default is 5. |
| eSize | font size in percentage of equation. Default is 1. |
| CI.fill | fill the confidance interval? (TRUE by default, see 'CI.level' to control) |
| CI.level | level of confidence interval to use (0.95 by default) |
| CI.color | line or fill color of confidence interval. |
| CI.alpha | alpha value of fill color of confidence interval. |
| CI.lty | line type of confidence interval. |
| CI.lwd | line width of confidence interval. |
| las | style of axis labels. (0=parallel, 1=all horizontal, 2=all perpendicular to axis, 3=all vertical) |
| xlab, ylab | labels of x- and y-axis. |
| ... | additional parameters to plot, such as type, main, sub, pch, col. |

## Details

The linear models (line2P, line3P, log2P) in this package are estimated by lm function,
while the nonlinear models (exp2P, exp3P, power2P, power3P) are estimated by nls function (i.e., least-squares method).

The argument 'Pvalue.corrected' is workful for non-linear regression only.

If "Pvalue.corrected = TRUE", the P-value is calculated by using "Residual Sum of Squares" and "Corrected Total Sum of Squares (i.e. sum((y-mean(y))^2))".
If "Pvalue.corrected = TRUE", the P-value is calculated by using "Residual Sum of Squares" and "Uncorrected Total Sum of Squares (i.e. sum(y^2))".

## Note

Confidence intervals for nonlinear regression (i.e., objects of class nls) are based on the linear approximation described in Bates & Watts (2007) and Greenwell & Schubert-Kabban (2014).

## Author(s)

Weiping Mei, Guangchuang Yu

## References

Bates, D. M., and Watts, D. G. (2007) *Nonlinear Regression Analysis and its Applications*. Wiley.

Greenwell B. M., and Schubert-Kabban, C. M. (2014) *investr: An R Package for Inverse Estimation*. The R Journal, 6(1), 90-100.

## See Also

trendline, SSexp3P, SSpower3P, nls, selfStart, plotFit

## Examples

```
library(basicTrendline)
x <- c(1, 3, 6,  9,  13,   17)
y <- c(5, 8, 11, 13, 13.2, 13.5)

# [case 1] default
trendline(x, y, model="line2P", ePos.x = "topleft", summary=TRUE, eDigit=5)

# [case 2]  draw lines of confidenc interval only (set CI.fill = FALSE)
trendline(x, y, model="line3P", CI.fill = FALSE, CI.color = "black", CI.lty = 2, linecolor = "blue")

# [case 3]  draw trendliine only (set CI.color = NA)
trendline(x, y, model="log2P", ePos.x= "top", linecolor = "red", CI.color = NA)

# [case 4]  show regression equation only (set show.Rpvalue = FALSE)
trendline(x, y, model="exp2P", show.equation = TRUE, show.Rpvalue = FALSE)

# [case 5]  specify the name of parameters in equation
# see Arguments c('xname', 'yname', 'yhat', 'Rname', 'Pname').
trendline(x, y, model="exp3P", xname="T", yname=paste(delta^15,"N"),
          yhat=FALSE, Rname=1, Pname=0, ePos.x = "bottom")

# [case 6]  change the digits, font size, and color of equation.
trendline(x, y, model="power2P", eDigit = 3, eSize = 1.4, text.col = "blue")

# [case 7]  don't show equation (set ePos.x = NA)
trendline(x, y, model="power3P", ePos.x = NA)

# [case 8]  set graphical parameters by par {graphics}
### NOT RUN

par(mgp=c(1.5,0.4,0), mar=c(3,3,1,1), tck=-0.01, cex.axis=0.9)

trendline(x, y)

dev.off()
```

```
### END (NOT RUN)
```

---

`trendline_summary` *Summarized Results of Each Regression Model*

---

## Description

Summarizing the results of each regression model which built in the 'trendline()' function. The function includes the following models in the latest version: "line2P" (formula as: y=a*x+b), "line3P" (y=a*x^2+b*x+c), "log2P" (y=a*ln(x)+b), "exp2P" (y=a*exp(b*x)),"exp3P" (y=a*exp(b*x)+c), "power2P" (y=a*x^b), and "power3P" (y=a*x^b+c).

## Usage

```
trendline_summary(x, y, model = "line2P", Pvalue.corrected = TRUE,
  summary = TRUE, eDigit = 5)
```

## Arguments

| | |
|---|---|
| `x, y` | the x and y arguments provide the x and y coordinates for the plot. Any reasonable way of defining the coordinates is acceptable. |
| `model` | select which model to fit. Default is "line2P". The "model" should be one of c("line2P", "line3P", "log2P", "exp2P", "exp3P", "power2P", "power3P"), their formulas are as follows:<br>"line2P": y=a*x+b<br>"line3P": y=a*x^2+b*x+c<br>"log2P": y=a*ln(x)+b<br>"exp2P": y=a*exp(b*x)<br>"exp3P": y=a*exp(b*x)+c<br>"power2P": y=a*x^b<br>"power3P": y=a*x^b+c |
| `Pvalue.corrected` | if P-value corrected or not, the vlaue is one of c("TRUE", "FALSE"). |
| `summary` | summarizing the model fits. Default is TRUE. |
| `eDigit` | the numbers of digits for summarized results. Default is 5. |

## Details

The linear models (line2P, line3P, log2P) in this package are estimated by `lm` function, while the nonlinear models (exp2P, exp3P, power2P, power3P) are estimated by `nls` function (i.e., least-squares method).

The argument 'Pvalue.corrected' is workful for non-linear regression only.

If "Pvalue.corrected = TRUE", the P-vlaue is calculated by using "Residual Sum of Squares" and

"Corrected Total Sum of Squares (i.e. sum((y-mean(y))^2))".
If "Pvalue.corrected = TRUE", the P-vlaue is calculated by using "Residual Sum of Squares" and "Uncorrected Total Sum of Squares (i.e. sum(y^2))".

## Value

R^2, indicates the R-Squared value of each regression model.

p, indicates the p-value of each regression model.

N, indicates the sample size.

AIC or BIC, indicate the Akaike's Information Criterion or Bayesian Information Criterion for fitted model. Click AIC for details. The smaller the AIC or BIC, the better the model.

RSS, indicate the value of "Residual Sum of Squares".

## Author(s)

Weiping Mei, Guangchuang Yu

## See Also

trendline, SSexp3P, SSpower3P, nls, selfStart

## Examples

```
library(basicTrendline)
x1<-1:5
x2<- -2:2
x3<- c(101,105,140,200,660)
x4<- -5:-1
x5<- c(1,30,90,180,360)

y1<-c(2,14,18,19,20)        # increasing convex trend
y2<- c(-2,-14,-18,-19,-20)  # decreasing concave trend
y3<-c(2,4,16,38,89)         # increasing concave trend
y4<-c(-2,-4,-16,-38,-89)    # decreasing convex trend
y5<- c(600002,600014,600018,600019,600020) # high y values with low range.

trendline_summary(x1,y1,model="line2P",summary=TRUE,eDigit=10)
trendline_summary(x2,y2,model="line3P",summary=FALSE)
trendline_summary(x3,y3,model="log2P")
trendline_summary(x4,y4,model="exp3P")
trendline_summary(x5,y5,model="power3P")
```

# Index