Package ‘easynls’

October 13, 2022

Type Package

Title Easy Nonlinear Model

Version 5.0

Date 2017-11-14

Author Emmanuel Arnhold

Maintainer Emmanuel Arnhold <emmanuelarnhold@yahoo.com.br>

Description Fit and plot some nonlinear models.

Depends R (>= 3.0.0)

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2017-11-15 07:44:13 UTC

R topics documented:

`easynls-package` *Easy Nonlinear Model*

Description

The package fit and plot some nonlinear models

Details

Package:	easynls
Type:	Package
Version:	5.0
Date:	2017-11-14
License:	GPL-2

Author(s)

Emmanuel Arnhold <emmanuelarnhold@yahoo.com.br>

References

KAPS, M. and LAMBERSON, W. R. Biostatistics for Animal Science: an introductory text. 2nd Edition. CABI Publishing, Wallingford, Oxfordshire, UK, 2009. 504p.

Examples

```
time=c(2,6,9,24,48,72,96)
deg=c(20,33,46,55,66,72,76)

data5=data.frame(time,deg)

nlsfit(data5, model=12)
```

`nlsfit` *Fit nonlinear models*

Description

The function fit some nonlinear models

Usage

```
nlsfit(data, model = 1, start = c(a = 1, b = 1, c = 1, d = 1, e = 1))
```

Arguments

`data` data is a data.frame

The first column should contain the treatments (explanatory variable) and the remaining columns the response variables.

`model` define the model

1 = "y~a+b*x" linear
2 = "y~a+b*x+c*x^2" quadratic
3 = "y ~ a + b * (x - c) * (x <= c)" linear plateau
4 = "y ~ (a + b * x + c * I(x^2)) * (x <= -0.5 * b/c) + (a + I(-b^2/(4 * c))) * (x > -0.5 * b/c)" quadratic plateau
5 = "ifelse(x>=d,(a-c*d)+(b+c)*x, a+b*x)" two linear
6 = "y~a*exp(b*x)" exponential
7 = "y~a*(1+b*(exp(-c*x)))^-1" logistic
8 = "y~a*(1-b*(exp(-c*x)))^3" van bertalanffy
9 = "y~a*(1-b*(exp(-c*x)))" brody
10 = "y~a*exp(-b*exp(-c*x)" gompertz
11 = "y~(a*x^b)*exp(-c*x)" lactation curve
12 = "y ~ a + b * (1 - exp(-c * x))" ruminal degradation curve
13 = "y~(a/(1+exp(2-4*c*(x-e))))+(b/(1+exp(2-4*d*(x-e))))" logistic bi-compartmental

`start` start iterations values of model

Value

Returns coefficients of the models, t test for coefficients, R squared, adjusted R squared, AIC, BIC and the maximum (or minimum) values of y and critical point of x

Author(s)

Emmanuel Arnhold <emmanuelarnhold@yahoo.com.br>

References

KAPS, M. and LAMBERSON, W. R. Biostatistics for Animal Science: an introductory text. 2nd Edition. CABI Publishing, Wallingford, Oxfordshire, UK, 2009. 504p.

See Also

nls, nls2

Examples

```
# data represent weights of an Angus cow at ages from 8 to 108 months (Kaps and Lamberson, 2009)

weight=c(280,340,430,480,550,580,590,600,590,600)
age=c(8,12,24,36,48,60,72,84,96,108)

data1=data.frame(age, weight)

# linear
nlsfit(data1, model=1)

# quadratic
```

```
nlsfit(data1, model=2)

# linear plateau
nlsfit(data1, model=3)

# quadratic plateau
nlsfit(data1, model=4)

# two linear
nlsfit(data1, model=5, start=c(250,6,2,50))

# exponential
nlsfit(data1, model=6, start=c(250,0.05))

# logistic
nlsfit(data1, model=7, start=c(600,4,0.05))

# van bertalanffy
nlsfit(data1, model=8, start=c(600,2,0.05))

# brody
nlsfit(data1, model=9, start=c(600,4,0.05))

# gompertz
nlsfit(data1, model=10, start=c(600,4,0.05))



# describe the growth of Zagorje turkeys (Kaps and Lamberson, 2009)


weight=c(44,66,100,150,265,370,455,605,770)
age=c(1,7,14,21,28,35,42,49,56)

data2=data.frame(age,weight)

# two linear
nlsfit(data2, model=5, start=c(25,6,10,20))

# using segmented regression to estimate a plateau
# the requirement for the methionine will be estimated measurements of gain of turkey poults
#(Kaps and Lamberson, 2009)

methionine=c(80,85,90,95,100,105,110,115,120)
gain=c(102,115,125,133,140,141,142,140,142)

data3=data.frame(methionine, gain)

# linear
nlsfit(data3, model=1)

# quadratic
nlsfit(data3, model=2)
```

```
# linear plateau
nlsfit(data3, model=3)

# quadratic plateau
nlsfit(data3, model=4)

# lactation curve
 milk=c(25,24,26,28,30,31,27,26,25,24,23,24,22,21,22,20,21,19,
18,17,18,18,16,17,15,16,14)
 days=c(15,15,15,75,75,75,135,135,135,195,195,195,255,255,255,
315,315,315,375,375,375,435,435,435,495,495,495)

 data4=data.frame(days,milk)


nlsfit(data4, model=11, start=c(16,0.25,0.004))

# ruminal degradation
time=c(2,6,9,24,48,72,96)
deg=c(20,33,46,55,66,72,76)

data5=data.frame(time,deg)

nlsfit(data5, model=12)

# logistic bi-compartmental (gas production)
time=c(0,12,24,36,48,60,72,84,96,108,120,144,168,192)
gas=c(0.002,3.8,8,14.5,16,16.5,17,17.4,17.9,18.1,18.8,
19,19.2,19.3)

data6=data.frame(time,gas)

nlsfit(data6, model=13, start=c(19,4,0.025,0.004,5))
```

`nlsplot` *Plot nonlinear models*

Description

The function plot some nonlinear models

Usage

```
nlsplot(data, model = 1, start = c(a = 1, b = 1, c = 1, d = 1, e = 1),
xlab = "Explanatory Variable" , ylab = "Response Variable", position = 1)
```

Arguments

`data` data is a data.frame

The first column should contain the treatments (explanatory variable) and the remaining columns the response variables.

`model` define the model

1 = "y~a+b*x" linear
2 = "y~a+b*x+c*x^2" quadratic
3 = "y ~ a + b * (x - c) * (x <= c)" linear plateau
4 = "y ~ (a + b * x + c * I(x^2)) * (x <= -0.5 * b/c) + (a + I(-b^2/(4 * c))) * (x > -0.5 * b/c)" quadratic plateau
5 = "ifelse(x>=d,(a-c*d)+(b+c)*x, a+b*x)" two linear
6 = "y~a*exp(b*x)" exponential
7 = "y~a*(1+b*(exp(-c*x)))^-1" logistic
8 = "y~a*(1-b*(exp(-c*x)))^3" van bertalanffy
9 = "y~a*(1-b*(exp(-c*x)))" brody
10 = "y~a*exp(-b*exp(-c*x)" gompertz
11 = "y~(a*x^b)*exp(-c*x)" lactation curve
12 = "y ~ a + b * (1 - exp(-c * x))" ruminal degradation curve
13 = "y~(a/(1+exp(2-4*c*(x-e))))+(b/(1+exp(2-4*d*(x-e))))" logistic bi-compartmental

`start` start iterations values of model

`xlab` names of variable x

`ylab` names of variable y

`position` position of equation in the graph

top=1
bottomright=2
bottom=3
bottomleft=4
left=5
topleft=6 (default)
topright=7
right=8
center=9

Author(s)

Emmanuel Arnhold <emmanuelarnhold@yahoo.com.br>

References

KAPS, M. and LAMBERSON, W. R. Biostatistics for Animal Science: an introductory text. 2nd Edition. CABI Publishing, Wallingford, Oxfordshire, UK, 2009. 504p.

See Also

nlsfit, nls,nls2

Examples

```
# weights of an Angus cow at ages from 8 to 108 months (Kaps and Lamberson, 2009)

weight=c(280,340,430,480,550,580,590,600,590,600)
age=c(8,12,24,36,48,60,72,84,96,108)

data1=data.frame(age, weight)

# linear
nlsplot(data1, model=1)

# quadratic
nlsplot(data1, model=2)

# linear plateau
nlsplot(data1, model=3)

# quadratic plateau
nlsplot(data1, model=4)

# two linear
nlsplot(data1, model=5, start=c(250,6,2,50))

# exponential
nlsplot(data1, model=6, start=c(250,0.05))

# logistic
nlsplot(data1, model=7, start=c(600,4,0.05))

# van bertalanffy
nlsplot(data1, model=8, start=c(600,2,0.05))

# brody
nlsplot(data1, model=9, start=c(600,4,0.05))

# gompertz
nlsplot(data1, model=10, start=c(600,4,0.05))



# growth of Zagorje turkeys (Kaps and Lamberson, 2009)


weight=c(44,66,100,150,265,370,455,605,770)
age=c(1,7,14,21,28,35,42,49,56)

data2=data.frame(age,weight)

# two linear
nlsplot(data2, model=5, start=c(25,6,10,20))

# using segmented regression to estimate a plateau
```

```
# requirement for the methionine will be estimated measurements gain of turkey poults
#(Kaps and Lamberson, 2009)

methionine=c(80,85,90,95,100,105,110,115,120)
gain=c(102,115,125,133,140,141,142,140,142)

data3=data.frame(methionine, gain)

# linear
nlsplot(data3, model=1)

# quadratic
nlsplot(data3, model=2)

# linear plateau
nlsplot(data3, model=3)

# quadratic plateau
nlsplot(data3, model=4)

# lactation curve
 milk=c(25,24,26,28,30,31,27,26,25,24,23,24,22,21,22,20,21,19,18,17,18,
18,16,17,15,16,14)
 days=c(15,15,15,75,75,75,135,135,135,195,195,195,255,255,255,315,315,
315,375,375,375,435,435,435,495,495,495)

 data4=data.frame(days,milk)


nlsplot(data4, model=11, start=c(16,0.25,0.004))

# ruminal degradation
time=c(2,6,9,24,48,72,96)
deg=c(20,33,46,55,66,72,76)

data5=data.frame(time,deg)

nlsplot(data5, model=12)

# logistic bi-compartmental (gas production)
time=c(0,12,24,36,48,60,72,84,96,108,120,144,168,192)
gas=c(0.002,3.8,8,14.5,16,16.5,17,17.4,17.9,18.1,18.8,19,19.2,19.3)

data6=data.frame(time,gas)

nlsplot(data6, model=13, start=c(19,4,0.025,0.004,5))
```

Index