Package ‘NISTnls’

October 12, 2022

Version 0.9-13

Date 2012-09-05

Title Nonlinear least squares examples from NIST

Maintainer Douglas Bates <bates@stat.wisc.edu>

Author original from National Institutes for Standards and Technology (NIST) http://www.itl.nist.gov/div898/strd/nls/nls_main.shtml R port by Douglas Bates <bates@stat.wisc.edu>

Description Datasets for testing nonlinear regression routines.

Depends stats, R (>= 2.14.0)

LazyData yes

License GPL (>= 2)

LicenseDetails The original data sets and descriptions are from the NIST web site and were not covered by an explicit copyright. Modifications for R data sets are covered by GPL (>= 2).

Repository CRAN

Date/Publication 2012-09-06 05:18:31

NeedsCompilation no

R topics documented:

Bennett5 — *Magentization modelling*

Description

The `Bennett5` data frame has 154 rows and 2 columns of data from a magnetism study

Format

This data frame contains the following columns:

- **y** A numeric vector of magnetism values.
- **x** A numeric vector of log(time).

Details

These data are the result of a NIST study involving superconductivity magnetization modeling. The response variable is magnetism, and the predictor variable is the log of time in minutes.

Source

Bennett, L., L. Swartzendruber, and H. Brown, NIST (1994). Superconductivity Magnetization Modeling.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Bennett5)
Try(fm1 <- nls(y ~ b1*(b2+x)**(-1/b3), data = Bennett5,
           start = c(b1 = -2000, b2 = 50, b3 = 0.8), trace = TRUE))
Try(fm1a <- nls(y ~ b1*(b2+x)**(-1/b3), data = Bennett5,
            start = c(b1 = -2000, b2 = 50, b3 = 0.8),
```

```
            trace = TRUE, alg = "port"))
Try(fm2 <- nls(y ~ b1*(b2+x)**(-1/b3), data = Bennett5,
           start = c(b1 = -1500, b2 = 45, b3 = 0.85), trace = TRUE))
Try(fm2a <- nls(y ~ b1*(b2+x)**(-1/b3), data = Bennett5,
            start = c(b1 = -1500, b2 = 45, b3 = 0.85),
            trace = TRUE, alg = "port"))
Try(fm3 <- nls(y ~ (b2+x)**(-1/b3), data = Bennett5, alg = "plinear",
           start = c( b2 = 50, b3 = 0.8), trace = TRUE))
Try(fm4 <- nls(y ~ (b2+x)**(-1/b3), data = Bennett5, alg = "plinear",
           start = c( b2 = 45, b3 = 0.8), trace = TRUE))
```

Chwirut1 *Ultrasonic calibration study 1*

Description

The Chwirut1 data frame has 214 rows and 2 columns giving

Format

This data frame contains the following columns:

y A numeric vector of ultrasonic response values

x A numeric vector or metal distance values

Details

These data are the result of a NIST study involving ultrasonic calibration. The response variable is ultrasonic response, and the predictor variable is metal distance.

Source

Chwirut, D., NIST (197?). Ultrasonic Reference Block Study.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Chwirut1)
Try(fm1 <- nls(y ~ exp(-b1*x)/(b2+b3*x), data = Chwirut1, trace = TRUE,
           start = c(b1 = 0.1, b2 = 0.01, b3 = 0.02)))
Try(fm1a <- nls(y ~ exp(-b1*x)/(b2+b3*x), data = Chwirut1, trace = TRUE,
           start = c(b1 = 0.1, b2 = 0.01, b3 = 0.02), alg = "port"))
Try(fm2 <- nls(y ~ exp(-b1*x)/(b2+b3*x), data = Chwirut1, trace = TRUE,
           start = c(b1 = 0.15, b2 = 0.008, b3 = 0.010)))
Try(fm2a <- nls(y ~ exp(-b1*x)/(b2+b3*x), data = Chwirut1, trace = TRUE,
            start = c(b1 = 0.15, b2 = 0.008, b3 = 0.010), alg = "port"))
Try(fm3 <- nls(y ~ exp(-b1*x)/(1+p3*x), data = Chwirut1, trace = TRUE,
           start = c(b1 = 0.1, p3 = 0.02/0.01), algorithm = "plinear"))
Try(fm4 <- nls(y ~ exp(-b1*x)/(1+p3*x), data = Chwirut1, trace = TRUE,
           start = c(b1 = 0.15, p3 = 0.01/0.008), algorithm = "plinear"))
```

Chwirut2 *Ultrasonic calibration data 2*

Description

The Chwirut2 data frame has nr rows and nc columns giving

Format

This data frame contains the following columns:

y A numeric vector of ultrasonic response values.

x A numeric vector of metal distance values.

Details

These data are the result of a NIST study involving ultrasonic calibration. The response variable is ultrasonic response, and the predictor variable is metal distance.

Source

Chwirut, D., NIST (197?). Ultrasonic Reference Block Study.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Chwirut2)
Try(fm1 <- nls(y ~ exp(-b1*x)/(b2+b3*x), data = Chwirut2, trace = TRUE,
            start = c(b1 = 0.1 , b2 = 0.01, b3 = 0.02)))
Try(fm1a <- nls(y ~ exp(-b1*x)/(b2+b3*x), data = Chwirut2, trace = TRUE,
             start = c(b1 = 0.1 , b2 = 0.01, b3 = 0.02), alg = "port"))
Try(fm2 <- nls(y ~ exp(-b1*x)/(b2+b3*x), data = Chwirut2, trace = TRUE,
            start = c(b1 = 0.15 , b2 = 0.008, b3 = 0.01)))
Try(fm2a <- nls(y ~ exp(-b1*x)/(b2+b3*x), data = Chwirut2, trace = TRUE,
             start = c(b1 = 0.15 , b2 = 0.008, b3 = 0.01), alg = "port"))
Try(fm3 <- nls(y ~ exp(-b1*x)/(1+p3*x), data = Chwirut2, trace = TRUE,
           start = c(b1 = 0.1, p3 = 2.), alg = "plinear"))
Try(fm4 <- nls(y ~ exp(-b1*x)/(1+p3*x), data = Chwirut2, trace = TRUE,
           start = c(b1 = 0.15, p3 = 0.01/0.008), alg = "plinear"))
```

DanielWood *Radiated energy*

Description

The DanielWood data frame has 6 rows and 2 columns giving the energy radiated from a carbon filament versus the absolute temperature of the filament.

Format

This data frame contains the following columns:

y A numeric vector of the energy radiated from a carbon filament lamp.

x A numeric vector of the temperature of the filament (1000 K).

Details

These data and model are described in Daniel and Wood (1980), and originally published in E.S.Keeping, "Introduction to Statistical Inference," Van Nostrand Company, Princeton, NJ, 1962, p. 354. The response variable is energy radiated from a carbon filament lamp per cm**2 per second, and the predictor variable is the absolute temperature of the filament in 1000 degrees Kelvin.

Source

Daniel, C. and F. S. Wood (1980). Fitting Equations to Data, Second Edition. New York, NY: John Wiley and Sons, pp. 428-431.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = DanielWood)
Try(fm1 <- nls(y ~ b1*x**b2, data = DanielWood, trace = TRUE,
            start = c(b1 = 1, b2 = 5)))
Try(fm1a <- nls(y ~ b1*x**b2, data = DanielWood, trace = TRUE,
            start = c(b1 = 1, b2 = 5), alg = "port"))
Try(fm2 <- nls(y ~ b1*x**b2, data = DanielWood, trace = TRUE,
            start = c(b1 = 0.7, b2 = 4)))
Try(fm2a <- nls(y ~ b1*x**b2, data = DanielWood, trace = TRUE,
            start = c(b1 = 0.7, b2 = 4), alg = "port"))
Try(fm3 <- nls(y ~ x**b2, data = DanielWood, trace = TRUE,
            start = c(b2 = 5), algorithm = "plinear"))
Try(fm4 <- nls(y ~ x**b2, data = DanielWood, trace = TRUE,
            start = c(b2 = 4), algorithm = "plinear"))
```

Eckerle4 *Circular interference data*

Description

The Eckerle4 data frame has 35 rows and 2 columns giving transmittance as a function of wavelength.

Format

This data frame contains the following columns:

y A numeric vector of transmittance values.

x A numeric vector of wavelengths.

Details

These data are the result of a NIST study involving circular interference transmittance. The response variable is transmittance, and the predictor variable is wavelength.

Source

Eckerle, K., NIST (197?). Circular Interference Transmittance Study.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Eckerle4)
## should fail - ridiculous starting value for b3
Try(fm1 <- nls(y ~ (b1/b2) * exp(-0.5*((x-b3)/b2)**2), Eckerle4,
               trace = TRUE,
               start = c(b1 = 1, b2 = 10, b3 = 500)))
Try(fm1a <- nls(y ~ (b1/b2) * exp(-0.5*((x-b3)/b2)**2), Eckerle4,
                  trace = TRUE, alg = "port",
                  start = c(b1 = 1, b2 = 10, b3 = 500)))
Try(fm2 <- nls(y ~ (b1/b2) * exp(-0.5*((x-b3)/b2)**2),
            Eckerle4, trace = TRUE,
            start = c(b1 = 1.5, b2 = 5, b3 = 450)))
Try(fm2a <- nls(y ~ (b1/b2) * exp(-0.5*((x-b3)/b2)**2),
            Eckerle4, trace = TRUE, alg = "port",
            start = c(b1 = 1.5, b2 = 5, b3 = 450)))
## should fail - ridiculous starting value for b3
Try(fm3 <- nls(y ~ (1/b2) * exp(-0.5*((x-b3)/b2)**2),
               Eckerle4, trace = TRUE,
               start = c(b2 = 10, b3 = 500), algorithm = "plinear"))
Try(fm4 <- nls(y ~ (1/b2) * exp(-0.5*((x-b3)/b2)**2), Eckerle4, trace = TRUE,
           start = c(b2 = 5, b3 = 450), algorithm = "plinear"))
```

ENSO *Atmospheric pressure differences*

Description

The ENSO data frame has 168 rows and 2 columns giving atmospheric pressure differences over time.

Format

This data frame contains the following columns:

y A numeric vector of monthly averaged atmospheric pressure differences between Easter Island and Darwin, Australia.

x A numeric vector of time values.

Details

The data are monthly averaged atmospheric pressure differences between Easter Island and Darwin, Australia. This difference drives the trade winds in the southern hemisphere. Fourier analysis of the data reveals 3 significant cycles. The annual cycle is the strongest, but cycles with periods of approximately 44 and 26 months are also present. These cycles correspond to the El Nino and the Southern Oscillation. Arguments to the SIN and COS functions are in radians.

Source

Kahaner, D., C. Moler, and S. Nash, (1989). Numerical Methods and Software. Englewood Cliffs, NJ: Prentice Hall, pp. 441-445.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = ENSO)
plot(y ~ x, data = ENSO, type = "l")  # to see the pattern more clearly
Try(fm1 <- nls(y ~ b1 + b2*cos( 2*pi*x/12 ) + b3*sin( 2*pi*x/12 )
                   + b5*cos( 2*pi*x/b4 ) + b6*sin( 2*pi*x/b4 )
                   + b8*cos( 2*pi*x/b7 ) + b9*sin( 2*pi*x/b7 ),
            data = ENSO, trace = TRUE,
            start = c(b1 = 11.0, b2 = 3.0, b3 = 0.5, b4 = 40.0, b5 = -0.7,
                      b6 = -1.3, b7 = 25.0, b8 = -0.3, b9 = 1.4)))
Try(fm1a <- nls(y ~ b1 + b2*cos( 2*pi*x/12 ) + b3*sin( 2*pi*x/12 )
                    + b5*cos( 2*pi*x/b4 ) + b6*sin( 2*pi*x/b4 )
                    + b8*cos( 2*pi*x/b7 ) + b9*sin( 2*pi*x/b7 ),
             data = ENSO, trace = TRUE, alg = "port",
             start = c(b1 = 11.0, b2 = 3.0, b3 = 0.5, b4 = 40.0, b5 = -0.7,
                       b6 = -1.3, b7 = 25.0, b8 = -0.3, b9 = 1.4)))
Try(fm2 <- nls(y ~ b1 + b2*cos( 2*pi*x/12 ) + b3*sin( 2*pi*x/12 )
                   + b5*cos( 2*pi*x/b4 ) + b6*sin( 2*pi*x/b4 )
                   + b8*cos( 2*pi*x/b7 ) + b9*sin( 2*pi*x/b7 ),
```

```
            data = ENSO, trace = TRUE,
            start = c(b1 = 10.0, b2 =  3.0, b3 =  0.5, b4 = 44.0, b5 = -1.5,
                     b6 =  0.5, b7 = 26.0, b8 = -0.1, b9 =  1.5)))
Try(fm2a <- nls(y ~ b1 + b2*cos( 2*pi*x/12 ) + b3*sin( 2*pi*x/12 )
                    + b5*cos( 2*pi*x/b4 ) + b6*sin( 2*pi*x/b4 )
                    + b8*cos( 2*pi*x/b7 ) + b9*sin( 2*pi*x/b7 ),
             data = ENSO, trace = TRUE, alg = "port",
             start = c(b1 = 10.0, b2 =  3.0, b3 =  0.5, b4 = 44.0, b5 = -1.5,
                     b6 =  0.5, b7 = 26.0, b8 = -0.1, b9 =  1.5)))
Try(fm3 <- nls(y ~ cbind(1, cos( 2*pi*x/12 ), sin( 2*pi*x/12 ), cos( 2*pi*x/b4 ),
                      sin( 2*pi*x/b4 ), cos( 2*pi*x/b7 ), sin( 2*pi*x/b7 )),
            data = ENSO, trace = TRUE,
            start = c(b4 = 40.0, b7 = 25.0), algorithm = "plinear"))
Try(fm4 <- nls(y ~ cbind(1, cos( 2*pi*x/12 ), sin( 2*pi*x/12 ), cos( 2*pi*x/b4 ),
                    sin( 2*pi*x/b4 ), cos( 2*pi*x/b7 ), sin( 2*pi*x/b7 )),
            data = ENSO, trace = TRUE,
            start = c(b4 = 44.0, b7 = 26.0), algorithm = "plinear"))
```

Gauss1 *Generated data*

Description

The Gauss1 data frame has 250 rows and 2 columns of generated data.

Format

This data frame contains the following columns:

y A numeric vector of generated responses.

x A numeric vector of generated input values.

Details

The data are generated data with two well-separated Gaussians on a decaying exponential baseline plus normally distributed zero-mean noise with variance = 6.25.

Source

Rust, B., NIST (1996).

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Gauss1)
Try(fm1 <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss1, trace = TRUE,
            start = c(b1 = 97.0, b2 = 0.009, b3 = 100.0, b4 = 65.0, b5 = 20.0,
                     b6 = 70.0, b7 = 178., b8 = 16.5)))
Try(fm1a <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
```

```
                + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss1, trace = TRUE,
             start = c(b1 = 97.0, b2 = 0.009, b3 = 100.0, b4 = 65.0, b5 = 20.0,
                     b6 = 70.0, b7 = 178., b8 = 16.5), alg = "port"))
Try(fm2 <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss1, trace = TRUE,
           start = c(b1 = 94.0, b2 =  0.0105, b3 = 99.0, b4 = 63.0, b5 = 25.0,
                     b6 = 71.0, b7 = 180.0, b8 = 20.0)))
Try(fm2a <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss1, trace = TRUE,
           start = c(b1 = 94.0, b2 =  0.0105, b3 = 99.0, b4 = 63.0, b5 = 25.0,
                     b6 = 71.0, b7 = 180.0, b8 = 20.0), alg = "port"))
Try(fm3 <- nls(y ~ cbind(exp(-b2*x), exp(-(x-b4)**2/b5**2), exp(-(x-b7)**2/b8**2)),
           data = Gauss1, trace = TRUE,
           start = c( b2 = 0.009, b4 = 65.0, b5 = 20.0, b7 = 178., b8 = 16.5),
           algorithm = "plinear"))
Try(fm4 <- nls(y ~ cbind(exp(-b2*x), exp(-(x-b4)**2/b5**2), exp(-(x-b7)**2/b8**2)),
           data = Gauss1, trace = TRUE,
           start = c( b2 = 0.0105, b4 = 63.0, b5 = 25.0, b7 = 180., b8 = 20.0),
           algorithm = "plinear"))
```

Gauss2 *Generated data*

Description

The Gauss2 data frame has 250 rows and 2 columns giving

Format

This data frame contains the following columns:

y A numeric vector of generated response values.

x A numeric vector of generated input values.

Details

The data are two slightly-blended Gaussians on a decaying exponential baseline plus normally distributed zero-mean noise with variance = 6.25.

Source

Rust, B., NIST (1996)

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Gauss2)
Try(fm1 <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss2, trace = TRUE,
           start = c(b1 = 96, b2 = 0.009, b3 = 103, b4 = 106, b5 = 18,
```

```
                     b6 = 72, b7 = 151, b8 = 18)))
Try(fm1a <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss2, trace = TRUE,
             start = c(b1 = 96, b2 = 0.009, b3 = 103, b4 = 106, b5 = 18,
                       b6 = 72, b7 = 151, b8 = 18), alg = "port"))
Try(fm2 <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss2, trace = TRUE,
           start = c(b1 = 98, b2 = 0.0105, b3 = 103, b4 = 105, b5 = 20,
                     b6 = 73, b7 = 150, b8 = 20)))
Try(fm2a <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss2, trace = TRUE,
           start = c(b1 = 98, b2 = 0.0105, b3 = 103, b4 = 105, b5 = 20,
                     b6 = 73, b7 = 150, b8 = 20), alg = "port"))
Try(fm3 <- nls(y ~ cbind(exp(-b2*x), exp(-(x-b4)**2/b5**2), exp(-(x-b7)**2/b8**2)),
           data = Gauss2, trace = TRUE,
           start = c(b2 = 0.009, b4 = 106, b5 = 18, b7 = 151, b8 = 18),
           algorithm = "plinear"))
Try(fm4 <- nls(y ~ cbind(exp(-b2*x), exp(-(x-b4)**2/b5**2), exp(-(x-b7)**2/b8**2)),
           data = Gauss2, trace = TRUE,
           start = c(b2 = 0.0105, b4 = 105, b5 = 20, b7 = 150, b8 = 20),
           algorithm = "plinear"))
```

`Gauss3` *Generated data*

Description

The `Gauss3` data frame has 250 rows and 2 columns giving generated data of Gaussian peaks with a decaying exponential background.

Format

This data frame contains the following columns:

y A numeric vector of generated responses.

x A numeric vector of generated inputs.

Details

The data are two strongly-blended Gaussians on a decaying exponential baseline plus normally distributed zero-mean noise with variance = 6.25.

Source

Rust, B., NIST (1996).

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Gauss3)
Try(fm1 <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss3, trace = TRUE,
           start = c(b1 = 94.9, b2 = 0.009, b3 = 90.1, b4 = 113, b5 = 20,
                     b6 = 73.8, b7 = 140, b8 = 20)))
Try(fm1a <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
                + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss3, trace = TRUE,
            start = c(b1 = 94.9, b2 = 0.009, b3 = 90.1, b4 = 113, b5 = 20,
                      b6 = 73.8, b7 = 140, b8 = 20), alg = "port"))
Try(fm2 <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss3, trace = TRUE,
           start = c(b1 = 96, b2 = 0.0096, b3 = 80, b4 = 110, b5 = 25,
                     b6 = 74, b7 = 139, b8 = 25)))
Try(fm2a <- nls(y ~ b1*exp( -b2*x ) + b3*exp( -(x-b4)**2 / b5**2 )
               + b6*exp( -(x-b7)**2 / b8**2 ), data = Gauss3, trace = TRUE,
           start = c(b1 = 96, b2 = 0.0096, b3 = 80, b4 = 110, b5 = 25,
                     b6 = 74, b7 = 139, b8 = 25), alg = "port"))
Try(fm3 <- nls(y ~ cbind(exp(-b2*x), exp(-(x-b4)**2/b5**2), exp(-(x-b7)**2/b8**2)),
           data = Gauss3, trace = TRUE,
           start = c(b2 = 0.009, b4 = 113, b5 = 20, b7 = 140, b8 = 20),
           algorithm = "plinear"))
Try(fm4 <- nls(y ~ cbind(exp(-b2*x), exp(-(x-b4)**2/b5**2), exp(-(x-b7)**2/b8**2)),
           data = Gauss3, trace = TRUE,
           start = c(b2 = 0.0096, b4 = 110, b5 = 25, b7 = 139, b8 = 25),
           algorithm = "plinear"))
```

Hahn1 *Thermal expansion data*

Description

The Hahn1 data frame has 236 rows and 2 columns of data from a study on the thermal expansion of copper.

Format

This data frame contains the following columns:

y A numeric vector of values of the coefficient of thermal expansion.

x A numeric vector of temperatures (K).

Details

These data are the result of a NIST study involving the thermal expansion of copper. The response variable is the coefficient of thermal expansion, and the predictor variable is temperature in degrees Kelvin.

Source

Hahn, T., NIST (197?). Copper Thermal Expansion Study.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Hahn1)
Try(fm1 <- nls(y ~ (b1+b2*x+b3*x**2+b4*x**3) / (1+b5*x+b6*x**2+b7*x**3),
           data = Hahn1, trace = TRUE,
           start = c(b1 = 10, b2 = -1, b3 = 0.05,
             b4 = -0.00001, b5 = -0.05, b6 = 0.001, b7 = -0.000001)))
Try(fm1a <- nls(y ~ (b1+b2*x+b3*x**2+b4*x**3) / (1+b5*x+b6*x**2+b7*x**3),
           data = Hahn1, trace = TRUE, alg = "port",
           start = c(b1 = 10, b2 = -1, b3 = 0.05,
             b4 = -0.00001, b5 = -0.05, b6 = 0.001, b7 = -0.000001)))
Try(fm2 <- nls(y ~ (b1+b2*x+b3*x**2+b4*x**3) / (1+b5*x+b6*x**2+b7*x**3),
           data = Hahn1, trace = TRUE,
           start = c(b1 = 1, b2 = -0.1, b3 = 0.005, b4 = -0.000001,
              b5 = -0.005, b6 = 0.0001, b7 = -0.0000001)))
Try(fm2a <- nls(y ~ (b1+b2*x+b3*x**2+b4*x**3) / (1+b5*x+b6*x**2+b7*x**3),
             data = Hahn1, trace = TRUE, alg = "port",
           start = c(b1 = 1, b2 = -0.1, b3 = 0.005, b4 = -0.000001,
              b5 = -0.005, b6 = 0.0001, b7 = -0.0000001)))
Try(fm3 <- nls(y ~ cbind(1, x, x^2, x^3)/(1+x*(b5+x*(b6+x*b7))),
           data = Hahn1, trace = TRUE, algorithm = "plinear",
           start = c(b5 = -0.05, b6 = 0.001, b7 = -0.000001)))
Try(fm4 <- nls(y ~ cbind(1, x, x^2, x^3)/(1+x*(b5+x*(b6+x*b7))),
           data = Hahn1, trace = TRUE, algorithm = "plinear",
           start = c(b5 = -0.005, b6 = 0.0001, b7 = -0.0000001)))
```

Kirby2 *Microscope line width standards*

Description

The Kirby2 data frame has 151 rows and 2 columns of data from an NIST study on scanning electron microscope line width standards.

Format

This data frame contains the following columns:

y A numeric vector of response values.

x A numeric vector of input values.

Details

These data are the result of a NIST study involving scanning electron microscope line with standards.

Source

Kirby, R., NIST (197?). Scanning electron microscope line width standards.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Kirby2)
Try(fm1 <- nls(y ~ (b1 + b2*x + b3*x**2) / (1 + b4*x + b5*x**2),
           data = Kirby2, trace = TRUE,
           start = c(b1 = 2, b2 = -0.1, b3 = 0.003,
                     b4 = -0.001, b5 = 0.00001)))
Try(fm1a <- nls(y ~ (b1 + b2*x + b3*x**2) / (1 + b4*x + b5*x**2),
           data = Kirby2, trace = TRUE, alg = "port",
           start = c(b1 = 2, b2 = -0.1, b3 = 0.003,
                     b4 = -0.001, b5 = 0.00001)))
Try(fm2 <- nls(y ~ (b1 + b2*x + b3*x**2) / (1 + b4*x + b5*x**2),
           data = Kirby2, trace = TRUE,
           start = c(b1 = 1.5, b2 = -0.15, b3 = 0.0025,
                     b4 = -0.0015, b5 = 0.00002)))
Try(fm2a <- nls(y ~ (b1 + b2*x + b3*x**2) / (1 + b4*x + b5*x**2),
             data = Kirby2, trace = TRUE, alg = "port",
             start = c(b1 = 1.5, b2 = -0.15, b3 = 0.0025,
                       b4 = -0.0015, b5 = 0.00002)))
Try(fm3 <- nls(y ~ cbind(1, x, x**2)/(1 + x*(b4 + b5*x)),
           data = Kirby2, trace = TRUE, algorithm = "plinear",
           start = c(b4 = -0.001, b5 = 0.00001)))
Try(fm4 <- nls(y ~ cbind(1, x, x**2)/(1 + x*(b4 + b5*x)),
           data = Kirby2, trace = TRUE, algorithm = "plinear",
           start = c(b4 = -0.0015, b5 = 0.00002)))
```

`Lanczos1` *Generated data*

Description

The `Lanczos1` data frame has 24 rows and 2 columns of generated data.

Format

This data frame contains the following columns:

y A numeric vector of generated responses.

x A numeric vector of generated input values

Details

These data are taken from an example discussed in Lanczos (1956). The data were generated to 14-digits of accuracy using `f(x) = 0.0951*exp(-x) + 0.8607*exp(-3*x) + 1.5576*exp(-5*x)`.

Source

Lanczos, C. (1956). Applied Analysis. Englewood Cliffs, NJ: Prentice Hall, pp. 272-280.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Lanczos1)
## plot on log scale to see the apparent number of exponential terms
plot(y ~ x, data = Lanczos1, log = "y")
## data are an exact fit so the convergence criterion fails
Try(fm1 <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos1, trace = TRUE,
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
Try(fm1a <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
                data = Lanczos1, trace = TRUE, alg = "port",
                start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6,
                          b4 = 5.5, b5 = 6.5, b6 = 7.6)))
## data are an exact fit so the convergence criterion fails
Try(fm2 <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos1, trace = TRUE,
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
Try(fm2a <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
                data = Lanczos1, trace = TRUE, alg = "port",
                start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6,
                          b4 = 4.2, b5 = 4, b6 = 6.3)))
## data are an exact fit so the convergence criterion fails
Try(fm3 <- nls(y ~ exp(outer(x,-c(b2, b4, b6))),
               data = Lanczos1, trace = TRUE, algorithm = "plinear",
               start = c(b2 = 0.3, b4 = 5.5, b6 = 7.6)))
## data are an exact fit so the convergence criterion fails
Try(fm4 <- nls(y ~ exp(outer(x,-c(b2, b4, b6))),
               data = Lanczos1, trace = TRUE, algorithm = "plinear",
               start = c(b2 = 0.7, b4 = 4.2, b6 = 6.3)))
```

`Lanczos2` *Generated data*

Description

The `Lanczos2` data frame has 24 rows and 2 columns of generated data.

Format

This data frame contains the following columns:

y A numeric vector of generated responses.

x A numeric vector of generated input values.

Details

These data are taken from an example discussed in Lanczos (1956). The data were generated to 6-digits of accuracy using `f(x) = 0.0951*exp(-x) + 0.8607*exp(-3*x) + 1.5576*exp(-5*x)`.

Source

Lanczos, C. (1956). Applied Analysis. Englewood Cliffs, NJ: Prentice Hall, pp. 272-280.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Lanczos2)
## plot log response to see the number of exponential terms
plot(y ~ x, data = Lanczos2, log = "y")
## Numerical derivatives do not produce sufficient accuracy to converge
Try(fm1 <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos2, trace = TRUE,
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
Try(fm1a <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos2, trace = TRUE, alg = "port",
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
## Numerical derivatives do not produce sufficient accuracy to converge
Try(fm2 <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos2, trace = TRUE,
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
Try(fm2a <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos2, trace = TRUE, alg = "port",
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
## Numerical derivatives do not produce sufficient accuracy to converge
Try(fm3 <- nls(y ~ exp(outer(x,-c(b2, b4, b6))),
           data = Lanczos2, trace = TRUE, algorithm = "plinear",
           start = c(b2 = 0.3, b4 = 5.5, b6 = 7.6)))
## Numerical derivatives do not produce sufficient accuracy to converge
Try(fm4 <- nls(y ~ exp(outer(x,-c(b2, b4, b6))),
           data = Lanczos2, trace = TRUE, algorithm = "plinear",
           start = c(b2 = 0.7, b4 = 4.2, b6 = 6.3)))
## Use analytic derivatives
Lanczos <- deriv(~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
                 paste("b", 1:6, sep = ""),
                 function(x, b1, b2, b3, b4, b5, b6){})
Try(fm5 <- nls(y ~ Lanczos(x, b1, b2, b3, b4, b5, b6),
           data = Lanczos2, trace = TRUE,
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
Try(fm5a <- nls(y ~ Lanczos(x, b1, b2, b3, b4, b5, b6),
           data = Lanczos2, trace = TRUE, alg = "port",
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
```

```
Try(fm6 <- nls(y ~ Lanczos(x, b1, b2, b3, b4, b5, b6),
           data = Lanczos2, trace = TRUE,
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
Try(fm6a <- nls(y ~ Lanczos(x, b1, b2, b3, b4, b5, b6),
           data = Lanczos2, trace = TRUE, alg = "port",
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
```

Lanczos3 *Generated data*

Description

The Lanczos3 data frame has 24 rows and 2 columns of generated data.

Format

This data frame contains the following columns:

y A numeric vector of generated responses.

x A numeric vector of generated input values.

Details

These data are taken from an example discussed in Lanczos (1956). The data were generated to 5-digits of accuracy using `f(x) = 0.0951*exp(-x) + 0.8607*exp(-3*x) + 1.5576*exp(-5*x)`.

Source

Lanczos, C. (1956). Applied Analysis. Englewood Cliffs, NJ: Prentice Hall, pp. 272-280.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Lanczos3)
## plot log response to see the number of exponential terms
plot(y ~ x, data = Lanczos3, log = "y")
Try(fm1 <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos3, trace = TRUE,
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
Try(fm1a <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos3, trace = TRUE, alg = "port",
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
Try(fm2 <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos3, trace = TRUE,
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
```

```
Try(fm2a <- nls(y ~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
           data = Lanczos3, trace = TRUE, alg = "port",
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
Try(fm3 <- nls(y ~ exp(outer(x,-c(b2, b4, b6))),
           data = Lanczos3, trace = TRUE, algorithm = "plinear",
           start = c(b2 = 0.3, b4 = 5.5, b6 = 7.6)))
Try(fm4 <- nls(y ~ exp(outer(x,-c(b2, b4, b6))),
           data = Lanczos3, trace = TRUE, algorithm = "plinear",
           start = c(b2 = 0.7, b4 = 4.2, b6 = 6.3)))
## Use analytic derivatives
Lanczos <- deriv(~ b1*exp(-b2*x) + b3*exp(-b4*x) + b5*exp(-b6*x),
                 paste("b", 1:6, sep = ""),
                 function(x, b1, b2, b3, b4, b5, b6){})
Try(fm5 <- nls(y ~ Lanczos(x, b1, b2, b3, b4, b5, b6),
           data = Lanczos3, trace = TRUE,
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
Try(fm5a <- nls(y ~ Lanczos(x, b1, b2, b3, b4, b5, b6),
           data = Lanczos3, trace = TRUE, alg = "port",
           start = c(b1 = 1.2, b2 = 0.3, b3 = 5.6, b4 = 5.5,
                     b5 = 6.5, b6 = 7.6)))
Try(fm6 <- nls(y ~ Lanczos(x, b1, b2, b3, b4, b5, b6),
           data = Lanczos3, trace = TRUE,
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
Try(fm6a <- nls(y ~ Lanczos(x, b1, b2, b3, b4, b5, b6),
           data = Lanczos3, trace = TRUE, alg = "port",
           start = c(b1 = 0.5, b2 = 0.7, b3 = 3.6, b4 = 4.2,
                     b5 = 4, b6 = 6.3)))
```

MGH09 — *More, Gabrow and Hillstrom example 9*

Description

The MGH09 data frame has 11 rows and 2 columns giving

Format

This data frame contains the following columns:

y A numeric vector of response values.

x A numeric vector of input values.

Details

This problem was found to be difficult for some very good algorithms. There is a local minimum at (+inf, -14.07..., -inf, -inf) with final sum of squares 0.00102734....

See More, J. J., Garbow, B. S., and Hillstrom, K. E. (1981). *Testing unconstrained optimization software.* **ACM Transactions on Mathematical Software**. 7(1): pp. 17–41.

Source

Kowalik, J.S., and M. R. Osborne, (1978). Methods for Unconstrained Optimization Problems. New York, NY: Elsevier North-Holland.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = MGH09)
## starting values for this attempt are ridiculous
Try(fm1 <- nls(y ~ b1*(x**2+x*b2) / (x**2+x*b3+b4),
           data = MGH09, trace = TRUE,
           start = c(b1 = 25, b2 = 39, b3 = 41.5, b4 = 39)))
Try(fm1a <- nls(y ~ b1*(x**2+x*b2) / (x**2+x*b3+b4),
           data = MGH09, trace = TRUE, alg = "port",
           start = c(b1 = 25, b2 = 39, b3 = 41.5, b4 = 39)))

Try(fm2 <- nls(y ~ b1*(x**2+x*b2) / (x**2+x*b3+b4),
           data = MGH09, trace = TRUE,
           start = c(b1 = 0.25, b2 = 0.39, b3 = 0.415, b4 = 0.39)))
Try(fm2a <- nls(y ~ b1*(x**2+x*b2) / (x**2+x*b3+b4),
           data = MGH09, trace = TRUE, alg = "port",
           start = c(b1 = 0.25, b2 = 0.39, b3 = 0.415, b4 = 0.39)))
Try(fm3 <- nls(y ~ cbind(x, x**2) / (x**2+x*b3+b4),
           data = MGH09, trace = TRUE, algorithm = "plinear",
           start = c(b3 = 41.5, b4 = 39)))
Try(fm4 <- nls(y ~ cbind(x, x**2) / (x**2+x*b3+b4),
           data = MGH09, trace = TRUE, algorithm = "plinear",
           start = c(b3 = 0.415, b4 = 0.39)))
```

MGH10 *More, Gabrow and Hillstrom example 10*

Description

The MGH10 data frame has 16 rows and 2 columns.

Format

This data frame contains the following columns:

y A numeric vector of response values.

x A numeric vector of input values.

Details

This problem was found to be difficult for some very good algorithms.

See More, J. J., Garbow, B. S., and Hillstrom, K. E. (1981). *Testing unconstrained optimization software.* **ACM Transactions on Mathematical Software.** 7(1): pp. 17-41.

Source

Meyer, R. R. (1970). Theoretical and computational aspects of nonlinear regression. In Nonlinear Programming, Rosen, Mangasarian and Ritter (Eds). New York, NY: Academic Press, pp. 465-486.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = MGH10)
## check plot on log scale for shape
plot(y ~ x, data = MGH10, log = "y")
## starting values for this run are ridiculous
Try(fm1 <- nls(y ~ b1 * exp(b2/(x+b3)), data = MGH10, trace = TRUE,
           start = c(b1 = 2, b2 = 400000, b3 = 25000)))
Try(fm1a <- nls(y ~ b1 * exp(b2/(x+b3)), data = MGH10,
                trace = TRUE, alg = "port",
                start = c(b1 = 2, b2 = 400000, b3 = 25000)))
Try(fm2 <- nls(y ~ b1 * exp(b2/(x+b3)), data = MGH10, trace = TRUE,
           start = c(b1 = 0.02, b2 = 4000, b3 = 250)))
Try(fm2a <- nls(y ~ b1 * exp(b2/(x+b3)), data = MGH10,
                trace = TRUE, alg = "port",
                start = c(b1 = 0.02, b2 = 4000, b3 = 250)))
Try(fm3 <- nls(y ~ exp(b2/(x+b3)), data = MGH10, trace = TRUE,
               start = c(b2 = 400000, b3 = 25000),
               algorithm = "plinear"))
Try(fm4 <- nls(y ~ exp(b2/(x+b3)), data = MGH10, trace = TRUE,
           start = c(b2 = 4000, b3 = 250),
           algorithm = "plinear"))
```

MGH17 *More, Gabrow and Hillstrom example 17*

Description

The MGH17 data frame has 33 rows and 2 columns

Format

This data frame contains the following columns:

y A numeric vector of response values.

x A numeric vector of input values.

Details

This problem was found to be difficult for some very good algorithms.

See More, J. J., Garbow, B. S., and Hillstrom, K. E. (1981). *Testing unconstrained optimization software.* **ACM Transactions on Mathematical Software.** 7(1): pp. 17-41.

Source

Osborne, M. R. (1972). Some aspects of nonlinear least squares calculations. In Numerical Methods for Nonlinear Optimization, Lootsma (Ed). New York, NY: Academic Press, pp. 171-189.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = MGH17)

## Starting values here are ridiculous
Try(fm1 <- nls(y ~ b1 + b2*exp(-x*b4) + b3*exp(-x*b5),
           data = MGH17, trace = TRUE,
           start = c(b1 = 50, b2 = 150, b3 = -100, b4 = 1, b5 = 2)))
Try(fm1a <- nls(y ~ b1 + b2*exp(-x*b4) + b3*exp(-x*b5),
           data = MGH17, trace = TRUE, alg = "port",
           start = c(b1 = 50, b2 = 150, b3 = -100, b4 = 1, b5 = 2)))

Try(fm2 <- nls(y ~ b1 + b2*exp(-x*b4) + b3*exp(-x*b5),
           data = MGH17, trace = TRUE,
           start = c(b1 = 0.5, b2 = 1.5, b3 = -1, b4 = 0.01, b5 = 0.02)))
Try(fm2a <- nls(y ~ b1 + b2*exp(-x*b4) + b3*exp(-x*b5),
           data = MGH17, trace = TRUE, alg = "port",
           start = c(b1 = 0.5, b2 = 1.5, b3 = -1, b4 = 0.01, b5 = 0.02)))

Try(fm3 <- nls(y ~ cbind(1, exp(-x*b4), exp(-x*b5)),
           data = MGH17, trace = TRUE, algorithm = "plinear",
           start = c(b4 = 1, b5 = 2)))

Try(fm4 <- nls(y ~ cbind(1, exp(-x*b4), exp(-x*b5)),
           data = MGH17, trace = TRUE, algorithm = "plinear",
           start = c(b4 = 0.01, b5 = 0.02)))
```

Misra1a *Monomolecular Absorption Data*

Description

The Misra1a data frame has 14 rows and 2 columns.

Format

This data frame contains the following columns:

y A numeric vector of volume values.

x A numeric vector of pressure values.

Details

These data are the result of a NIST study regarding dental research in monomolecular adsorption. The response variable is volume, and the predictor variable is pressure.

Source

Misra, D., NIST (1978). Dental Research Monomolecular Adsorption Study.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Misra1a)
Try(fm1 <- nls(y ~ b1*(1-exp(-b2*x)), data = Misra1a, trace = TRUE,
           start = c(b1 = 500, b2 = 0.0001) ))
Try(fm1a <- nls(y ~ b1*(1-exp(-b2*x)), data = Misra1a, trace = TRUE,
           alg = "port", start = c(b1 = 500, b2 = 0.0001) ))
Try(fm2 <- nls(y ~ b1*(1-exp(-b2*x)), data = Misra1a, trace = TRUE,
           start = c(b1 = 250, b2 = 0.0005) ))
Try(fm2a <- nls(y ~ b1*(1-exp(-b2*x)), data = Misra1a, trace = TRUE,
           alg = "port", start = c(b1 = 250, b2 = 0.0005) ))
Try(fm3 <- nls(y ~ 1-exp(-b2*x), data = Misra1a, trace = TRUE,
           start = c(b2 = 0.0001), algorithm = "plinear" ))
Try(fm4 <- nls(y ~ 1-exp(-b2*x), data = Misra1a, trace = TRUE,
           start = c(b2 = 0.0005), algorithm = "plinear" ))

## Using a self-starting model
Try(fm5 <- nls(y ~ SSasympOrig(x, Asym, lrc), data = Misra1a))
```

Misra1b *Monomolecular Absorption Data*

Description

The Misra1b data frame has 14 rows and 2 columns. It is the same data as Misra1a but a different model is fit.

Format

This data frame contains the following columns:

y A numeric vector of volume values.

x A numeric vector of pressure values.

Details

These data are the result of a NIST study regarding dental research in monomolecular adsorption. The response variable is volume, and the predictor variable is pressure.

Source

Misra, D., NIST (1978). Dental Research Monomolecular Adsorption Study.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Misra1b)
Try(fm1 <- nls(y ~ b1 * (1-(1+b2*x/2)**(-2)), data = Misra1b, trace = TRUE,
           start = c(b1 = 500, b2 = 0.0001) ))
Try(fm1a <- nls(y ~ b1 * (1-(1+b2*x/2)**(-2)), data = Misra1b, trace = TRUE,
           alg = "port", start = c(b1 = 500, b2 = 0.0001) ))
Try(fm2 <- nls(y ~ b1 * (1-(1+b2*x/2)**(-2)), data = Misra1b, trace = TRUE,
           start = c(b1 = 300, b2 = 0.0002) ))
Try(fm2a <- nls(y ~ b1 * (1-(1+b2*x/2)**(-2)), data = Misra1b, trace = TRUE,
           alg = "port", start = c(b1 = 300, b2 = 0.0002) ))
Try(fm3 <- nls(y ~ 1-(1+b2*x/2)**(-2), data = Misra1b, trace = TRUE,
           start = c(b2 = 0.0001), algorithm = "plinear" ))
Try(fm4 <- nls(y ~ 1-(1+b2*x/2)**(-2), data = Misra1b, trace = TRUE,
           start = c(b2 = 0.0005), algorithm = "plinear" ))
```

Misra1c *Monomolecular Absorption data*

Description

The `Misra1c` data frame has 14 rows and 2 columns. This is the same data as `Misra1a` but a different model is fit.

Format

This data frame contains the following columns:

y A numeric vector of volume values.

x A numeric vector of pressure values.

Details

These data are the result of a NIST study regarding dental research in monomolecular adsorption. The response variable is volume, and the predictor variable is pressure.

Source

Misra, D., NIST (1978). Dental Research Monomolecular Adsorption Study.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Misra1c)
Try(fm1 <- nls(y ~ b1*(1-(1+2*b2*x)**(-.5)), data = Misra1c, trace = TRUE,
           start = c(b1 = 500, b2 = 0.0001) ))
Try(fm1a <- nls(y ~ b1*(1-(1+2*b2*x)**(-.5)), data = Misra1c, trace = TRUE,
           alg = "port", start = c(b1 = 500, b2 = 0.0001) ))
Try(fm2 <- nls(y ~ b1*(1-(1+2*b2*x)**(-.5)), data = Misra1c, trace = TRUE,
```

```
          start = c(b1 = 600, b2 = 0.0002) ))
Try(fm2a <- nls(y ~ b1*(1-(1+2*b2*x)**(-.5)), data = Misra1c, trace = TRUE,
           alg = "port", start = c(b1 = 600, b2 = 0.0002) ))
Try(fm3 <- nls(y ~ 1-(1+2*b2*x)**(-.5), data = Misra1c, trace = TRUE,
           start = c(b2 = 0.0001), algorithm = "plinear" ))
Try(fm4 <- nls(y ~ 1-(1+2*b2*x)**(-.5), data = Misra1c, trace = TRUE,
           start = c(b2 = 0.0002), algorithm = "plinear" ))
```

Misra1d *Monomolecular Absorption data*

Description

The `Misra1d` data frame has 14 rows and 2 columns. This is the same data as `Misra1a` but a different model is fit.

Format

This data frame contains the following columns:

y A numeric vector of volume values.

x A numeric vector of pressure values.

Details

These data are the result of a NIST study regarding dental research in monomolecular adsorption. The response variable is volume, and the predictor variable is pressure.

Source

Misra, D., NIST (1978). Dental Research Monomolecular Adsorption Study.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Misra1d)
Try(fm1 <- nls(y ~ b1*b2*x*((1+b2*x)**(-1)), data = Misra1d, trace = TRUE,
           start = c(b1 = 500, b2 = 0.0001) ))
Try(fm1a <- nls(y ~ b1*b2*x*((1+b2*x)**(-1)), data = Misra1d, trace = TRUE,
           alg = "port", start = c(b1 = 500, b2 = 0.0001) ))
Try(fm2 <- nls(y ~ b1*b2*x*((1+b2*x)**(-1)), data = Misra1d, trace = TRUE,
           start = c(b1 = 450, b2 = 0.0003) ))
Try(fm2a <- nls(y ~ b1*b2*x*((1+b2*x)**(-1)), data = Misra1d, trace = TRUE,
           alg = "port", start = c(b1 = 450, b2 = 0.0003) ))
Try(fm3 <- nls(y ~ b2*x*((1+b2*x)**(-1)), data = Misra1d, trace = TRUE,
           start = c(b2 = 0.0001), algorithm = "plinear" ))
Try(fm4 <- nls(y ~ b2*x*((1+b2*x)**(-1)), data = Misra1d, trace = TRUE,
           start = c(b2 = 0.0005), algorithm = "plinear" ))
```

Nelson — *Dialectric breakdown data*

Description

The `Nelson` data frame has 128 rows and 3 columns of data from an accelerated test of dialectric breakdown.

Format

This data frame contains the following columns:

y A numeric vector of dialectric breakdown strength values.

x1 A numeric vector of time values.

x2 A numeric vector of temperature values.

Details

These data are the result of a study involving the analysis of performance degradation data from accelerated tests, published in IEEE Transactions on Reliability. The response variable is dialectric breakdown strength in kilo-volts, and the predictor variables are time in weeks and temperature in degrees Celsius.

Source

Nelson, W. (1981). Analysis of Performance-Degradation Data. IEEE Transactions on Reliability. Vol. 2, R-30, No. 2, pp. 149-155.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x1, data = Nelson, log = "y")
plot(y ~ x2, data = Nelson, log = "y")
coplot(y ~ x1 | x2, data = Nelson)
coplot(y ~ x2 | x1, data = Nelson)

Try(fm1 <- nls(log(y) ~ b1 - b2*x1 * exp(-b3*x2), data = Nelson,
           start = c(b1 = 2, b2 = 0.0001, b3 = -0.01), trace = TRUE))
Try(fm1a <- nls(log(y) ~ b1 - b2*x1 * exp(-b3*x2), data = Nelson,
            trace = TRUE, alg = "port",
            start = c(b1 = 2, b2 = 0.0001, b3 = -0.01)))

Try(fm2 <- nls(log(y) ~ b1 - b2*x1 * exp(-b3*x2), data = Nelson,
           start = c(b1 = 2.5, b2 = 0.000000005, b3 = -0.05), trace = TRUE))
Try(fm2 <- nls(log(y) ~ b1 - b2*x1 * exp(-b3*x2), data = Nelson,
           trace = TRUE, alg = "port",
           start = c(b1 = 2.5, b2 = 0.000000005, b3 = -0.05)))

Try(fm3 <- nls(log(y) ~ cbind(1, -x1 * exp(-b3*x2)), data = Nelson,
```

```
          start = c(b3 = -0.01), trace = TRUE, algorithm = "plinear"))

Try(fm4 <- nls(log(y) ~ cbind(1, -x1 * exp(-b3*x2)), data = Nelson,
          start = c(b3 = -0.05), trace = TRUE, algorithm = "plinear"))
```

Ratkowsky2 *Pasture yield data*

Description

The Ratkowsky2 data frame has 9 rows and 2 columns.

Format

This data frame contains the following columns:

y A numeric vector of pasture yields.

x A numeric vector of growing times.

Details

This model and data are an example of fitting sigmoidal growth curves taken from Ratkowsky (1983). The response variable is pasture yield, and the predictor variable is growing time.

Source

Ratkowsky, D.A. (1983). Nonlinear Regression Modeling. New York, NY: Marcel Dekker, pp. 61 and 88.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Ratkowsky2)

Try(fm1 <- nls(y ~ b1 / (1+exp(b2-b3*x)), data = Ratkowsky2, trace = TRUE,
          start = c(b1 = 100, b2 = 1, b3 = 0.1)))
Try(fm1a <- nls(y ~ b1 / (1+exp(b2-b3*x)), data = Ratkowsky2,
           trace = TRUE, alg = "port",
           start = c(b1 = 100, b2 = 1, b3 = 0.1)))
Try(fm2 <- nls(y ~ b1 / (1+exp(b2-b3*x)), data = Ratkowsky2, trace = TRUE,
          start = c(b1 = 75, b2 = 2.5, b3 = 0.07)))
Try(fm2a <- nls(y ~ b1 / (1+exp(b2-b3*x)), data = Ratkowsky2,
           trace = TRUE, alg = "port",
           start = c(b1 = 75, b2 = 2.5, b3 = 0.07)))
Try(fm3 <- nls(y ~ 1 / (1+exp(b2-b3*x)), data = Ratkowsky2, trace = TRUE,
          start = c(b2 = 1, b3 = 0.1), alg = "plinear"))
Try(fm4 <- nls(y ~ 1 / (1+exp(b2-b3*x)), data = Ratkowsky2, trace = TRUE,
          start = c(b2 = 2.5, b3 = 0.07), alg = "plinear"))
```

```
## Using a self-starting model
Try(fm5 <- nls(y ~ SSlogis(x, Asym, xmid, scal), data = Ratkowsky2))
summary(fm5)
```

Ratkowsky3 *Onion growth data*

Description

The Ratkowsky3 data frame has 15 rows and 2 columns.

Format

This data frame contains the following columns:

y A numeric vector of dry weights of onion bulbs and tops.

x A numeric vector of growing times.

Details

This model and data are an example of fitting sigmoidal growth curves taken from Ratkowsky (1983). The response variable is the dry weight of onion bulbs and tops, and the predictor variable is growing time.

Source

Ratkowsky, D.A. (1983). Nonlinear Regression Modeling. New York, NY: Marcel Dekker, pp. 62 and 88.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Ratkowsky3)

## causes NA/NaN/Inf error
Try(fm1 <- nls(y ~ b1 / ((1+exp(b2-b3*x))**(1/b4)), data = Ratkowsky3,
           start = c(b1 = 100, b2 = 10, b3 = 1, b4 = 1),
           trace = TRUE))
Try(fm1a <- nls(y ~ b1 / ((1+exp(b2-b3*x))**(1/b4)), data = Ratkowsky3,
           start = c(b1 = 100, b2 = 10, b3 = 1, b4 = 1),
           alg = "port", trace = TRUE))

Try(fm2 <- nls(y ~ b1 / ((1+exp(b2-b3*x))**(1/b4)), data = Ratkowsky3,
           start = c(b1 = 700, b2 = 5, b3 = 0.75, b4 = 1.3),
           trace = TRUE))
Try(fm2a <- nls(y ~ b1 / ((1+exp(b2-b3*x))**(1/b4)), data = Ratkowsky3,
           start = c(b1 = 700, b2 = 5, b3 = 0.75, b4 = 1.3),
           alg = "port", trace = TRUE))
```

```
Try(fm3 <- nls(y ~ 1 / ((1+exp(b2-b3*x))**(1/b4)), data = Ratkowsky3,
           start = c(b2 = 10, b3 = 1, b4 = 1), algorithm = "plinear",
           trace = TRUE))
Try(fm4 <- nls(y ~ 1 / ((1+exp(b2-b3*x))**(1/b4)), data = Ratkowsky3,
           start = c(b2 = 5, b3 = 0.75, b4 = 1.3), algorithm = "plinear",
           trace = TRUE))
```

Roszman1 *Quantum defects in iodine*

Description

The Roszman1 data frame has 25 rows and 2 columns of data on the number of quantum defects in iodine atoms at different energy states.

Format

This data frame contains the following columns:

y A numeric vector of number of quantum defects.

x A numeric vector of the excited energy state.

Details

These data are the result of a NIST study involving quantum defects in iodine atoms. The response variable is the number of quantum defects, and the predictor variable is the excited energy state. The argument to the ARCTAN function is in radians.

Source

Roszman, L., NIST (19??). Quantum Defects for Sulfur I Atom.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Roszman1)
Try(fm1 <- nls(y ~ b1 - b2*x - atan(b3/(x-b4))/pi, data = Roszman1,
           start = c(b1 = 0.1, b2 = -0.00001, b3 = 1000, b4 = -100),
           trace = TRUE))
Try(fm1a <- nls(y ~ b1 - b2*x - atan(b3/(x-b4))/pi, data = Roszman1,
           start = c(b1 = 0.1, b2 = -0.00001, b3 = 1000, b4 = -100),
           alg = "port", trace = TRUE))
Try(fm2 <- nls(y ~ b1 - b2*x - atan(b3/(x-b4))/pi, data = Roszman1,
           start = c(b1 = 0.2, b2 = -0.0000015, b3 = 1200, b4 = -150),
           trace = TRUE))
Try(fm2a <- nls(y ~ b1 - b2*x - atan(b3/(x-b4))/pi, data = Roszman1,
           start = c(b1 = 0.2, b2 = -0.0000015, b3 = 1200, b4 = -150),
           alg = "port", trace = TRUE))
```

Thurber *Electron mobility data*

Description

The Thurber data frame has 37 rows and 2 columns.

Format

This data frame contains the following columns:

y A numeric vector of electron mobility values.

x A numeric vector of logs of electron density values.

Details

These data are the result of a NIST study involving semiconductor electron mobility. The response variable is a measure of electron mobility, and the predictor variable is the natural log of the density.

Source

Thurber, R., NIST (197?). Semiconductor electron mobility modeling.

Examples

```
Try <- function(expr) if (!inherits(val <- try(expr), "try-error")) val
plot(y ~ x, data = Thurber)
Try(fm1 <- nls(y ~ (b1+x*(b2+x*(b3+b4*x))) / (1+x*(b5+x*(b6+x*b7))),
           data = Thurber, trace = TRUE,
           start = c(b1 = 1000, b2 = 1000, b3 = 400, b4 = 40,
                     b5 = 0.7, b6 = 0.3, b7 = 0.03)))
Try(fm1a <- nls(y ~ (b1+x*(b2+x*(b3+b4*x))) / (1+x*(b5+x*(b6+x*b7))),
           data = Thurber, trace = TRUE, alg = "port",
           start = c(b1 = 1000, b2 = 1000, b3 = 400, b4 = 40,
                     b5 = 0.7, b6 = 0.3, b7 = 0.03)))
Try(fm2 <- nls(y ~ (b1+x*(b2+x*(b3+b4*x))) / (1+x*(b5+x*(b6+x*b7))),
           data = Thurber, trace = TRUE,
           start = c(b1 = 1300, b2 = 1500, b3 = 500, b4 = 75,
                     b5 = 1, b6 = 0.4, b7 = 0.05)))
Try(fm2a <- nls(y ~ (b1+x*(b2+x*(b3+b4*x))) / (1+x*(b5+x*(b6+x*b7))),
           data = Thurber, trace = TRUE, alg = "port",
           start = c(b1 = 1300, b2 = 1500, b3 = 500, b4 = 75,
                     b5 = 1, b6 = 0.4, b7 = 0.05)))
Try(fm3 <- nls(y ~ outer(x, 0:3, "^")/(1+x*(b5+x*(b6+x*b7))),
           data = Thurber, trace = TRUE,
           start = c(b5 = 0.7, b6 = 0.3, b7 = 0.03), alg = "plinear"))
Try(fm4 <- nls(y ~ outer(x, 0:3, "^")/(1+x*(b5+x*(b6+x*b7))),
           data = Thurber, trace = TRUE,
           start = c(b5 = 1, b6 = 0.4, b7 = 0.05), alg = "plinear"))
```

Index