Package ‘RobPer’

March 2, 2015

Type Package

Title Robust Periodogram and Periodicity Detection Methods

Version 1.2.1

Date 2015-03-02

Author Anita M. Thieler, Jonathan Rathjens and Roland Fried, with contributions from Brenton R. Clarke, Uwe Ligges, Matias Salibian-Barrera, Gert Willems and Victor Yohai

Maintainer Jonathan Rathjens <jonathan.rathjens@tu-dortmund.de>

Depends robustbase, quantreg, splines, BB, rgenoud

Description Calculates periodograms based on (robustly) fitting periodic functions to light curves (irregularly observed time series, possibly with measurement accuracies, occurring in astroparticle physics). Three main functions are included: RobPer calculates the periodogram. Outlying periodogram bars (indicating a period) can be detected with betaCvMfit. Artificial light curves can be generated using the function tsgen. For more details see Thieler, Fried and Rathjens (2013).

License GPL-2 | GPL-3

Copyright The data in star_groj0422.32 have been kindly provided by the NASA. The data in Mrk421 and Mrk501 have been kindly provided by the Deutsches Elektronen-Synchrotron. See the respective documentation.

NeedsCompilation no

Repository CRAN

Date/Publication 2015-03-02 09:40:20

R topics documented:

RobPer-package .................................................. 2
betaCvMfit .................................................... 4
disturber ....................................................... 7
FastS ............................................................ 8
FastTau ......................................................... 10
le_noise ......................................................... 11
Mrk421 .......................................................... 13
RobPer-package

Description

Calculates periodograms based on (robustly) fitting periodic functions to light curves and other irregularly observed time series and detects high periodogram bars.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>RobPer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.2.1</td>
</tr>
<tr>
<td>Date:</td>
<td>2015-03-02</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-2</td>
</tr>
</tbody>
</table>

Light curves occur in astroparticle physics and are irregularly sampled times series \((t_i, y_i)_{i=1,...,n}\) or \((t_i, y_i, s_i)_{i=1,...,n}\) consisting of unequally spaced observation times \(t_1, \ldots, t_n\), observed values \(y_1, \ldots, y_n\) and possibly measurement accuracies \(s_1, \ldots, s_n\). The pattern of the observation times \(t_i\) may be periodic with sampling period \(p_s\). The observed values \(y_i\) may possibly contain a periodic fluctuation \(y_{f,i}\) with fluctuation period \(p_f\). One is interested in finding \(p_f\). The measurement accuracies \(s_i\) give information about how precise the \(y_i\) were measured. They can be interpreted as estimates for the standard deviations of the observed values. For more details see Thieler et al. (2013) or Thieler, Fried and Rathjens (2013).

This package includes three main functions: RobPer calculates the periodogram, possibly taking into account measurement accuracies. With betaCvMfit, outlying periodogram bars (indicating a period) can be detected. This function bases on robustly fitting a distribution using Cramér-von-Mises (CvM) distance minimization (see also Clarke, McKinnon and Riley 2012). The function tsgen can be used to generate artificial light curves. For more details about the implementation see Thieler, Fried and Rathjens (2013).

The financial support of the DFG (SFB 876 "Providing Information by Resource-Constrained Data Analysis", project C3, and GK 1032 "Statistische Modellbildung") is gratefully acknowledged. We thank the ITMC at TU Dortmund University for providing computer resources on LiDO.

Author(s)

Anita M. Thieler, Jonathan Rathjens and Roland Fried, with contributions from Brenton R. Clarke (see betaCvMfit), Matias Salibian-Barrera, Gert Willems and Victor Yohai (see FastS and FastTau) and Uwe Ligges (see TK95).

Maintainer: Jonathan Rathjens <jonathan.rathjens@tu-dortmund.de>

References


Examples

# Generate a disturbed light curve:
set.seed(22)
lightcurve <- tsgen(ttype="sine", ytype="peak", pf=7, redpart=0.1, s.outlier.fraction=0.1, interval=TRUE, npoints=200, ncycles=25, ps=20, SNR=3, alpha=0)

# Plotting the light curve (vertical bars show measurement accuracies)
plot(lightcurve[,1], lightcurve[,2], pch=16, cex=0.5, xlab="t", ylab="y",
main="a Light Curve")
rect(lightcurve[,1], lightcurve[,2]+lightcurve[,3], lightcurve[,1],
lightcurve[,2]-lightcurve[,3])

# The lightcurve has a period of 7:
plot(lightcurve[,1]%7, lightcurve[,2], pch=16, cex=0.5, xlab="t", ylab="y",
main="Phase Diagram of a Light Curve")
rect(lightcurve[,1]%7, lightcurve[,2]+lightcurve[,3], lightcurve[,1]%7,
lightcurve[,2]-lightcurve[,3])

# Calculate a periodogram of a light curve:
PP <- RobPer(lightcurve, model="splines", regression="huber", weighting=FALSE, var1=FALSE, periods=1:50)
# Searching for extremely high periodogram bars:
betavalues <- betaCvmfit(PP)
crit.val <- qbeta(0.95)^(1/50), shape1=betavalues[1], shape2=betavalues[2])

hist(PP, breaks=20, freq=FALSE, ylim=c(0,100), xlim=c(0,0.08), col=8, main ="")
betafun <- function(x) dbeta(x, shape1=betavalues[1], shape2=betavalues[2])
curve(betafun, add=TRUE, lwd=2)
abline(v=crit.val, lwd=2)

# alternatives for fitting beta distributions:
# method of moments:
par.mom <- betaCvmfit(PP, rob=FALSE, Cvm=FALSE)
myf.mom <- function(x) dbeta(x, shape1=par.mom[1], shape2=par.mom[2])
curve(myf.mom, add=TRUE, lwd=2, col="red")
crit.mom <- qbeta(0.95)^(1/50), shape1=par.mom[1], shape2=par.mom[2])
abline(v=crit.mom, lwd=2, col="red")

# robust method of moments
par.rob <- betaCvmfit(PP, rob=TRUE, Cvm=FALSE)
myf.rob <- function(x) dbeta(x, shape1=par.rob[1], shape2=par.rob[2])
curve(myf.rob, add=TRUE, lwd=2, col="blue")
crit.rob <- qbeta(0.95)^(1/50), shape1=par.rob[1], shape2=par.rob[2])
abline(v=crit.rob, lwd=2, col="blue")

legend("topright", fill=c("black","red","blue"),
       legend=c("CvM", "moments", "robust moments"), bg="white")
box()

# Detect fluctuation period:
plot(1:50, PP, xlab="Trial Period", ylab="Periodogram", type="l",
     main="Periodogram fitting periodic splines using M-regression (Huber function)"
abline(h=crit.val, lwd=2)
text(c(7,14), PP[c(7,14)], c(7,14), adj=1, pos=4)
axis(1, at=7, labels=expression(pp[7,14]))

# Comparison with non-robust periodogram
# (see package vignette, section 5.1 for further graphical analysis)
PP2 <- RobPer(lightcurve, model="splines", regression="L2",
               weighting=False, var1=False, periods=1:50)
betavalues2 <- betaCvmfit(PP2)
crit.val2 <- qbeta((0.95)^(1/50), shape1=betavalues2[1], shape2=betavalues2[2])

plot(1:50, PP2, xlab="Trial Period", ylab="Periodogram", type="l",
     main="Periodogram fitting periodic splines using L2-regression"
abline(h=crit.val2, lwd=2)
betaCvMfit

Description

Robustly fits a Beta distribution to data using Cramér-von-Mises (CvM) distance minimization.

Usage

betaCvMfit(data, CvM = TRUE, rob = TRUE)

Arguments

data numeric vector: The sample, a Beta distribution is fitted to.

CvM logical: If FALSE the Cramér-von-Mises-distance is not minimized, but only moment estimates for the parameters of the Beta distribution are returned (see Details).

rob logical: If TRUE, mean and standard deviation are replaced by median and MAD when calculating moment estimates for the parameters of the Beta distribution (see Details).

Details

betaCvMfit fits a Beta distribution to data by minimizing the Cramér-von-Mises distance. Moment estimates of the parameters of the Beta distribution, clipped to positive values, are used as starting values for the optimization process. They are calculated using

\[
\hat{a} = -\bar{x} \cdot \left( -\bar{x} + \frac{\bar{x}^2 + \hat{s}^2}{\hat{s}^2} \right),
\]

\[
\hat{b} = \frac{\hat{a} - \hat{a}\bar{x}}{\bar{x}}.
\]

These clipped moment estimates can be returned instead of CvM-fitted parameters setting CvM = FALSE. The Cramér-von-Mises distance is defined as (see Clarke, McKinnon and Riley 2012)

\[
\frac{1}{n} \sum_{i=1}^{n} \left( F(u_{(i)}) - \frac{i - 0.5}{n} \right)^2 + \frac{1}{12n^2},
\]

where \(u_{(1)}, \ldots, u_{(n)}\) is the ordered sample and \(F\) the distribution function of Beta(\(a, b\)).

Value

numeric vector: Estimates for the Parameters \(a, b\) of a Beta(\(a, b\)) distribution with mean \(a/(a + b)\).

Note

Adapted from R-Code from Brenton R. Clarke to fit a Gamma distribution (see Clarke, McKinnon and Riley 2012) using Cramér-von-Mises distance minimization. Used in Thieler et al. (2013). See also Thieler, Fried and Rathjens (2013).

Author(s)

Anita M. Thieler, with contributions from Brenton R. Clarke.
References


See Also

See RobPer-package for an example applying betaCvMfit to detect valid periods in a periodogram.

Examples

```r
# data:
set.seed(12)
PP <- c(rbeta(45, shape1=4, shape2=15), runif(5, min=0.8, max=1))
hist(PP, freq=FALSE, breaks=30, ylim=c(0,7), xlab="Periodogram bar")

# true parameters:
myf.true <- function(x) dbeta(x, shape1=4, shape2=15)
curve(myf.true, add=TRUE, lwd=2)

# method of moments:
par.mom <- betaCvMfit(PP, rob=FALSE, cvm=FALSE)
myf.mom <- function(x) dbeta(x, shape1=par.mom[1], shape2=par.mom[2])
curve(myf.mom, add=TRUE, lwd=2, col="red")

# robust method of moments
par.rob <- betaCvMfit(PP, rob=TRUE, cvm=FALSE)
myf.rob <- function(x) dbeta(x, shape1=par.rob[1], shape2=par.rob[2])
curve(myf.rob, add=TRUE, lwd=2, col="blue")

# CvM distance minimization
par.CvM <- betaCvMfit(PP, rob=TRUE, cvm=TRUE)
myf.CvM <- function(x) dbeta(x, shape1=par.CvM[1], shape2=par.CvM[2])
curve(myf.CvM, add=TRUE, lwd=2, col="green")

# Searching for outliers...
abline(v=qbeta(0.95)^(1/50), shape1=par.CvM[1], shape2=par.CvM[2]), col="green")

legend("topright", fill=c("black", "green", "blue", "red"),
        legend=c("true", "CvM", "robust moments", "moments"))
box()
```
Disturber

Disturbing light curve data

Description

Disturbs a light curve replacing measurement accuracies by outliers and/or observed values by atypical values. See RobPer-package for more information about light curves.

Usage

disturber(tt, y, s, ps, s.outlier.fraction = 0, interval)

Arguments

- **tt**: numeric vector: Observation times \(t_1, \ldots, t_n\) (see Details).
- **y**: numeric vector: Observed values \(y_1, \ldots, y_n\) (see Details).
- **s**: numeric vector: Measurement accuracies \(s_1, \ldots, s_n\) (see Details).
- **ps**: positive value: Sampling period \(p_s\). Indirectly defines the length of the time interval, in which observed values \(y_i\) are replaced by atypical values (see Details).
- **s.outlier.fraction**: numeric value in \([0,1]\): Defines the proportion of measurement accuracies that is replaced by outliers (see Details). A value of 0 means that no measurement accuracy is replaced by an outlier.
- **interval**: logical: If TRUE, the observed values belonging to a random time interval of length \(3p_s\) are replaced by atypical values (see Details). If TRUE and the light curve is shorter than \(3p_s\), the function will stop with an error message.

Details

This function disturbs the light curve \((t_i, y_i, s_i)_{i=1,\ldots,n}\) given. It randomly chooses a proportion of \(s.outlier.fraction\) measurement accuracies \(s_i\) and replaces them by 0.5 \(\min(s_1, \ldots, s_n)\). In case of interval=TRUE a time interval \([t_{\text{start}}, t_{\text{start}} + 3p_s]\) within the interval \([t_1, t_n]\) is randomly chosen and all observed values belonging to this time interval are replaced by a peak function:

\[
y_{i\text{changed}} = 6y_0.9 \frac{d_N(t_{\text{start}}+1.5p_s, p_s^2)(t_i)}{d_N(0,p_s^2)(0)} \quad \forall \ i : t_i \in [t_{\text{start}}, t_{\text{start}} + 3p_s],
\]

where \(d_N(a,b^2)(x)\) denotes the density of a normal distribution with mean \(a\) and variance \(b^2\) at \(x\).

In case of \(s.outlier.fraction=0\) and interval=FALSE, \(y\) and \(s\) are returned unchanged.

Value

- **y**: numeric vector: New \(y_i\)-values, partly different from the old ones if interval=TRUE (see Details).
- **s**: numeric vector: New \(s_i\)-values, partly different from the old ones if \(s.outlier.fraction>0\) (see Details).
Note

A former version of this function is used in Thieler et al. (2013). See also Thieler, Fried and Rathjens (2013).

Author(s)

Anita M. Thieler

References


See Also

Applied in tsgen (see there for example).

FastS

S-Regression using the Fast-S-Algorithm

Description

Performs S-Regression using the Fast-S-Algorithm.

Usage

```
FastS(x, y, Scontrol=list(int = FALSE, N = 100, kk = 2, tt = 5, b= .5, cc = 1.547, seed=NULL), beta_gamma)
```

Arguments

- `x`: numeric \((n \times p)\)-matrix: Designmatrix.
- `y`: numeric vector: \(n\) observations.
- `Scontrol`: list of length seven: control parameters (see Details).
- `beta_gamma`: numeric vector: Specifies one parameter candidate of length \(p\) (see Details).
Details

The Fast-S-Algorithm to efficiently perform S-Regression was published by Salibian-Barrera and Yohai (2006). It bases on starting with a set of \( n \) parameter candidates, locally optimizing them, but only with \( kk \) iterations, optimizing the \( tt \) best candidates to convergence and then choosing the best parameter candidate. The rho-function used is the biweight function with tuning parameter \( cc \), the value \( b \) is set to the expected value of the rho-function applied to the residuals. The default \( cc=1.547 \) and \( b=.5 \) is chosen following Rousseeuw and Yohai (1984) to obtain an approximative breakdown point of 0.5. When setting int to TRUE, this adds an intercept column to the design matrix. For more details see Salibian-Barrera and Yohai (2006) or Thieler, Fried and Rathjens (2013).

The R-function FastS used in RobPer is a slightly changed version of the R-code published in Salibian-Barrera and Yohai (2006). It was changed in order to work more efficiently, especially when fitting step functions, and to specify one parameter candidate in advance. For details see Thieler, Fried and Rathjens (2013).

Value

- \texttt{beta} numeric vector: Fitted parameter vector.
- \texttt{scale} numeric: Value of the objective function

Author(s)

Matias Salibian-Barrera and Victor Yohai, modified by Anita M. Thieler

References


See Also

Applied in \texttt{RobPer}. See \texttt{FastTau} for example.
FastTau

**Description**

Performs tau-Regression using the Fast-tau-Algorithm.

**Usage**

```
FastTau(x, y, taucontrol = list(N = 500, kk = 2, tt = 5, rr = 2, approximate = 0),
        beta_gamma)
```

**Arguments**

- `x` numeric \((n \times p)\)-matrix: Designmatrix.
- `y` numeric vector: \(n\) observations.
- `taucontrol` list of four integer and one logical value: Control parameters (see Details).
- `beta_gamma` numeric vector: Specifies one parameter candidate of length \(p\) (see Details).

**Details**

The Fast-tau-Algorithm to efficiently perform tau-Regression was published by Salibian-Barrera, Willems and Zamar (2008). It bases on starting with a set of \(N\) parameter candidates, locally optimizing them using \(kk\) iterations, then optimizing the \(tt\) best candidates to convergence and finally choosing the best parameter candidate. Since calculation of the objective value is computationally expensive, it is possible to approximate it with \(rr\) iteration steps when choosing `approximate=TRUE`. For more details see Salibian-Barrera, Willems and Zamar (2008).

The R-function `FastTau` used in RobPer is a slightly changed version of the R-code published in Salibian-Barrera, Willems and Zamar (2008). It was changed in order to work more efficiently, especially when fitting step functions, and to specify one parameter candidate in advance. For details see Thieler, Fried and Rathjens (2013).

**Value**

- `beta` numeric vector: Fitted parameter vector.
- `scale` numeric: Value of the objective function

**Author(s)**

Matias Salibian-Barrera and Gert Willems, modified by Anita M. Thieler
References


See Also

Applied in RobPer.

Examples

```r
set.seed(22)
# Generate a disturbed light curve
lightcurve <- tsgen(ttype="unif",ytype="sine", pf=7, redpart=0.1, interval=TRUE,
   npoints=100, ncycles=10, ps=7, SNR=4, alpha=0)
tt <- lightcurve[,1]
y <- lightcurve[,2]
s <- rep(1,100) # unweighted regression

plot(tt, y, type="l", main="Fitting a sine to a disturbed light curve")

# Fit the true model (a sine of period 7)... designmatrix:
X <- Xgen(tt, n=100, s, pp=7, design="sine")
# Robust tau-fit:
beta_FastTau <- FastTau(X, y)$beta
# Robust S-fit:
beta_FastS <- FastS(X, y)$beta
# Least squares fit:
beta_lm <- lm(y~0*X)$coeff

# Plot:
sin7_fun <- function(t, beta){beta[1]+beta[2]*sin(t*2*pi/7)+beta[3]*cos(t*2*pi/7)}
sin_FastTau <- function(t) sin7_fun(t, beta_FastTau)
sin_FastS <- function(t) sin7_fun(t, beta_FastS)
sin_lm <- function(t) sin7_fun(t, beta_lm)
curve(sin_FastTau, col="green", add=TRUE)
curve(sin_FastS, col="blue", add=TRUE, lty=2)
curve(sin_lm, col="red", add=TRUE)

legend("topleft", fill=c("black", "red", "green", "blue"),
   legend=c("Light Curve (disturbed)", "Least Squares Fit", "FastTau Fit", "FastS Fit"))
```

**lc_noise**

Noise and measurement accuracy generator for light curves
Description

Generates measurement accuracies, a white noise component depending on them and a second (possibly power law, i.e. red) noise component which does not depend on the measurement accuracies. For more details see tsgen or Thieler, Fried and Rathjens (2013). See RobPer-package for more information about light curves.

Usage

lc_noise(tt, sig, SNR, redpart, alpha = 1.5)

Arguments

- **tt**: numeric vector: Observation times given.
- **sig**: numeric vector of same length as tt: A given signal to which the noise will be added.
- **SNR**: positive number: Defines the relation between signal and noise (see tsgen for Details).
- **redpart**: numeric value in [0,1]: Proportion of the power law noise in noise components (see tsgen for Details).
- **alpha**: numeric value: Power law index for the power law noise component (see tsgen for Details).

Value

- **y**: numeric vector: Observed values: signal + noise.
- **s**: numeric vector: Measurement accuracies related to the white noise component.

Note

A former version of this function is used in Thieler et al. (2013).

Author(s)

Anita M. Thieler and Jonathan Rathjens

References


See Also

Applied in tsgen (see there for an example), applies TK95_uneq.
Data: Light curve from Mrk 421

Description

Gamma ray light curve from Markarian 421.

Usage

Mrk421

Format

A data frame of three variables, with a time series of length 655 appropriate to RobPer.

Details

The data in Mrk421 and Mrk501 have been collected from various original sources, combined, and published by Tłuczykont et al. (2010) of the Deutsches Elektronen-Synchrotron, available from http://astro.desy.de/gamma_astronomy/magic/projects/light_curve_archive/index_eng.html.

Their sources are data from the experiments:

Whipple (Kerrick et al. 1995; Schubnell et al. 1996; Buckley et al. 1996; Maraschi et al. 1999)


CAT (Piron 2000; Piron et al. 2001)

HESS (Aharonian et al. 2005, 2006)

MAGIC (Albert et al. 2008; Donnarumma et al. 2009)

VERITAS (Rebillot et al. 2006; Donnarumma et al. 2009)

Note

See Vignette Section 5.3 for example.

Source

Data kindly provided by the Deutsches Elektronen-Synchrotron, Gamma Astronomy group (see Details).
References


Their data sources:


Aharonian, F., Akhperjanian, A., Beilicke, M., et al. (2002): Variations of the TeV energy spectrum at different flux levels of Mkn 421 observed with the HEGRA system of Cherenkov telescopes. Astronomy & Astrophysics, 393(1), 89


Description

Gamma ray light curve from Markarian 501.

Usage

Mrk501

Format

A data frame of three variables, with a time series of length 210 appropriate to RobPer.

Details


Their sources are data from the experiments:
Whipple (Kerrick et al. 1995; Schubnell et al. 1996; Buckley et al. 1996; Maraschi et al. 1999)
CAT (Piron 2000; Piron et al. 2001)
HESS (Aharonian et al. 2005, 2006)
MAGIC (Albert et al. 2008; Donnarumma et al. 2009)
VERITAS (Rebillot et al. 2006; Donnarumma et al. 2009)

Note

See Vignette Section 5.3 for example.
Source

Data kindly provided by the Deutsches Elektronen-Synchrotron, Gamma Astronomy group (see Details).

References


Their data sources:


Aharonian, F., Akhperjanian, A., Beilicke, M., et al. (2002): Variations of the TeV energy spectrum at different flux levels of Mkn 421 observed with the HEGRA system of Cherenkov telescopes. Astronomy & Astrophysics, 393(1), 89


---

**RobPer**

Periodogram based on (robustly) fitting a periodic function to a light curve

**Description**

Calculates a periodogram by fitting a periodic function to a light curve, using a possibly robust regression technique and possibly taking into account measurement accuracies. See RobPer-package for more information about light curves. For a lot of more details see Thieler, Fried and Rathjens (2013) and Thieler et al. (2013).

**Usage**

```r
RobPer(ts, weighting, periods, regression, model, steps = 10, tol = 1e-03,
       var1 = weighting, genoudcontrol = list(pop.size = 50, max.generations = 50,
                                              wait.generations = 5), LTSopt = TRUE,
       taucontrol = list(N = 100, kk = 2, tt = 5, rr = 2, approximate = FALSE),
       Scontrol = list(N = ifelse(weighting, 200, 50), kk = 2, tt = 5, b = .5, cc = 1.547,
                      seed = NULL) )
```

**Arguments**

- `ts` : dataframe or matrix with three (or two) numeric columns containing the light curve to be analyzed: observation times (first column), observed values (second column), measurement accuracies (thirs column). If it is intended to calculate the periodogram of a time series without measurement accuracies (weighting=FALSE), the third column may be omitted.
- `weighting` : logical: Should measurement accuracies be taken into account performing weighted regression?
periods vector of positive numeric values: Trial periods.

regression character string specifying the regression method used: Possible choices are "L2" (least squares regression using the R-function lm, package stats), "L1" (least absolute deviation regression, using the R-function rq, package quantreg), "LTS" (least trimmed squares regression, using the R-function ltsReg, package robustbase), "huber" (M-regression using the Huber function), "bisquare" (M-regression using the bisquare function), "S" (S-regression using adapted code from Salibian-Barrera and Yohai 2006, see FastS), "tau" (tau-regression using adapted code from Salibian-Barrera, Willems and Zamar 2008, see FastTau).

model character string specifying the periodic function fitted to the light curve: Possible choices are "step" (periodic step function), "2step" (two overlapping periodic step functions, see Details), "sine" (sine function), "fourier(2)" and "fourier(3)" (Fourier series of second or third degree), "splines" (periodic spline function with four B-splines per cycle, generated using spline.des, package splines).

steps integer value: Number of steps per cycle for the periodic step function(s).

tol (small) positive number: Precision for convergence criteria. Used in case of regression="huber" or "bisquare" or if regression="LTS" and LTSopt=TRUE.

var1 logical: Should variance estimate be set to 1 in case of weighted M-regression?

genoudcontrol list of three integers pop.size, max.generations, wait.generations: Control parameters for the R-function genoud, package rgenoud, see Details and Mebane Jr. and Sekhon (2011). Used in case of regression="bisquare" or if regression="LTS" and LTSopt=TRUE.

LTSopt logical: In case of LTS-regression, should regression result of ltsReg be optimized using the R-function genoud, package rgenoud?

taucontrol list of four integer values N, kk, tt, rr and one logical approximate: Control parameters for the R-function FastTau. For more details see FastTau and Salibian-Barrera, Willems and Zamar (2008).

Scontrol list of three integers N, kk and tt, two positive numbers b and cc and another integer seed: Control parameters for the R-function FastS. For more details see FastS and Salibian-Barrera and Yohai (2006). Please notice that the further Scontrol entry int expected by FastS is automatically set to FALSE in order to let RobPer work properly.

Details

For each trial period, a periodic function (defined by model) is fitted to the light curve using regression technique regression. The periodogram bar is the coefficient of determination. In case of model="2step", two different step functions with opposed jumping times are fitted separately and the periodogram bar is the mean of both coefficients of determination. For a lot of more details see Thieler, Fried and Rathjens (2013) and Thieler et al. (2013).

Value

numeric vector: Periodogram bars related to the trial periods.
Performing weighting = FALSE, regression="L2", model="sine" on a equidistantly sampled time series is equivalent to calculating the standard periodogram of Fourier analysis, see Example.

Performing regression="L2", model="sine" is equivalent to calculating a Generalized Lomb-Scargle periodogram (see Zechmeister and Kürster 2009).

Performing regression="L2", model="step" is equivalent to calculating an Epoch Folding (Leahy et al. 1983) or Analysis of Variance (Schwarzenberg-Czerny 1989) periodogram.

Performing regression="L2", model="step" is equivalent to calculating a Phase Dispersion Minimization periodogram (Stellingwerf 1978).

A former version of this function is used in Thieler et al. (2013). For more equivalences see there.

Author(s)

Anita M. Thieler, Jonathan Rathjens and Roland Fried

References


See Also

Applies FastS and FastTau, Xgen, examples in RobPer-package and TK95_uneq.
Examples

# For more examples see RobPer-package and TK95_uneq!

# Example to show the equivalence between the periodogram from Fourier analysis
# and the Lomb-Scargle periodogram in case of equidistant sampling and equal weighting:
set.seed(7)

n <- 120

# equidistant time series:
zr <- tsgen(ttype="equi", ytype="const", pf=1, redpart= 0, s.outlier.fraction=0.2,
           interval=FALSE, npoints=n, ncycles=n, ps=1, SNR=1, alpha=1.5)

# periodogram of Fourier analysis
PP_konv <- spec.pgram(zr[,2], taper = 0, pad = 0, fast = FALSE, demean = TRUE,
detrend = TRUE, plot = TRUE)
# Lomb-Scargle periodogram - Note: Due to the regression ansatz,
# RobPer is not able to compute period 2 in this case.
PP_new <- RobPer(ts=zr, weighting=FALSE, periods=1/PP_konv$freq,
                  regression="L2", model='sine')

plot(PP_konv$freq, PP_konv$spec, ylab="periodogram", xlab="frequency",
     main="Comparison of RobPer(...regression='LS', model='sine') and spec.pgram")
points(PP_konv$freq, PP_new$var(zr[,2]*n/2, type="1")
      legend("top", lty=c(1,0), pch=c(-5,1), legend=c("RobPer\times\var(y)*n/2", "spec.pgram"))
# Due to different ways of computation, the scaled periodograms are not exactly
# identical, but show very similar behavior.

sampler

Generator for irregularly sampled observation times

Description

Generates irregularly sampled observation times with a periodic sampling pattern

Usage

sampler(ttype, npoints, ncycles, ps = 1)

Arguments

| ttype     | character string: Specifying the sampling pattern. Possible options: "equi" and "unif" for unperiodic sampling, "sine" and "trian" for sampling with a periodic density (see Details). |
| npoints   | integer: Sample size n (see Details). |
| ncycles   | integer: Number of sampling cycles n_s (see Details). |
| ps        | positive numeric value: Sampling period p_s (see Details). |
Details

sampler generates observation times \( t_1, \ldots, t_n \) with a periodic sampling of period \( p_s \). Four distributions are possible: In case of type="equi", the \( t_i \) are equidistantly sampled with \( t_i = i \frac{p_s}{n_s} \). For type="unif", the observation times are independently drawn from a uniform distribution on \([0, n_s p_s]\). Both these sampling schemes are aperiodic, the sampling period \( p_s \) only influences the length \( t_n - t_1 \) of the series of observation times.

For type="sine" and type="trian", observation cycles \( z_i^* \) are drawn from a uniform distribution on \( \{1, \ldots, n_s\} \) and observation phases \( \varphi_i^* \) are drawn from a density

\[
    d_{\text{sine}}(x) = \sin(2\pi x) + 1
\]

(for type="sine") or

\[
    d_{\text{trian}}(x) = \begin{cases} 
    3x, & 0 \leq x \leq \frac{2}{3}, \\
    6 - 6x, & \frac{2}{3} < x \leq 1 
    \end{cases}
\]

(for type="trian"). The unsorted observation times \( t_i^* \) are then generated using

\[
    t_i^* = \varphi_i^* + (z_i^* - 1)p_s.
\]

Separately sampling observation cycle and phase was proposed by Hall and Yin (2003). For more details see Thieler, Fried and Rathjens (2013) or Thieler et al. (2013).

Value

numeric vector: Ordered observation times.

Note

To sample from \( d_{\text{sine}} \), the function BBsolve, package BB, is used.

A former version of this function is used in Thieler et al. (2013).

Author(s)

Anita M. Thieler and Jonathan Rathjens

References


signalgen

Generator for periodic signal in a light curve

Description
Calculates periodically varying values for given observation times.

Usage
signalgen(tt, ytype, pf = 1)

Arguments
- `tt` numeric vector: Observation times \( t_1, \ldots, t_n \) (see Details).
- `ytype` character string: Specifying the shape of the periodic fluctuation (see Details).
  Possible choices are "const", "sine", "trian", "peak".
- `pf` positive numeric value: Fluctuation period \( p_f \).

Details
The values \( y_{f;1}, \ldots, y_{f;n} \) with fluctuation period \( p_f \) and related to observation times \( t_1, \ldots, t_n \) are generated using

\[
y_{f;i} = f \left( \frac{t_i}{p_f} \right), \quad i = 1, \ldots, n.
\]

Depending on `ytype` (see above), \( f \) is defined as:

\[
f_{\text{const}}(t) = 0,
\]

\[
f_{\text{sine}}(t) = \sin \left( \frac{2\pi t}{p_f} \right),
\]

\[
f_{\text{trian}}(t) = 3\varphi_1(t), \quad 0 \leq \varphi_1(t) \leq \frac{2}{3},
\]

\[
f_{\text{trian}}(t) = 6 - 6\varphi_1(t), \quad \frac{2}{3} < \varphi_1(t) \leq 1,
\]

\[
f_{\text{peak}}(t) = 9 \exp \left( -3p_f^2 \left( \varphi_1(t) - \frac{2}{3} \right)^2 \right), \quad 0 \leq \varphi_1(t) \leq \frac{2}{3},
\]
\[ f_{\text{peak}}(t) = 9 \exp \left( -12p_f^2 \left( \varphi_1(t) - \frac{2}{3} \right)^2 \right), \quad \frac{2}{3} < \varphi_1(t) \leq 1, \]

with \( \varphi_1(t) = \text{tmod1} = (t - \lfloor t/p_f \rfloor p_f)/p_f = (t\text{mod}1)/p_f \). \( f_{\text{const}} \) means that there is no (periodic) fluctuation, \( f_{\text{sine}} \) defines a sine function, \( f_{\text{trian}} \) defines a triangular shaped periodic function and \( f_{\text{peak}} \) a periodically repeating peak.

**Value**

numeric vector: Values \( y_{f;1}, \ldots, y_{f;n} \).

**Note**

This function is used in Thieler et al. (2013). See also Thieler, Fried and Rathjens (2013).

**Author(s)**

Anita M. Thieler and Jonathan Rathjens

**References**


**See Also**

Applied in \texttt{tsgen} (see there for an example).

---

**star_groj0422.32**  
*Data: Light curve from GROJ0422+32*

**Description**

Light curve for gamma ray emission of the source GROJ0422+32.

**Usage**

\texttt{star_groj0422.32}
**Format**

A matrix of three columns, with a time series of length 729 appropriate to RobPer.

**Details**

Data obtained by the BATSE Earth Occultation Monitoring project of the NSSTC, available from [http://gammaray.nsstc.nasa.gov/batse/occultation](http://gammaray.nsstc.nasa.gov/batse/occultation). The experiments are described in Harmon et al. (2002) and Harmon et al. (2004).

**Note**

See Vignette Section 5.2 for example.

**Source**

Data kindly provided by the National Aeronautics and Space Administration (NASA); National Space, Science, and Technology Center (NSSTC); Gamma-Ray Astrophysics Team (see Details).

**References**


---

**TK95**

*Power law noise generator*

**Description**

Generates an equidistant time series of power law noise according to Timmer and König (1995).

**Usage**

```
TK95(N = 1000, alpha = 1.5)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>integer value: Length of the generated time series.</td>
</tr>
<tr>
<td>alpha</td>
<td>numeric value: Exponent of the power law. White noise has exponent 0, flicker noise (pink noise) has exponent 1, brown noise has exponent 2.</td>
</tr>
</tbody>
</table>
Value

numeric vector: The generated time series.

Note

This function is used in Thieler et al. (2013). See also Thieler, Fried and Rathjens (2013).

Author(s)

Anita M. Thieler with contributions of Uwe Ligges

References


See Also

Applied in tsgen by TK95_uneq.

Examples

```r
set.seed(31)
# Generate power law noise with exponent alpha=1.5:
y <- TK95(N=2000, alpha=1.5)
tt <- seq(along=y)

# Show time series:
plot(tt, y, type="l", main="Power Law Noise", xlab="t", ylab="y")

# Plot Fourier periodogram with log-axes:
temp <- spectrum(y, plot=FALSE)
plot(log(temp$freq), log(temp$spec), main="log-log-Fourier periodogram",
xlab="log(frequency)", ylab="log(periodogram)")

# A line with slope -alpha for comparison
abline(a=8, b=-1.5, col="red")
text(-2, 12, expression(alpha==1.5), col="red")
```
TK95_uneq  

Power law noise generator for unequally sampled observation times

Description

Generates power law noise using TK95 according to Timmer and König (1995), with modifications proposed in Uttley, McHardy and Papadakis (2002) for given irregular observation times.

Usage

TK95_uneq(tt, alpha = 1.5)

Arguments

tt  numeric vector: Observation times given.
alpha numeric value: exponent of the power law. White noise has exponent 0, flicker noise (pink noise) has exponent 1, brown noise has exponent 2.

Value

numeric vector: Noise values related to the observation times.

Note

This function is applied in Thieler et al. (2013). See also Thieler, Fried and Rathjens (2013).

Author(s)

Anita M. Thieler

References


See Also

Applies TK95, applied in tsgen.

Examples

# Compare with example in TK95 to see that the power law is much more clear in
# equally sampled data!
set.seed(31)
# Generate power law noise with exponent alpha=1.5:
tt <- sampler(ttype="unif", ps=1, ncycles=2000, npoints=2000)
y <- TK95_uneq(tt, alpha=1.5)

# Show time series:
plot(tt,y, type="l", main="Irregular Power Law Noise", xlab="t", ylab="y")

# Plot Lomb-Scargle periodogram with log-axes:
temp <- RobPer(cbind(tt,y,1), weighting=FALSE, model="sine", regression="L2",
    periods=seq(2, 1000, 2))
plot(log(seq(2, 1000, 2)/2000), log(temp), main="log-log-Fourier periodogram",
    xlab="log(frequency)", ylab="log(periodogram)")
title(main = "Power Law not so obvious", cex.main=0.8, line=0.5)

# A line with slope -alpha for comparison
abline(a=-10, b=-1.5, col="red")
text(-5, -1.5, expression(alpha==1.5), col="red")

---

tsgen

**Artificial light curve generator**

Description

This function generates light curves (special time series) with unequally sampled observation times, different periodicities both in sampling and observed values, with white and power law (red) noise in the observed values and possibly disturbed observations. See RobPer-package for more information about light curves and also Thieler, Fried and Rathjens (2013) for more details in general.

Usage

tsgen(ttype, ytype, pf, redpart, s.outlier.fraction = 0, interval, npoints, ncycles, ps, SNR, alpha = 1.5)

Arguments

ttype character string: Specifying the sampling pattern. Possible choices are "equi" for equidistant sampling without gaps (unperiodic), "unif" for uniform non-equidistant unperiodic sampling, "sine" for sampling with periodic sine density, "trian" for sampling with periodic triangular density, both with period $p_s$ (see Details and sampler).
ytype  character string: Specifying the shape of the periodic fluctuation with period \( p_f \). Possible choices are "const" for constantly being zero (so no periodicity), "sine" for a sine, "trian" for a periodic triangular function, "peak" for a peak function (see Details and signalgen for more details).

\( p_f \)  positive number: Period \( p_f \) of the periodic fluctuation, argument of signalgen (see Details and signalgen).

redpart  numeric value in \([0,1]\): Proportion of the power law noise in noise components (see Details). The generated measurement accuracies \( s_i \) do not contain information about this noise component.

s.outlier.fraction  numeric value in \([0,1]\): Fraction of measurement accuracies to be replaced by outliers. A value of 0 means that no measurement accuracy is replaced by an outlier (for more details see disturber).

interval  logical: If TRUE, the observed values belonging to a random time interval of length \( 3 p_s \) are replaced by atypical values (for more details see disturber).

npoints  integer value: Defines the sample size \( n \) for the generated light curve.

ncycles  integer value: number \( n \) of sampling cycles that is observed (see Details).

ps  positive number: Sampling period \( p_s \), influencing the sampling and how the light curve is disturbed (see Details and disturber).

SNR  positive number: Defines the relation between signal \( y_f \) and noise \( y_w + y_r \) (see Details).

alpha  numeric value: Power law index \( \alpha \) for the power law noise component \( y_r \) (see Details).

Details
tsgen generates an artificial light curve consisting of observation times \( t_1, \ldots, t_n \), observation values \( y_1, \ldots, y_n \) and measurement accuracies \( s_1, \ldots, s_n \). It calls several subfunctions (see there for details):
sampler is used to sample observation times \( t_1, \ldots, t_n \) in the interval \([0, n_s \times p_s]\) with a possibly periodic sampling of period \( p_s \).
signalgen generates periodically varying values \( y_{f1}, \ldots, y_{fn} \) at time points \( t_1, \ldots, t_n \) with fluctuation period \( p_f \).
lc_noise samples measurement accuracies \( s_1, \ldots, s_n \) from a Gamma(3,10)-distribution and a white noise component \( y_{w1}, \ldots, y_{wn} \) with from \( \mathcal{N}(0, s_i^2) \) distributions. A second noise component \( y_{r1}, \ldots, y_{rn} \) does not depend on the \( s_i \). It is generated as red noise, i.e. following a power law with power law index \( \alpha \). For white noise choose \( \alpha = 0 \), for flicker noise (pink noise) \( \alpha = 1 \), for brown noise \( \alpha = 2 \). The power law noise is generated using TK95.uneq and TK95. The noise components are scaled so that the variance of the \( y_{ri} \) has approximately the proportion \( \text{redpart} \) in the overall noise variance and that \( \text{SNR} \) is the ratio \( \text{var}(y_f)/\text{var}(y_w + y_r) \). The observed values are set to \( y_i = y_{fi} + y_{wi} + y_{ri}/s_i \).
disturber disturbs the light curve replacing measurement accuracies \( s_i \) by outliers (if \( \text{s.outlier.fraction}>0 \)) and observed values \( y_i \) by atypical values (if interval=TRUE). In case of \( \text{s.outlier.fraction}=0 \) and interval=FALSE, the function returns all values unchanged.
Value

- **tt**: numeric vector: Generated observation times $t_1, \ldots, t_n$.
- **y**: numeric vector: Generated observation values $y_1, \ldots, y_n$.
- **s**: numeric vector: Generated measurement accuracies $s_1, \ldots, s_n$.

Note

Note that the white noise components’ variances are exactly $s_i^2$, so the $s_i$ are no estimates, but true values. In this sense, the measurement accuracies of a generated light curve are more informative than for real light curves, where the measurement accuracies are estimates, see Thieler et al. (2013), where also a former version of this function is applied.

To lower the informativity of the measurement accuracies, set redpart to a strictly positive value, possibly with alpha=0 if no other noise components than white ones are required.

Author(s)

Anita M. Thieler and Jonathan Rathjens

References


See Also

Applies sampler, signalgen, lc_noise, disturber, TK95, TK95_uneq.

Examples

```r
# Generate a light curve:
s.set.seed(22)
l Brace cases< tsgen(tttype="sine", ytype="peak", pf=7, redpart=0.1, s.outlier.fraction=0,
 interval=FALSE, npoints=200, ncycles=100, ps=5, SNR=3, alpha=0)

# Or do it step by step:
# First sampling observation times:
s.set.seed(22)
tt <- sampler(tttype="sine", npoints=200, ncycles=100, ps=5)

# show obviously irregular observation times as grey vertical bars on a red time line:
plot(tt, tt*0, type="n", axes=FALSE, xlab="Observation Times", ylab="")
aline(v=tt, col="grey")
axis(1, pos=0, col="red", col.axis="red")
```
### Description

This function is used to create the designmatrices needed in RobPer to fit periodic functions. See RobPer or Thieler, Fried and Rathjens (2013) for Details.

### Usage

```r
Xgen(tt, n, s, pp, design, steps = 10)
```
Arguments

- **tt**: real-valued vector of length \( n \): Observation times.
- **n**: integer: Sample size.
- **s**: numeric vector of length \( n \): Measurement errors to perform weighted regression, else set \( s = \text{rep}(1, n) \).
- **pp**: positive number: Trial period.
- **design**: character string: Shape of the periodic function to be fitted, possible choices are "step", "sine", "fourier(2)", "fourier(3)", "splines" (see RobPer for details) and "stepb". The latter generates a step function like "step", but with opposite jumping time points. This is needed for RobPer with \( \text{model} = "2\text{step}" \) (see RobPer).
- **steps**: Number of steps for the step functions

Value

numeric matrix: Designmatrix.

Note

A former version of this function is used in Thieler et al. (2013).

Author(s)

Anita M. Thieler and Jonathan Rathjens

References


See Also

Applied in RobPer, see FastTau for an example.
Index

*Topic **datasets**
  * Mrk421, 13
  * Mrk501, 15
  * star_groj0422.32, 23

betaCvMfit, 3, 4

disturber, 7, 28, 29

FastS, 3, 8, 18, 19
FastTau, 3, 9, 10, 18, 19, 31

lc_noise, 11, 28, 29

Mrk421, 13
Mrk501, 15

RobPer, 9, 11, 13, 15, 17, 24, 31
RobPer-package, 2

sampler, 20, 27–29
signalgen, 22, 28, 29
star_groj0422.32, 23

TK95, 3, 24, 26–29
TK95_uneq, 12, 19, 25, 26, 28, 29
tsgen, 8, 12, 22, 23, 25, 27, 27

Xgen, 19, 30