Package: TrendLSW (via r-universe)

October 27, 2024

Type Package

Title Wavelet Methods for Analysing Locally Stationary Time Series

Version 1.0.2.9000

Depends R (>= 4.1.0)

Maintainer Euan T. McGonigle <e.t.mcgonigle@soton.ac.uk>

Description Fitting models for, and simulation of, trend locally stationary wavelet (TLSW) time series models, which take account of time-varying trend and dependence structure in a univariate time series. The TLSW model, and its estimation, is described in McGonigle, Killick and Nunes (2022a) <doi:10.1111/jtsa.12643>, (2022b) <doi:10.1214/22-EJS2044>. New users will likely want to start with the TLSW function.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports wavethresh, locits

URL https://github.com/EuanMcGonigle/TrendLSW

BugReports https://github.com/EuanMcGonigle/TrendLSW/issues

RoxygenNote 7.3.1

Suggests testthat (>= 3.0.0), vdiffr

Config/testthat/edition 3

Repository https://euanmcgonigle.r-universe.dev

RemoteUrl https://github.com/euanmcgonigle/trendlsw

RemoteRef HEAD

RemoteSha 638f22a1a2481cc0a4032eec1bb409b1dd1357d9

Contents

`TrendLSW-package` *Wavelet Methods for Analysing Locally Stationary Time Series*

Description

Provides wavelet-based methods for trend, spectrum and autocovariance estimation of locally stationary time series. See `TLSW` for the main estimation function.

Details

Package:	TrendLSW
Type:	Package
Version:	1.0.0
Date:	2024-04-17
License:	GPL
LazyLoad:	yes

Author(s)

Euan T. McGonigle <e.t.mcgonigle@soton.ac.uk>, Rebecca Killick <r.killick@lancs.ac.uk>, and Matthew Nunes <m.a.nunes@bath.ac.uk>

Maintainer: Euan T. McGonigle <e.t.mcgonigle@soton.ac.uk>

References

Spectral estimation with differencing/nonlinear trend estimator: McGonigle, E. T., Killick, R., and Nunes, M. (2022). Modelling time-varying first and second-order structure of time series via wavelets and differencing. *Electronic Journal of Statistics*, 6(2), 4398-4448.

Spectral estimation in presence of trend/linear trend estimator: McGonigle, E. T., Killick, R., and Nunes, M. (2022). Trend locally stationary wavelet processes. *Journal of Time Series Analysis*, 43(6), 895-917.

LSW processes without trend: Nason, G. P., von Sachs, R., and Kroisandt, G. (2000). Wavelet processes and adaptive estimation of the evolutionary wavelet spectrum. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **62(2)**, 271–292.

lacf estimation without trend: Nason, G. P. (2013). A test for second-order stationarity and approximate confidence intervals for localized autocovariances for locally stationary time series. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **75(5)**, 879–904.

See Also

TLSW,TLSWsim,plot.TLSW

Examples

```
# simulates an example time series and estimates its trend and evolutionary wavelet spectrum

spec <- matrix(0, nrow = 9, ncol = 512)
spec[1,] <- 1 + sin(seq(from = 0, to = 2 * pi, length = 512))^2

trend <- seq(from = 0, to = 5, length = 512)

set.seed(1)

x <- TLSWsim(trend = trend, spec = spec)

x.TLSW <- TLSW(x)

summary(x.TLSW)

plot(x.TLSW)
```

---

celegensbio *Bioluminescence of C. Elegens*

Description

This dataset gives the time series of bioluminescence of an experiment monitoring C. Elegens as they feed and forage. The observations are taken 6-minutes apart with no missing data.

Usage

```
celegensbio
```

Format

A vector of length 623.

Source

Experiment from Alexandre Benedetto's research group at Lancaster University.

`plot.TLSW` *Plot Trend and/or Spectrum Information in a* `TLSW` *Object*

Description

Plots information contained within a `TLSW` object. Depending on the `plot.type` option this will produce a plot of the data with trend estimate overlayed, a plot of the spectral estimate, or both (default). If the `TLSW` object does not contain trend or spectral estimates and these are requested a warning will be given.

Usage

```
## S3 method for class 'TLSW'
plot(
  x,
  plot.type = c("trend", "spec"),
  trend.plot.args,
  spec.plot.args,
  plot.CI = TRUE,
  ...
)
```

Arguments

`x`	A `TLSW` object
`plot.type`	A string object indicating what is to be plotted. Can be: • `"trend"`: will plot the trend estimate only. • `"spec"`: will plot the spectral estimate only. • `c("trend", "spec")`: the default value will plot both the trend and spectral estimate.
`trend.plot.args`	A list object, that includes any choices for the graphical parameters used for plotting the trend estimate.
`spec.plot.args`	A list object, that includes any choices for the graphical parameters used for plotting the spectral estimate.
`plot.CI`	A logical variable. If TRUE, the confidence interval of the trend estimate (if computed) will be included in the plot.
`...`	Any additional arguments that will be applied to the graphical parameters of both the trend and spectrum plotting.

Details

A TLSW object can be plotted using the standard `plot` function in R to display the estimated trend function and wavelet spectrum. The estimated trend is visualised using `plot.default`. Visualisation of the estimated spectrum is based on `plot.wd`, for which credit belongs to Guy Nason. Graphical parameters for customising the display of the trend or spectrum plots should be given to the `trend.plot.args` and `spec.plot.args` arguments respectively. For graphical parameters for the trend plot:

- Parameters related to the overall plot should be provided as they usually would be when using the `plot` function, in the `trend.plot.args` list object. For example, to change the title of the plot to "Plot", use `main = "Plot"`.
- Parameters affecting the display of the estimated trend line should begin with the prefix `"T."`. For example, to set the colour of the trend line to blue, use `T.col = "blue"`.
- Parameters affecting the display of the confidence interval lines should begin with the prefix `"CI."`. For example, to set the line width of the confidence interval to 2, use `CI.lwd = 2`.
- Parameters affecting the display of the polygon drawn by the confidence interval should begin with the prefix `"poly."`. For example, to set the colour of the confidence interval region to green, use `poly.col = "green"`.

Value

No return value, called for side effects

References

McGonigle, E. T., Killick, R., and Nunes, M. (2022). Modelling time-varying first and second-order structure of time series via wavelets and differencing. *Electronic Journal of Statistics*, 6(2), 4398-4448.

McGonigle, E. T., Killick, R., and Nunes, M. (2022). Trend locally stationary wavelet processes. *Journal of Time Series Analysis*, 43(6), 895-917.

See Also

`TLSW`, `summary.TLSW`, `print.TLSW`, `plot.wd`

Examples

```
# Simulates an example time series and estimates its trend and evolutionary wavelet spectrum.
# Then plots both estimates.

spec <- matrix(0, nrow = 9, ncol = 512)

spec[1, ] <- 4 + 4 * sin(seq(from = 0, to = 2 * pi, length = 512))^2

trend <- seq(from = 0, to = 10, length = 512) + 2 * sin(seq(from = 0, to = 2 * pi, length = 512))

set.seed(1)

x <- TLSWsim(trend = trend, spec = spec)
```

```
x.TLSW <- TLSW(x)

plot(x.TLSW, trend.plot.args = list(
  ylab = "Simulated Data", T.col = 4,
  T.lwd = 2, T.lty = 2
))
```

`print.TLSW` *Print an Object of Class* `TLSW`

Description

Prints a `TLSW` object, alongside summary information. The first part prints details of the class, specifically the names of elements within. Then prints out the summary, which gives information about a `TLSW` object. If spectral estimation was performed, then the type of smoothing and binwidth is printed, along with the differencing performed if it is used, the maximum wavelet scale analysed, and whether or not boundary handling was used. If trend estimation is performed, then the type of wavelet thresholding and transform used is printed, as well as the maximum wavelet scale used, whether or not boundary handling was used, and the significance of the confidence interval if it was calculated.

Usage

```
## S3 method for class 'TLSW'
print(x, ...)
```

Arguments

`x`	A `TLSW` object.
`...`	Other arguments.

Value

No return value, called for side effects

References

McGonigle, E. T., Killick, R., and Nunes, M. (2022). Modelling time-varying first and second-order structure of time series via wavelets and differencing. *Electronic Journal of Statistics*, 6(2), 4398-4448.

McGonigle, E. T., Killick, R., and Nunes, M. (2022). Trend locally stationary wavelet processes. *Journal of Time Series Analysis*, 43(6), 895-917.

See Also

`TLSW`, `summary.TLSW`

Examples

```
# simulates an example time series and estimates its trend and evolutionary wavelet spectrum

spec <- wavethresh::cns(512)
spec <- wavethresh::putD(spec, level = 8, 1 + sin(seq(from = 0, to = 2 * pi, length = 512))^2)

trend <- seq(from = 0, to = 5, length = 512)

set.seed(1)

x <- TLSWsim(trend = trend, spec = spec)

x.TLSW <- TLSW(x)

print(x.TLSW)
```

`summary.TLSW` *Summary of Output Provided by the* `TLSW` *Function*

Description

Summary method for objects of class `TLSW`.

Usage

```
## S3 method for class 'TLSW'
summary(object, ...)
```

Arguments

`object`	A `TLSW` object.
`...`	Other arguments.

Details

Prints out information about a `TLSW` object. If spectral estimation was performed, then the type of smoothing and binwidth is printed, along with the differencing performed if it is used, the maximum wavelet scale analysed, and whether or not boundary handling was used. If trend estimation is performed, then the type of wavelet thresholding and transform used is printed, as well as the maximum wavelet scale used, whether or not boundary handling was used, and the significance of the confidence interval if it was calculated.

Value

No return value, called for side effects

References

McGonigle, E. T., Killick, R., and Nunes, M. (2022). Modelling time-varying first and second-order structure of time series via wavelets and differencing. *Electronic Journal of Statistics*, 6(2), 4398-4448.

McGonigle, E. T., Killick, R., and Nunes, M. (2022). Trend locally stationary wavelet processes. *Journal of Time Series Analysis*, 43(6), 895-917.

See Also

TLSW, print.TLSW

Examples

```
# simulates an example time series and estimates its trend and evolutionary wavelet spectrum

spec <- matrix(0, nrow = 10, ncol = 2^10)

spec[1, ] <- seq(from = 1, to = 10, length = 1024)

trend <- sin(pi * (seq(from = 0, to = 4, length = 1024)))

set.seed(1)

x <- TLSWsim(trend = trend, spec = spec)

x.TLSW <- TLSW(x)

summary(x.TLSW)
```

TLSW *Estimate Trend and Spectrum of Trend Locally Stationary Wavelet Process*

Description

Using wavelet-based methods, this function estimates the trend and evolutionary wavelet spectrum (EWS) of a nonstationary time series.

Two methods are implemented (see references), the direct estimator (`T.est.type="linear"` and `S.do.diff=FALSE`), and the difference estimator (`T.est.type="nonlinear"`) and `S.do.diff=TRUE`) The defaults give the direct estimator.

All the defaults are set carefully. Key times to change defaults are

- if the data contains "cusps", then the difference estimator is preferred.
- to assess stability of the estimate to the wavelet, change the wavelet number `T.filter.number` and `S.filter.number` and/or the wavelet type `T.family` and `S.family`, see details.

The arguments affecting trend are preceded by `T.` and those affecting spectral estimation are preceded by `S.`.

Usage

```
TLSW(
  x,
  do.trend.est = TRUE,
  do.spec.est = TRUE,
  T.est.type = "linear",
  T.filter.number = 4,
  T.family = "DaubExPhase",
  T.transform = "nondec",
  T.boundary.handle = TRUE,
  T.max.scale = floor(log2(length(x)) * 0.7),
  T.thresh.type = "hard",
  T.thresh.normal = TRUE,
  T.CI = FALSE,
  T.sig.lvl = 0.05,
  T.reps = 200,
  T.CI.type = "normal",
  T.lacf.max.lag = floor(10 * (log10(length(x)))),
  S.filter.number = 4,
  S.family = "DaubExPhase",
  S.smooth = TRUE,
  S.smooth.type = "mean",
  S.binwidth = floor(6 * sqrt(length(x))),
  S.max.scale = floor(log2(length(x)) * 0.7),
  S.boundary.handle = TRUE,
  S.inv.mat = NULL,
  S.do.diff = FALSE,
  S.lag = 1,
  S.diff.number = 1,
  gen.filter.number = S.filter.number,
  gen.family = S.family
)
```

Arguments

x	The time series you wish to analyse.
do.trend.est	Logical variable, indicating whether trend estimation is to be performed on the time series.
do.spec.est	Logical variable, indicating whether spectral estimation is to be performed on the time series.
T.est.type	String indicating type of wavelet thresholding used. Can be "linear" (default), which means that all non-boundary wavelet coefficients are set to zero, or "nonlinear", where each wavelet coefficient is thresholded using a time-varying, noise-dependent threshold.
T.filter.number	The index number for the wavelet used for trend estimation.
T.family	The family of the wavelet used for trend estimation.

`T.transform` String giving the type of wavelet transform used for trend estimation. Can be `"dec"`, in which case a standard (decimated) wavelet transform is used, or `"nondec"` (default), in which case a nondecimated transform is used.

`T.boundary.handle` Logical variable, if `TRUE`, the time series is boundary corrected when estimating the trend.

`T.max.scale` Integer variable, selects the number of scales of the wavelet transform to apply thresholding to for trend estimation.

`T.thresh.type` String variable, used only if `T.est.type = "nonlinear"`; the type of thresholding function used in the trend estimation. Can be `"soft"` or `"hard"` (default).

`T.thresh.normal` Logical variable, used only if `T.est.type = "nonlinear"`; if `TRUE`, uses a threshold assuming the data are normally distributed. If `FALSE`, uses a larger threshold to reflect non-normality.

`T.CI` Logical variable. If `TRUE`, a `(1-T.sig.lvl)` pointwise confidence interval is computed for the trend estimate. When `T.transform = "dec"` and `T.est.type = "linear"`, this is computed using the asymptotic distribution of the trend estimator. Otherwise, it is computed via bootstrapping.

`T.sig.lvl` Used only if `T.CI = TRUE`; a numeric value (`0 <= T.sig.lvl <= 1`) with which a `(1-T.sig.lvl)` pointwise confidence interval for the trend estimate is generated.

`T.reps` Used only if `T.transform = "nondec"` and `T.CI = TRUE`; the number of bootstrap replications used to calculate the confidence interval.

`T.CI.type` Used only if `T.transform = "nondec"` and `T.CI = TRUE`; the type of confidence interval computed. Can be `"percentile"`, in which case empirical percentiles are used, or `"normal"` (default), in which case the (symmetric) normal approximation is used.

`T.lacf.max.lag` Used only if `T.est.type = "linear"` and `T.CI = TRUE`; the maximum lag of the autocovariance to compute needed for calculating the asymptotic confidence interval.

`S.filter.number` The index number for the wavelet used for spectrum estimation.

`S.family` The family of the wavelet used for spectrum estimation.

`S.smooth` A logical variable to indicate whether smoothing is performed on the wavelet periodogram.

`S.smooth.type` String indicating which type of smoothing to use on wavelet periodogram. Can be one of

- `"mean"`: (default) running mean smoother.
- `"median"`: running median smoother.
- `"epan"`: Epanechnikov kernel smoother.

`S.binwidth` The bin width of the smoother used to smooth the raw wavelet periodogram.

`S.max.scale` The coarsest wavelet scale used to estimate the spectrum. Should be a positive integer less than J, where $n = 2^J$ is the length of the time series.

`S.boundary.handle`
: Logical variable, if TRUE, the time series is boundary corrected, to get a more accurate spectrum estimate at the boundaries of the times series. If FALSE, no boundary correction is applied. Recommended to use TRUE.

`S.inv.mat`
: The user can pre-calculate and supply the appropriate correction matrix used to correct the raw wavelet periodogram. If left blank, then the correction matrix is calculated when performing spectral estimation.

`S.do.diff`
: Logical variable, indicating if the time series is to be differenced before spectral estimation is performed.

`S.lag`
: The lag of differencing used, only applicable if `S.do.dif` = `TRUE`.

`S.diff.number`
: The number of differencing operations performed, only applicable if `S.do.diff` = `TRUE`. A first difference is strongly recommended as default.

`gen.filter.number`
: The index number for the wavelet that generates the stochastic component of the time series. For the "DaubExPhase" family, the filter number can be between 1 to 10. For the "DaubLeAsymm" family, the filter number can be between 4 to 10. Recommended to leave as the default, set to the same as `S.filter.number`.

`gen.family`
: The family of the generating wavelet. It is recommended to use either the Daubechies Extremal Phase family, or the Daubechies Least Asymmetric family, corresponding to the "DaubExPhase" or the "DaubLeAsymm" options. Recommended to leave as the default, set to the same as `S.family`.

Details

The fitted *trend LSW process* $X_{t,n}$, $t = 0, \ldots, n-1$, and $n = 2^J$ is a doubly-indexed stochastic process with the following representation in the mean square sense:

$$X_t = T_t + \varepsilon_t = T_t + \sum_{j=1}^{\infty} \sum_{k \in \mathbb{Z}} w_{j,k;n} \psi_{j,k}(t) \xi_{j,k},$$

where $\{\xi_{j,k}\}$ is a random, uncorrelated, zero-mean orthonormal increment sequence, $\{w_{j,k;n}\}$ is a set of amplitudes, and $\{\psi_{j,k}\}_{j,k}$ is a set of discrete non-decimated wavelets. The trend component $T_t := T(t/n)$ is assumed to be a general smooth (Holder) continuous function. See the referenced papers for full details of the model. The key considerations for users are:

- The model assumes smooth trend and spectral components. The larger the `T.filter.number` the smoother the assumption on the underlying trend and similarly for `S.filter.number` and the spectral estimate.
- The choice of wavelet (smoothness assumption) does affect the estimation so one should check the robustness of their conclusions to the choice of wavelet (`T.filter.number` and `S.filter.number`). This is akin to selecting the kernel in nonparametric modelling.
- The underlying methods are designed for signals of length $n = 2^J$ and so modifications are made to signals which are not of this form. A natural approach is to extend the data (at both ends) and the default approach does this by reflection with a trend correction to avoid discontinuities.

Value

An object of class "TLSW", a list that contains the following components:

x	Input data
do.spec.est	Input parameter, logical variable specifying if spectral estimation was performed.
spec.est	A list object, returned if do.spec.est = TRUE. Contains relevant input parameters and the following fields related to the spectrum estimate: • S: The evolutionary wavelet spectral (smoothed and corrected) estimate of the input data. This object is of class wd and so can be plotted and printed in the usual way using wavethresh functionality. • WavPer: The raw wavelet periodogram of the input data. The EWS estimate (S, above) is the smoothed corrected version of this raw wavelet periodogram. • SmoothWavPer: The smoothed, but uncorrected raw wavelet periodogram of the input data.
do.trend.est	Input parameter, logical variable specifying if trend estimation was performed.
trend.est	A list object, returned if do.trend.est = TRUE. Contains relevant input parameters and the following fields related to the trend estimate: • T: A vector of length length(x) containing the trend estimate. • lower.CI: Returned if T.CI = TRUE. The lower limit of the pointwise confidence interval. • upper.CI: Returned if T.CI = TRUE. The upper limit of the pointwise confidence interval.

References

McGonigle, E. T., Killick, R., and Nunes, M. (2022a). Trend locally stationary wavelet processes. *Journal of Time Series Analysis*, 43(6), 895-917.

McGonigle, E. T., Killick, R., and Nunes, M. (2022b). Modelling time-varying first and second-order structure of time series via wavelets and differencing. *Electronic Journal of Statistics*, 6(2), 4398-4448.

See Also

plot.TLSW, summary.TLSW, print.TLSW, wd, ewspec3

Examples

```
# simulates an example time series and estimates its trend and evolutionary wavelet spectrum

spec <- matrix(0, nrow = 10, ncol = 2^10)

spec[1, ] <- seq(from = 1, to = 10, length = 1024)

trend <- sin(pi * (seq(from = 0, to = 4, length = 1024)))

set.seed(1)
```

```
x <- TLSWsim(trend = trend, spec = spec)

plot.ts(x)

x.TLSW <- TLSW(x)

summary(x.TLSW)

plot(x.TLSW) # by default plots both the trend and spectrum estimates
```

TLSWlacf *Compute Localised Autocovariance Estimate of a TLSW Object*

Description

Computes the local autocovariance and autocorrelation estimates, given an input of an object of class TLSW containing the estimated spectrum. Provides the same functionality as the function `lacf` from the `locits` package, but user provides an object of class `TLSW` as the main argument.

Usage

```
TLSWlacf(x.TLSW, lag.max = NULL)
```

Arguments

`x.TLSW`	a `TLSW` object.
`lag.max`	The maximum lag of acf required. If NULL then the same default as in the regular acf function is used.

Value

An object of class `lacf` which contains the following components:

- `lacf`: a matrix containing the estimate of the local autocovariance. Columns represent lags (beginning at lag 0), and rows represent time points.
- `lacr`: a matrix containing the estimate of the local autocorrelation. Columns represent lags (beginning at lag 0), and rows represent time points.
- `name`: the name of the time series (if applicable).
- `date`: the date the function was executed.
- `SmoothWP`: The smoothed, un-corrected raw wavelet periodogram of the input data.
- `S`: the spectral estimate used to compute the local autocovariance.
- `J`: the number of total wavelet scales.

References

McGonigle, E. T., Killick, R., and Nunes, M. (2022). Trend locally stationary wavelet processes. *Journal of Time Series Analysis*, 43(6), 895-917.

Nason, G. P. (2013). A test for second-order stationarity and approximate confidence intervals for localized autocovariances for locally stationary time series. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **75(5)**, 879–904.

Nason, G. P. (2016). locits: Tests of stationarity and localized autocovariance. R package version 1.7.3.

See Also

lacf

Examples

```
## ---- computes estimate of local autocovariance function


spec <- matrix(0, nrow = 9, ncol = 512)
spec[2, ] <- 1 + sin(seq(from = 0, to = 2 * pi, length = 512))^2

trend <- seq(from = 0, to = 10, length = 512)

set.seed(123)

x <- TLSWsim(trend = trend, spec = spec)

## ---- first estimate the spectrum:

x.TLSW <- TLSW(x)

#---- estimate the lacf:

lacf.est <- TLSWlacf(x.TLSW)

#---- plot the variance (lag 0 lacf) over time:

plot.ts(lacf.est$lacf[, 1], ylab = "Variance")
```

TLSWsim *Simulate Trend Locally Stationary Wavelet Process*

Description

Simulates a trend locally stationary wavelet process with a given trend function and spectrum. Extends the LSWsim function from the wavethresh package.

Usage

```
TLSWsim(
  trend,
  spec,
  filter.number = 4,
  family = "DaubExPhase",
  innov.func,
  ...
)
```

Arguments

`trend`	Either: • A numeric vector of length n giving the values of the deterministic trend function, • A real-valued function of one argument defined on rescaled time $[0, 1)$. When using a numeric vector for `trend`, if n is not a power of 2 then `spec` must be specified using a numeric matrix of dimensions $\lfloor \log_2(n) \rfloor \times n$.
`spec`	Either: • A `wavethresh` object of class wd which contains the spectrum for simulating an LSW process, • A numeric matrix of dimensions $J \times n$, where the j-th row corresponds to the spectrum values at scale j and $\lfloor \log_2(n) \rfloor = J$, • A list of length $J = \log_2(n)$, where the j-th element of the list is a function of one argument specifying the spectrum function at scale j on rescaled time $[0, 1)$. When using a numeric matrix for `spec`, if n is not a power of 2 then `trend` must be specified using a numeric vector of length n.
`filter.number`	The filter number for the wavelet used to simulate the LSW process (default 4)
`family`	The family of the wavelet used to simulate the LSW process (default `DaubExPhase`).
`innov.func`	A function with first argument `n` used for simulating the innovations. By default, normal random innovations are sampled using the `rnorm` function.
`...`	Optional arguments to be passed to the function `innov.func` for sampling the innovation process.

Value

A n-length vector containing a TLSW process simulated from the trend and spectral description given by the trend and spec arguments.

See Also

`LSWsim`

Examples

```
#---- simulate with numeric trend, and spec a wd object as in wavethresh-----

spec <- wavethresh::cns(1024)

spec <- wavethresh::putD(spec, level = 8, seq(from = 2, to = 8, length = 1024))

trend <- sin(pi * (seq(from = 0, to = 4, length = 1024)))

x <- TLSWsim(trend = trend, spec = spec)

plot.ts(x)

#---- simulate with numeric trend, and spec a matrix, with non-dyadic n-----

spec <- matrix(0, nrow = 9, ncol = 1000)

spec[1, ] <- seq(from = 1, to = 10, length = 1000)

trend <- sin(pi * (seq(from = 0, to = 4, length = 1000)))

x <- TLSWsim(trend = trend, spec = spec)

plot.ts(x)

#---- simulate with functional trend, and spec a list of functions-----

spec <- vector(mode = "list", length = 10)

spec[[1]] <- function(u) {
  1 + 9 * u
}

trend <- function(u) {
  sin(pi * u)
}

x <- TLSWsim(trend = trend, spec = spec)

plot.ts(x)
```

---

z.acc *Z-Axis Acceleration for Human Activity Monitoring*

Description

This dataset is a section of data from Experiment 3, User 2, based on accelerometer readings from a smartphone (Reyes-Ortiz, Oneto, Sama, Parra, and Anguita (2016)), obtained from the UCI data

repository (Kelly, Longjohn, and Nottingham (2024)). The data gives the time series of the acceleration along the Z-axis of an experiment participant as they perform the activities of walking up and downstairs several times.

Usage

```
z.acc
```

Format

A vector of length 6000.

Source

Kelly M., Longjohn R., and Nottingham, K. (2024). The UCI Machine Learning Repository. doi:10.24432/C54G7M.

References

Reyes-Ortiz, J. L., Oneto, L., Sama, A., Parra, X., and Anguita, D. (2016). Transition-Aware Human Activity Recognition Using Smartphones. *Neurocomputing*, 171, 754–767.

See Also

z.labels

---

z.labels *Activity Labels for Human Activity Monitoring*

Description

This dataset gives the labelled activities recorded during the time period of observations given in the data object `z.acc`.

Usage

```
z.labels
```

Format

A data frame with 6 rows and 3 variables:

activity The activity recorded, either `"downstairs"` or `"upstairs"`, corresponding to walking downstairs and upstairs respectively.

start the starting time of the activity.

end the ending time of the activity.

Source

Kelly M., Longjohn R., and Nottingham, K. (2024). The UCI Machine Learning Repository. doi:10.24432/C54G7M.

References

Reyes-Ortiz, J. L., Oneto, L., Sama, A., Parra, X., and Anguita, D. (2016). Transition-Aware Human Activity Recognition Using Smartphones. *Neurocomputing*, 171, 754–767.

See Also

`z.acc`

Index