Package 'lpacf'

May 12, 2023

Type Package

Title Local Partial Autocorrelation Function Estimation for Locally Stationary Wavelet Processes

Version 1.0.1

Date 2023-05-12

Author Rebecca Killick [aut, cre],
Guy Nason [aut],
Marina Knight [aut],
Matt Nunes [aut],
Idris Eckley [ctb]

Maintainer Rebecca Killick <r.killick@lancs.ac.uk>

Description

Provides the method for computing the local partial autocorrelation function for locally stationary wavelet time series from Killick, Knight, Nason, Eckley (2020) <doi:10.1214/20-EJS1748>.

Depends R(>= 3.0), stats, locits, wavethresh, parallel, methods

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2023-05-12 10:00:03 UTC

R topics documented:

`lpacf-package` *Compute localized partial autocorrelation for locally stationary wavelet time series.*

Description

This package computes a localized partial autocorrelation of a time series assuming a locally stationary wavelet model.

Details

Package:	lpacf
Type:	Package
Version:	1.0.1
Date:	2023-05-12
License:	GPL-2

The `lpacf` function computes the localized partial autocorrelation of a locally stationary time series. The resultant object can be printed using `print.lpacf` or plotted by `plot.lpacf`.

Author(s)

Rebecca Killick, Marina Knight, Guy Nason and Matt Nunes

Maintainer: Rebecca Killick <r.killick@lancs.ac.uk>

References

Killick, R., Knight, M. I., Nason, G.P. and Eckley, I. A. (2020) The local partial autocorrelation function and some applications. *Electron. J. Statist.* **14** (2), 3268-3314. DOI: 10.1214/20-EJS1748.

See Also

`lpacf`

Examples

```
#
# See examples in each of the functions' help pages linked above.
#
```

`AutoBestBW.Epan` *Choose a good bandwidth for smoothing of a EWS spectral estimator using the Epanechnikov kernel.*

Description

Computes Epanechnikov kernel estimator closest to wavelet estimator of evolutionary wavelet spectrum. The idea is to obtain a good linear bandwidth.

Usage

```
AutoBestBW.Epan(x, filter.number = 1, family = "DaubExPhase",
    smooth.dev = var, AutoReflect = TRUE, tol = 0.1, maxits = 5,
    plot.it = FALSE, verbose = 0, ReturnAll = FALSE)
```

Arguments

`x`	Time series you want to analyze.
`filter.number`	The wavelet filter used to carry out smoothing operations.
`family`	The wavelet family used to carry out smoothing operations.
`smooth.dev`	The deviance estimate used for the smoothing (see ewspec help)
`AutoReflect`	Mitigate periodic boundary conditions of wavelet transforms by reflecting time series about RHS end before taking transforms (and is undone before returning the answer).
`tol`	Tolerance for golden section search for the best bandwidth
`maxits`	Maximum number of iterations for the golden section search
`plot.it`	Plot the values of the bandwidth and its closeness of the linear smooth to the wavelet smooth, if TRUE.
`verbose`	If nonzero prints out informative messages about the progress of the golden section search. Higher integers produce more messages.
`ReturnAll`	If TRUE then return the best bandwidth (in the ans component), the wavelet smooth (in EWS.wavelet) and the closest linear smooth (EWS.linear). If FALSE then just the bandwidth is returned.

Details

Tries to find the best running mean fit to an estimated spectrum obtained via wavelet shrinkage. The goal is to try and find a reasonable linear bandwidth.

Value

If ReturnAll argument is FALSE then the best bandwidth is returned.

Author(s)

Guy Nason.

References

Nason, G.P. (2013) A test for second-order stationarity and approximate confidence intervals for localized autocovariances for locally stationary time series. *J. R. Statist. Soc.* B, **75**, 879-904.

See Also

AutoBestBW

Examples

```
#
# Generate synthetic data
#
x <- rnorm(256)
#
# Compute best linear bandwidth
#
tmp <- AutoBestBW(x=x)
#
# Printing it out in my example gives:
# tmp
# [1] 168
```

---

lpacf *Calculates the lpacf (local partial autocorrelation function).*

Description

This calculates the local partial autocorrelation function for data x. Up to lag.max lags are calculated.

Usage

```
lpacf(x, binwidth, lag.max=NULL, filter.number=10, family="DaubLeAsymm", smooth.dev=var,
   AutoReflect=TRUE, tol=0.1, maxits=5, ABBverbose=0, lapplyfn=lapply,allpoints=FALSE,
     verbose=FALSE, ...)
```

Arguments

x	The time series you wish to analyze
binwidth	The bandwidth for the spectral smoothing. If the argument is missing or zero then the bandwidth is chosen automatically using the AutoBestBW function from the locits package. The smoothing is a simple running mean.
lag.max	The maximum lag of pacf required. If this argument is NULL then the maximum lag is computed internally to be floor(10*log10(n)) where n is the length of the series. E.g. if n=512 then the lag.max will be set to 27.

filter.number	The wavelet filter number for helping choose the smoothing binwidth, used only by AutoBestBW.
family	The wavelet family for helping choose the smoothing binwidth, used only by AutoBestBW.
smooth.dev	The type of deviance used by AutoBestBW to smooth an internal wavelet spectral estimate. The usual variance var is a good option, but something like a rescaled mad could be an alternative.
AutoReflect	Mitigate periodic boundary conditions of wavelet transforms by reflecting time series about RHS end before taking transforms (and is undone before returning the answer). Setting this to be TRUE usually dramatically improves the results as the wavelet smoothing is performed using periodic transforms.
tol	Tolerance for golden section search for the best bandwidth used only by AutoBestBW.
maxits	Maximum number of iterations for the golden section search used by AutoBestBW.
ABBverbose	If nonzero prints out informative messages about the progress of the golden section search. Higher integers produce more messages. Setting it to zero suppresses messages.
lapplyfn	Function which applies a specified function to a vector. lapply is default but this argument can be used to replace this by something more efficient, e.g. a parallel version such as mclapply. Note, if you use mclapply then you should set the option mc.cores to something sensible. E.g. I have a quad-core machine, so when I am using the machine myself, alone, I set options(mc.cores=4). If you have many cores on a multicore machine, but with several users, you might want the number of codes to be used by your code to be less than the max to be nice to others.
allpoints	The lpacf is calculated using a window centered on a time point, if allpoints=TRUE then the edges of the data are also estimated (where a smaller, non-centered binwidth is used). Note, if allpoints=TRUE you obtain an estimator for the whole length of the series, but it will be more variable nearer the ends.
verbose	If TRUE then some informative messages are printed, otherwise they're not
...	Other arguments for AutoBestBW.

Details

Calculates the local partial autocorrelation function (lpacf) for 1,...,lag.max lags. See paper in the references for more details. NOTE: Often when local (windowed) estimates are created one assigns the estimated value to the central point in the window. This is the approach we take here when calculating the lacv and lpacf. This differs from the lpacf calculated for the forecastlpacf function which assign the estimated value to the last point in the window.

The function works by using the regular R pacf applied to carefully chosen windows of the original series. Note: code in the forecastlpacf from the forecastLSW package can compute a slightly different version of the localized pacf using a wavelet method.

Value

An object of class lpacf. This is a list with the following components:

the.x	the time coordinates of the localized partial autocorrelations. Call the length of this vector n. These coordinates can be non-integer (usually at halves) even if the original times are at the integers (which is assumed by this function). You can think of these indexing the value of the localized partial autocorrelation centred at these locations.
lpacf	The localized partical autocorrelations. Matrix of dimension n x lag.max containing the lpacf for each time point at lags 1,...,lag.max.
the.vacc	This is TRUE if allpoints=TRUE and FALSE otherwise.
the.x1	If allpoints=TRUE this vector contains the x coordinate (time coordinate) values of the left-hand end of the series that are computed using increasingly reduced numbers of data points.
the.x2	As for the.x1 but for the right-hand end of the series.
vacc	A vector of length two containing the interval of time points for which the localized autocovariance is computed using the largest number of points in the calculation — ie the full binwidth.
binwidth	The smoothing binwidth that was used.
AutoBinWidth	This is TRUE if the binwidth was automatically selected, and FALSE if it was not.

Author(s)

Guy Nason and Rebecca Killick

References

Killick, R., Knight, M. I., Nason, G.P. and Eckley, I. A. (2020) The local partial autocorrelation function and some applications. *Electron. J. Statist.* **14** (2), 3268-3314. DOI: 10.1214/20-EJS1748.

See Also

lpacf.plot, forecastlpacf, plot.lacf, print.lpacf, summary.lpacf

Examples

```
# first generate a time-varying process
set.seed(1)
x=tvar2sim()

x.lpacf <- lpacf(x)

#
# There are two functions to plot lpacf class objects
#
# One is via the generic function plot which uses plot.lpacf
# the other is a bespoke function lpact.plot. We'll look at
# the generic function first which behaves similarly to the equivalent
# function in the locits package: plot.lacf
#
#
```

```
# This plot shows all of the localized partial autocovariances up to lag 27
# which is the default calculated value for this length of time series.
plot(x.lpacf)

#
# In the previous plot, maybe there were too many lags. So, let's restrict to
# five lags and colour them differently.
#
plot(x.lpacf, lags=1:5, lcol=1:5)
#
# By default, the lpacf is not computed for the whole time series range.
# Let's do it for all the points now, and replot.
#
x.lpacf.all <- lpacf(x, allpoints=TRUE)
plot(x.lpacf.all, lags=1:5, lcol=1:5)
#
#
# Suppose we wanted to look at the localized partial autocorrelation at a
# particular time point, using the regular acf-like plot. We can do this by:
#
# We will choose the time point to examine the localized pacf at as 150.
#
plot(x.lpacf, type="acf", the.time=150)

# calculate the lpacf
ans<-lpacf(x,lag.max=10,filter.number=2,family="DaubExPhase")

# then maybe plot it by lag
lpacf.plot(ans,atLag=1:10,atTime=150)
```

`lpacf.Epan` *Calculates the lpacf (local partial autocorrelation function) using an Epanechnikov kernel spectral smoother.*

Description

This calculates the local partial autocorrelation function for data `x` based on spectral smoothing using the Epanechnikov kernel. Up to `lag.max` lags are calculated.

Usage

```
lpacf.Epan(x, binwidth, lag.max=NULL, filter.number=10, family="DaubLeAsymm",
   smooth.dev=var, AutoReflect=TRUE, tol=0.1, maxits=5, ABBverbose=0, lapplyfn=lapply,
    allpoints=FALSE, verbose=FALSE, ...)
```

Arguments

`x`	The time series you wish to analyze

binwidth	The bandwidth for the spectral smoothing. If the argument is missing or zero then the bandwidth is chosen automatically using the AutoBestBW.Epan function. The smoothing is done using the Epanechnikov kernel in the Epanechnikov function.
lag.max	The maximum lag of pacf required. If this argument is NULL then the maximum lag is computed internally to be floor(10*log10(n)) where n is the length of the series. E.g. if n=512 then the lag.max will be set to 27.
filter.number	The wavelet filter number for helping choose the smoothing binwidth, used only by AutoBestBW.
family	The wavelet family for helping choose the smoothing binwidth, used only by AutoBestBW.
smooth.dev	The type of deviance used by AutoBestBW to smooth an internal wavelet spectral estimate. The usual variance var is a good option, but something like a rescaled mad could be an alternative.
AutoReflect	Mitigate periodic boundary conditions of wavelet transforms by reflecting time series about RHS end before taking transforms (and is undone before returning the answer). Setting this to be TRUE usually dramatically improves the results as the wavelet smoothing is performed using periodic transforms.
tol	Tolerance for golden section search for the best bandwidth used only by AutoBestBW.
maxits	Maximum number of iterations for the golden section search used by AutoBestBW.
ABBverbose	If nonzero prints out informative messages about the progress of the golden section search. Higher integers produce more messages. Setting it to zero suppresses messages.
lapplyfn	Function which applies a specified function to a vector. lapply is default but this argument can be used to replace this by something more efficient, e.g. a parallel version such as mclapply. Note, if you use mclapply then you should set the option mc.cores to something sensible. E.g. I have a quad-core machine, so when I am using the machine myself, alone, I set options(mc.cores=4). If you have many cores on a multicore machine, but with several users, you might want the number of codes to be used by your code to be less than the max to be nice to others.
allpoints	The lpacf is calculated using a window centered on a time point, if allpoints=TRUE then the edges of the data are also estimated (where a smaller, non-centered binwidth is used). Note, if allpoints=TRUE you obtain an estimator for the whole length of the series, but it will be more variable nearer the ends.
verbose	If TRUE then some informative messages are printed, otherwise they're not
...	Other arguments for AutoBestBW.

Details

Calculates the local partial autocorrelation function (lpacf) for 1,...,lag.max lags. See paper in the references for more details. NOTE: Often when local (windowed) estimates are created one assigns the estimated value to the central point in the window. This is the approach we take here when calculating the lacv and lpacf. This differs from the lpacf calculated for the forecastlpacf function which assign the estimated value to the last point in the window.

The function works by using the regular R pacf applied to carefully chosen windows of the original series. Note: code in the forecastlpacf from the forecastLSW package can compute a slightly different version of the localized pacf using a wavelet method.

Value

An object of class lpacf. This is a list with the following components:

the.x	the time coordinates of the localized partial autocorrelations. Call the length of this vector n. These coordinates can be non-integer (usually at halves) even if the original times are at the integers (which is assumed by this function). You can think of these indexing the value of the localized partial autocorrelation centred at these locations.
lpacf	The localized partical autocorrelations. Matrix of dimension n x lag.max containing the lpacf for each time point at lags 1,...,lag.max.
the.vacc	This is TRUE if allpoints=TRUE and FALSE otherwise.
the.x1	If allpoints=TRUE this vector contains the x coordinate (time coordinate) values of the left-hand end of the series that are computed using increasingly reduced numbers of data points.
the.x2	As for the.x1 but for the right-hand end of the series.
vacc	A vector of length two containing the interval of time points for which the localized autocovariance is computed using the largest number of points in the calculation — ie the full binwidth.
binwidth	The smoothing binwidth that was used.
AutoBinWidth	This is TRUE if the binwidth was automatically selected, and FALSE if it was not.

Author(s)

Guy Nason and Rebecca Killick

References

Killick, R., Knight, M. I., Nason, G.P. and Eckley, I. A. (2020) The local partial autocorrelation function and some applications. *Electron. J. Statist.* **14** (2), 3268-3314. DOI: 10.1214/20-EJS1748.

See Also

lpacf, lpacf.plot, forecastlpacf, plot.lacf, print.lpacf, summary.lpacf

Examples

```
# first generate a time-varying process
set.seed(1)
x=tvar2sim()

x.lpacf <- lpacf(x)
```

```
#
# There are two functions to plot lpacf class objects
#
# One is via the generic function plot which uses plot.lpacf
# the other is a bespoke function lpact.plot. We'll look at
# the generic function first which behaves similarly to the equivalent
# function in the locits package: plot.lacf
#
#
# This plot shows all of the localized partial autocovariances up to lag 27
# which is the default calculated value for this length of time series.
plot(x.lpacf)

#
# In the previous plot, maybe there were too many lags. So, let's restrict to
# five lags and colour them differently.
#
plot(x.lpacf, lags=1:5, lcol=1:5)
#
# By default, the lpacf is not computed for the whole time series range.
# Let's do it for all the points now, and replot.
#
x.lpacf.all <- lpacf(x, allpoints=TRUE)
plot(x.lpacf.all, lags=1:5, lcol=1:5)
#
#
# Suppose we wanted to look at the localized partial autocorrelation at a
# particular time point, using the regular acf-like plot. We can do this by:
#
# We will choose the time point to examine the localized pacf at as 150.
#
plot(x.lpacf, type="acf", the.time=150)

# calculate the lpacf
ans<-lpacf.Epan(x,lag.max=10,filter.number=2,family="DaubExPhase")

# then maybe plot it by lag
lpacf.plot(ans,atLag=1:10,atTime=100)
```

`lpacf.plot` *Function to produce various plots of the lpacf.*

Description

This function can produce plots of the lpacf at specified times and/or lags.

Usage

```
lpacf.plot(lpacf,atTime=NULL,atLag=NULL,SaveToFile=FALSE,alpha=0.95,...)
```

Arguments

lpacf	An object produced by lpacf.
atTime	Vector of the times of the lpacf to be plotted named according to lpacf$the.x.
atLag	Vector of the lags (columns) of the lpacf to be plotted.
SaveToFile	If large numbers of plots are needed then set SaveToFile=TRUE to save them to a file. By default this will save in the current working directory with the name Rplotlpacf%03d.pdf.
alpha	alpha level for the confidence intervals, default 95% confidence.
...	Additional arguments can be supplied which will be passed to plot.

Details

Produces the desired 1-d plots of the lpacf at times and lags as specified by atTime and atLag.

Value

Silently returns the desired 1-d plots of the lpacf at times and lags as specified by atTime and atLag. If SaveToFile=TRUE then these are saved to a file in the current working directory rather than displayed.

Author(s)

Rebecca Killick

See Also

lpacf

Examples

```
# first generate a time-varying process
set.seed(879)
x=tvar2sim()

# calculate the lpacf
ans<-lpacf(x,lag.max=10,filter.number=2,family="DaubExPhase")

# then plot it at the first 10 lags at a couple of points in the data.
lpacf.plot(ans,atLag=1:10,atTime=c(150,350))
```

plot.lpacf *Plot localized partial autocorrelation information in an* lpacf *object.*

Description

Plots information contained within a lpacf object. Plot arrangement and options are similar to that in plot.lacf in the locits package.

Usage

```
## S3 method for class 'lpacf'
plot(x, type = "line", lags = 1:min(as.integer(10 * log10(nrow(x$lpacf))),
   ncol(x$lpacf) - 1), tcex = 1, lcol = 1, llty = 1, the.time = NULL, plot.it = TRUE,
    xlab, ylab, ...)
```

Arguments

x	The lpacf class object you wish to plot.
type	The type of plot you want. This can be "line" where each partial autocorrelation is plotted as a line over time, or "persp" where the partial autocorrelation across time and lag is plotted or "acf" where a regular-style plot is drawn, like the one produced by the standard R acf function, and this is the partial autocorrelation fixed at a given point in time.
lags	A vector of integers containing the lags you wish to show. Note that, unlike regular autocorrelation, the smallest lag is lag one.
tcex	For the type="line" plot integers are displayed along the lines of their corresponding partial autocorrelations. This parameter controls the scaling of the lag numbers.
lcol	A vector of colors, the same length as the lags vector which controls the colour of each line corresponding to each partial autocorrelation line.
llty	As for lcol but the line types of each line.
the.time	A time has to be specified for the type="acf" plot, as this argument supplies it.
plot.it	If TRUE a plot is produced. If FALSE then no plot is produced but the function still executes and produces the same (invisible) output.
xlab	A label for the x-axis.
ylab	A label for the y-axis
...	Other arguments to plot.

Details

Produces a graphical representation of localized partial autocorrelation.

Value

The localized partial autocorrelation values are returned. Essentially the ones that are, or would have been, plotted are returned. The lags can be selected using the `lags` argument. A matrix is returned: each row corresponds to a time point, each column corresponds to a lag. All time points are returned. Only the lags specified in the `lags` component are returned. The `dimnames` component indicates which lags were returned.

Note

This function was adapted from `plot.lacf` from the `locits` package.

Author(s)

Guy Nason

References

Killick, R., Knight, M. I., Nason, G.P. and Eckley, I. A. (2020) The local partial autocorrelation function and some applications. *Electron. J. Statist.* **14** (2), 3268-3314. DOI: 10.1214/20-EJS1748.

See Also

`lpacf`

Examples

```
#
# Generate a test series
#
x.test <- tvar2sim()
#
# Compute its localized partial autocorrelation
#
x.lpacf <- lpacf(x.test)
#
# Perform a line plot of the localized partial autocorrelation of x
# draw the lags in colours 1 thru 5
#
#
plot(x.lpacf, lags=1:5, lcol=1:5)
#
# Now produce the same plot, but omit lag 3
#
plot(x.lpacf, lags=c(1,2,4,5), lcol=c(1,2,4,5))
#
# Now plot localized autocovariance around time 175 using the regular acf
# style plot.
#
plot(x.lpacf, type="acf", the.time=175)
```

`print.lpacf` *Prints a* `lpacf` *object.*

Description

Prints a `lpacf` object, basically telling you what's there.

Usage

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

Arguments

`x`	The `lpacf` object to print.
`...`	Other arguments

Details

Prints a `lpacf` object.

Value

None

Author(s)

Guy Nason

References

Killick, R., Knight, M. I., Nason, G.P. and Eckley, I. A. (2020) The local partial autocorrelation function and some applications. *Electron. J. Statist.* **14** (2), 3268-3314. DOI: 10.1214/20-EJS1748.

See Also

`lpacf`, `summary.lpacf`

Examples

```
#
# Simulate an example
#
x.test <- tvar2sim()
#
# Compute the lpacf
#
x.lpacf <- lpacf(x.test)
#
```

```
# Print it out - note, can normally just type name of object
#
print(x.lpacf)
#Class 'lpacf' : Localized Partial Autocorrelation Object:
#       ~~~~  : List with 5 components with names
#               the.x lpacf the.vacc binwidth AutoBinWidth
#
#
#summary(.):
#----------
#Number of times:  220
#Number of lags:  27
#Range of times from:  147  to  366
#Part series was analyzed (alltimes=FALSE)
#Smoothing binwidth used was:  293
#        Binwidth was chosen automatically
```

`summary.lpacf` *Print out summary information about a* `lpacf` *object.*

Description

Print out summary information about a `lpacf` object.

Usage

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

Arguments

`object`	The `lpacf` object you want to print out summary info for.
`...`	Other arguments

Details

Prints out number of times that we have localized partial autocorrelation for and the number of lags computed. The localized partial autocorrelation is computed at a number of time points that might not be identical to the times in the original series, this function prints out the min and max of the range of times and an indicator of whether the whole series' localized pacf was computed. The bandwidth associated with spectral smoothing is printed and a note made of whether it was computed automatically or supplied as an earlier argument.

Value

None

Author(s)

Guy Nason

References

Killick, R., Knight, M. I., Nason, G.P. and Eckley, I. A. (2020) The local partial autocorrelation function and some applications. *Electron. J. Statist.* **14** (2), 3268-3314. DOI: 10.1214/20-EJS1748.

See Also

`lpacf`, `print.lacf`

Examples

```
#
# See example for print.lacf
#
```

`tvar2sim` *Simulate a realization from a particular TVAR(2) model.*

Description

Simulates a realization from a TVAR(2) model where both parameters move from -1.1 to 0.5 in equal steps over 512 time points. The realization is of length 512. The innovations are normally distributed with mean zero and standard deviation of `sd`.

Usage

```
tvar2sim(sd = 1)
```

Arguments

`sd`	This is the standard deviation of the Gaussian innovation.

Details

This function is easily converted into one that does the same thing but for a different sample size.

Value

A realization of the aforementioned TVAR(2) process.

Author(s)

Guy Nason.

See Also

lpacf

Examples

```
#
# Generate realization from the TVAR(2) process
#
x <- tvar2sim()
#
# Maybe plot it
#
ts.plot(x)
```

Index