Package 'PoMoS'

July 2, 2014

Type Package

Title Polynomial (ordinary differential equation) Model Search

Version 1.1.1

Date 2010-10-10

Author Mangiarotti S., Coudret R., Drapeau L.

Depends R (>= 2.10), RGtk2 (>= 2.12.18), cairoDevice (>= 2.0), igraph (>= 0.5.4)

Maintainer Drapeau L. <laurent.drapeau@ird.fr>

Description PoMoS-package aims to determine from a set of N time series the optimal polynomial structure of a model built on first-order ordinary differential equations. The core of the package is based on the poMoS function: an evolutionary algorithm combined with a least square fitting. Optimality is estimated with AIC (Akaike, 1974) or AIC-like criterions. Although efficient in its selection, the identification of the optimal structure cannot be guaranteed. Therefore, both selected and rejected models are reconsidered after optimal solutions are obtained from the evolutionary algorithm for another analysis. This analysis is based on a statistical evaluation of the regressors' quality.

SystemRequirements Gtk+ (>= 2.16.0)

Lazyload no

License CeCILL-2

Repository CRAN

Date/Publication 2011-02-11 10:05:44

NeedsCompilation no

R topics documented:

`bpRegress` *Boxplot representation of models criterion.*

Description

Creates boxplots of the criterions, sorted by regressors, obtained for the whole set of tested models. This function is used by `poMoS` in graphical mode.

Usage

```
bpRegress(models,labels=NULL,plot=TRUE,modByNReg=1,logPlot=FALSE,nVar=1,groupReg=NULL)
```

Arguments

`models`	A list of elements providing information about tested models such as their criterion or their structure. A detailed description of this list can be found in the documentation of `poMoS`
`labels`	Text to be shown below each column of the graphic. As an example, it can be the output of labelPoly `labelPoly`
`plot`	A boolean which indicate if boxplots must be represented graphically or not.
`modByNReg`	When a graphical output is required and if this type of model exists, `bpRegress` also draws black points to highlight models that contain every regressors but the one of the column. Models with the regressor of the column plus a fixed number of others can also be shown (with color points); the maximum number of regressors to be represented by points can be interactively fixed. Thus, to be able to see the criterion of models with only the regressor of the column, those with it and one other, and those with it and two others, the user should enter `modByNReg=3`
`logPlot`	Determine if criteria must be shown as they are or in logarithm scale. It is usefull to choose the last one if data are packed at the bottom of the graphic.

nVar	In order to separate regressors of different variables by a line, nVar should be set to 1. The columns of boxplots are then divided into nVar groups of equal size. If there is not the same number of regressors for each variable, use groupeReg.
groupReg	The vector of cumulated number of regressors for each variable. It can also be defined as the vector of the choosen boxplots indexes to draw separation lines on their right. It is related to the codenVar parameter, but has priority over it.

Value

A list with one element by regressor, which is a vector of criteria of the models within this regressor is included.

Note

Boxplot colours depend on the relative size of the sample plotted. The larger data length, the darker, the smaller data length, the lighter.

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

See Also

poMoS, labelPoly

Examples

```
#Sprott K attractor loading
data(sprottK,package="PoMoS")
sprottK <- as.matrix(sprottK)

# We only select regressors found in 'labelPoly' example for the first variable
filter <- c(FALSE, rep(TRUE,4), rep(FALSE,2), rep(TRUE,3))
filterTot <- cbind(filter, rep(TRUE,10))
labelsForBp <- c(labelPoly(3,2,toFind=filter),labelPoly(3,2))

#We search for the best model and then draw boxplots
res <- poMoS(sprottK,200,20,20,10,nModStart=100,filterToExpl=c(1,2),show=0,dMax=2,filterReg=filterTot,critCalc=
toPlot <- list(def=res$rejectedMod$def[as.vector(filterTot),],
   crit=res$rejectedMod$crit,
   coeff=res$rejectedMod$coeff[as.vector(filterTot),])
trash <- bpRegress(toPlot,labels=labelsForBp,groupReg=sum(filter))
```

convGraph *Graphical plot of neighboring models*

Description

Draws models as points and links them if their structure is almost the same. The nearness of models connected can be set. This function can be launched graphically with poMoS.

Usage

```
convGraph(data,titleText=gettext("New graph"), subTitleText="", adj="remAdd",limit=1)
```

Arguments

data	A matrix in which informations to be represented are stored. Each column corrépsonds to a model. The first rows follow the same pattern as the def element of the list presented in the note of the documentation of poMoS. The last one contain criteria.
titleText	The title of the graph.
subTitleText	The subtitle of the graph.
adj	A string to choose the distance that determines neighbourhood relations. With "remAdd", the distance refers to the number of regressors we have to add or remove to transform a model into another. With "permut", the distance refers to the minimum number of permutation. A permutation is defined as the action of remove a regressor and then add another one. As a result, when adj="permut", two models of different sizes have an infinite distance.
limit	Available if adj="remAdd", only. Sets the maximum distance at which models are linked together. Anyway, models separated with small distances will appear closer from each other, because the representation of the relations depends on distances.

Value

A list of two elements :

graph	An graph object as it is defined in the igraph package. It contains vertices that represents models and edges which are links between them.
layout	For the user to understand which models are close to the others, the function calls an iterative algorithm to display them the most clearly it is possible. This object sends the models position in a 2D space after this optimisation.

Note

The graph produced shows coloured points. The red ones are those with lowest criterion, which are consquently the optimum ones. As for the colour yellow, it indicates highest criterion. The best model is shown by a big point.

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

See Also

poMoS

Examples

```
#Rossler attractor loading
data(rossler,package="PoMoS")
rossler <- as.matrix(rossler)

#We try to explain the third series
res <- poMoS(rossler,100,20,50,5,nModStart=100,filterToExpl=3,dMax=3,show=0,critCalc="modif")
toPlot <- rbind(res$rejectedMod$def,res$rejectedMod$crit)
trash <- convGraph(toPlot,"Tested models graph","Maximum Distance : 3",limit=3)
```

---

findRegData *Information searching from poMoS output*

Description

Find a model structure from a list of results produced by poMoS

Usage

```
findRegData(models, toFind, neigh=FALSE)
```

Arguments

models	A list of information as provided by poMoS function. See the documentation of this function for more information.
toFind	A vector of model definition, as long as there are regressors to take into consideration, with 1 if the regressor is used and 0 otherwise. To see which element of the vector corresponds to which regressor, labelPoly can be used.
neigh	The maximal distance between the model to be found toFind and the obtained from the model search. To enter FALSE is equivalent to use 0 for this distance. That means that only the informations concerning toFind is wanted. The distance is defined as the minimum number of modification to be applied to a model (which structure is defined thanks to a vector) to obtain the other.

Value

A list of information like poMoS output but containing the selected models, only.

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

See Also

poMoS.

Examples

```
data(sprottK,package="PoMoS")
res <- poMoS(as.matrix(sprottK),100,20,4,3,nModStart=15,filterToExpl=3,dt=1/20,dMax=1,show=0,critCalc="modif")
findRegData(res$rejectedMod,c(0,1,1,0))
```

---

labelPoly *Polynomial labels systematic ordering in a set of N variables*

Description

Generate a systematic correspondence of polynomial terms for a polynomial of degree dMax with nVar variables, and displays the terms in a formal way. This correspondence is used in the package poMoS

Usage

```
labelPoly(nVar, dMax, toFind=NULL)
```

Arguments

nVar	The number of time series from which the polynomial series are built.
dMax	The maximum degree of the polynomial function. The same dMax must be provided to poMoS when the functions are used together.
toFind	It is allowed to select only one or several polynomial terms among those returned automatically. This selection can be achieved by using toFind that filters the results before the end of the function. It can be a vector of subscripts or booleans : TRUE for the selected labels and FALSE elsewhere. It is usefull if someone only wants the regressors of a model and is not interested in all the tested ones.

Value

A vector of characters. Each element is the expression of a regressor such as $X_1^2 X_3 X_4$

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

See Also

poMoS

Examples

```
data(sprottK,package="PoMoS")
res <- poMoS(as.matrix(sprottK),200,20,5,4,nModStart=200,filterToExpl=1,dMax=2,show=0,critCalc="modif")
testedMod <- list()
testedMod$def <- res$selectedMod$def[,!is.na(res$selectedMod$crit)]
testedMod$crit <- res$selectedMod$crit[!is.na(res$selectedMod$crit)]
bestModDef <- unique(testedMod$def[,testedMod$crit == min(testedMod$crit)],MARGIN=2)
labelPoly(3,2,toFind=bestModDef==1)
#Note : The data in 'x' represent a Sprott H attractor. The returned regressors may not be
#the initial ones because this use of poMoS gives a lot of false positives.
#For more accuracy, this function should be launched in graphical mode, with boxplots.
#In our tests we find with this method :
# "X3 " "X3^2 " "X2 " "X2 X3 " "X1 X3 " "X1 X2 " "X1^2 "
#And the initial regressors are :
# "X2 " "X1 X3 "
```

---

lorenz *Time series from Lorenz (1963) attractor*

Description

This dataset contains simulated variables of the differential equations of the Lorenz (1963) attractor, that is:
$dx/dt = s(y - x)$
$dy/dt = x(r - z) - y$
$dz/dt = xy - bz$

Usage

```
lorenz
```

Format

A data frame with 10000 observations on 3 variables. The columns correspond to x, y and z, respectively.

Details

The integration was made thanks to the ordinary differential equations solver of `deSolve` package. The following parameter values were used:

s 10

r 28

b $8/3$

Besides, the time between two points of the attractor is given by :

dt 0.005

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

References

Lorenz E.N. (1963). Deterministic Nonperiodic Flow. Journal of the atmospheric sciences, 20, pp. 130141.

Examples

```
#Selecting a single side of the Lorenz attractor
data(lorenz,package="PoMoS")
k <- kmeans(lorenz,2)
plot(lorenz[k$cluster==1,1],lorenz[k$cluster==1,2])
```

NDVI *Normalized Difference Vegetation Index (NDVI) time series*

Description

Time series of Normalized Difference Vegetation Index (NDVI) derived from the AVHRR (Advanced Very High Resolution Radiometer) sensors of the NOAA (National Oceanic and Atmospheric Administration) space missions. These values have been spatially averaged over the province of Settat, Morocco (see hereafter for details).

Usage

```
NDVI
```

Format

A data frame of 4 variables with 9618 interpolated values each. The first column represents the vegetation index while the others are the successive derivatives.

Details

NDVI data from AVHRR sensor are available since 1981. The present time series is derived from the GIMMS product of the Global Land Cover Facility. This product is 10-day sampling with an 8x8km2 resolution. The present time series has been spatially averaged over the province of Settat (Morocco), representing 150 pixels of 8x8km2. The time series has been smoothed and resampled at a higher time resolution (roughly daily sampling) with a Savitzky-Golay method.

`dt 1/365`

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

Source

Global Land Cover Facility
University of Maryland, Department of Geography
http://www.landcover.org

References

Pinzon, J., Brown, M.E. and Tucker, C.J., 2005.
Satellite time series correction of orbital drift artifacts using empirical mode decomposition.
In: N. Huang (Editor), Hilbert-Huang Transform: Introduction and Applications, pp. 167-186.

Tucker C.J., Pinzon J.E., Brown M.E., Slayback D.A., Pak E.W., Mahoney R., Vermote E.F. and Saleous N.E., 2005.
An extended AVHRR 8-km NDVI dataset compatible with MODIS and SPOT vegetation NDVI data.
International Journal of Remote Sensing, 26:20, 4485-4498. http://dx.doi.org/10.1080/01431160500168686

Examples

```
#2D plots like those drawn by the gloMo function of the GloMo package
data(NDVI,package="PoMoS")
par(mfrow=c(2,2))
for (i in 1:2) {
for (j in (i+1):3) {
plot(NDVI[,i],NDVI[,j],type="l",main="NDVI Series",
xlab=paste("X",i,sep=""),ylab=paste("X",j,sep=""))
}
}
```

`polyFilter` *Polynomial terms with the exponents of the variables*

Description

Generate a systematic correspondence of polynomial terms for a polynomial of degree `dMax` with `nVar` variables, and store it in a matrix form. Each column corresponds to one algebraic term; each line gives the exponent of the variables.

Usage

```
polyFilter(nVar,dMax)
```

Arguments

`nVar`	The number of variables for which polynomial terms are built
`dMax`	The maximum degree of the polynomial terms. The same `dMax` value must be provided to `poMoS`

Value

A matrix of degrees. Each column corresponds to one polynomial term. Each line represents the degrees of the variables. The polynomial term is the product each variable contribution: for example, the column c(1,2,0,2) leads to the polynomial term $X_1 X_2^2 X_4^2$

Note

This function can be used in complement to `labelPoly` to formally display the polynomial terms. Each expression produced by `labelPoly` corresponds to a vector of degree of polyFilter.

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

See Also

poMoS labelPoly

Examples

```
#Two different ways to represent the 8th polynomial series
dMax <- 4
nVar <- 3
p <- polyFilter(nVar,dMax)
p[,8]
labelPoly(nVar,dMax,8)
```

`PoMoS` *Polynomial (ordinary differential equation) Model Search*

Description

This package aims to determine from a set of N time series the optimal polynomial structure of a model built on first-order ordinary differential equations. The core of the package is based on the poMoS function: an evolutionary algorithm combined with a least square fitting. Optimality is estimated with AIC (Akaike, 1974) or AIC-like criterions. Although efficient in its selection, the identification of the optimal structure cannot be guaranteed. Therefore, both selected and rejected models are reconsidered after optimal solutions are obtained from the evolutionary algorithm for another analysis. This analysis is based on a statistical evaluation of the regressors quality. An important level of handling can be kept during the model search process: regressors can thus be added and removed dynamically, combining both the evolutionary model search with the statistical analysis of the regressors. Parameters of the evolutionary algorithm can also be modified during the searching process etc. A blind run of the algorithm is also made technically possible.

Details

Package:	PoMoS
Type:	Package
Version:	1.1
Date:	2010-10-10
Depends:	R (>= 2.11.1), RGtk2, cairoDevice, igraph
SystemRequirements:	Gtk+ (>= 2.16.0)
License:	CeCILL-2
LazyLoad:	yes

Index:

The main function is `PoMoS` from which major part of the other functions are called. Gtk+ and RGtk2 provide a link between `PoMoS` and R and allows for an interactive monitoring of the algorithm search process. Several criterions can be chosen to evaluate the models skill, including Akaike's and the BIC (Bayesian Information Criterion). These criterions are based on the sum of two terms, one term corresponding to the likelihood, another term corresponding to a penalty on larger length models. `bpRegress` and `convGraph` are representation functions. `findRegData` helps in browsing the results obtained from `poMoS` algorithm. `labelPoly`, `polyFilter` and `seriesToPoly` manage the polynomial series and their incorporation into models. These functions are also required in the GloMo-package which, therefore, depends on PoMoS-package.

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

Maintainer: Drapeau L., : laurent.drapeau@ird.fr

References

Akaike, Hirotugu (1974). "A new look at the statistical model identification". IEEE Transactions on Automatic Control 19 (6): 716-723.

See Also

poMoS

poMoS *Polynomial Model Search*

Description

poMoS function aims at determining, from a set of N time series, the optimal polynomial structure for a model built on first-order ordinary differential equations. An evolutionary algorithm combined with a least square fitting is used for this purpose. The optimality is estimated with AIC or AIC-like criterions. Methodology: A simple evolutionary algorithm is used to generate sets of models structures and to test their efficiency to model the time series provided in input. Models are firstly identified through a least square fitting (the `lsfit` function is used for this purpose). Their performance is thus estimated with AIC-like criterions.

Usage

```
poMoS(series, nIter, samplFreq, nModAdd, nModSelec, nModStart=NULL,
dt=1, show=1, toAnalyse=NULL, filterToExpl=NULL, filterReg=NULL, dMax=1,
initSingle=F, initLog=NULL, critCalc="aic")
```

Arguments

`series`	The matrix of time series data. Each column contains a time series.
`nIter`	An integer that determines the maximum number of iterations for the algorithm. The algorithm stops earlier if in the pool of models `nModSelec` is greater than the number of models available for test.
`samplFreq`	Subsampling frequency of the input series. If `samplFreq` is too small, there can be noise in the analysis that involves false positives in the regressors selection. If it is too big, the effect of the useful regressors is reduced and may become undetectable.
`nModAdd`	How many models the algorithm adds to the pool at the end of each iteration. In practice, the number of added number can be less than `nModAdd` if not enough satisfying models are available.
`nModSelec`	Determines how many models are compared during each iteration. Afterwards it eliminates the `nModSelec-1` worst models from the pool of models to test.
`nModStart`	Initial number of models in the pool of models to be tested.
`dt`	Time sampling of the input series. Used for the estimation of the regression coefficients.
`show`	The depth of the interaction between the user and the function. It can be one of the following values : **0: Silent mode. No user interface, no messages.** **1: Normal mode. User interface, a few messages.** **2: Debug mode. User interface, a lot of messages.**

`toAnalyse` Defines the more general model's structures to be considered in the analysis. These models are automatically included in the testing pool when starting the algorithm. This structure is given by a matrix: each column corresponds to an equation, each line corresponds to an algebraic regressor. Value 1 indicates that the regressor is selected; value 0 indicates that it is not. The correspondence between lines and regressors can be obtained from labelPoly providing the number nVar of time series and the maximum degree`dMax` of the polynomial.

`filterToExpl` A vector setting that defines which series modeling to be analyzed. For example, with filterToExpl=c(2,3), the modeling of the second and third time series is considered. The derivatives of the time series 2 and 3 is thus computed and if no restriction is given (filterReg=NULL), the analysis of these two series is carried on with the whole set of polynomial regressors. If the user does not fill this field, all the time series are analyzed.

`filterReg` A matrix that defines which regressors to be considered in the analysis. Each column corresponds to one series as defined in filterToExpl; the number of columns and the length of filterToExpl must thus be identical. Each line corresponds to one algebraic regressor. Value is 1 if the regressor is available, 0 if it is not. In order to ensure an iterative use of the algorithm, note that the same format is used for the output variable def. To set filterReg=NULL is equivalent to setting a matrix full of 1.

`dMax` The maximum degree of the polynomials. For example, let call x_1, x_2 and x_3 our 3 times series. The regressor x_1 x_2^3 x_3^2 is not part of the models if dMax = 6.

`initSingle` A boolean that indicates if the pool of models has to be initialized by all the models with only one regressor. If `initSingle=T`, `nModStart` does not have to be set.

`initLog` It can be interesting for the user to keep a log of regressors removed after several launches of `poMoS`. To do so , he or she can set `initLog` to the previous version of the element log of the list returned by the last execution of `poMoS`.

`critCalc` A word that defines which criteria to consider when estimating the models skill. Criteria "aic" and "bic" are available, as well as "modif". The latter is better adapted for deterministic series with non gaussian residuals arising from the derivative approximation. It can be called by setting `critCalc="modif"`.

Value

A list of five elements :

`rejected` Information about rejected models.

`selected` Information about both selected and untested models.

`dMax` The value of dMax at the end of the run. It can be different from the input value if it was interactively changed by the user through the interface.

`filterReg` The value of `filterReg` at the end of the run. It can be different from the input value for the same reason as for `dMax`.

`log` The labels of the regressors removed thanks to the interactive interface during the run.

Note

The two first elemente of the returned list are lists that contain :

`def` The output models structure. Each column corresponds to one selected model. Lines inform about the presence (value 1) of the absence (value 0) of the corresponding regressors, as in `toAnalyse`.

`crit` A vector defining the models skill. The lower the value, the better the models skill. If crit==NA the model was not tested by the function.

`coeff` The values of the regression coefficients. A matrix which has the same size as variable def and where the value of the regression coefficient is given when def==1 whereas NA is given when def==0. More information about these coefficients can be found in the lsfit documentation

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

See Also

labelPoly lsfit

Examples

```
#Example 1
data(NDVI,package="PoMoS")
reg <- poMoS(as.matrix(NDVI)[,1:3],1000,20,20,10,nModStart=1000,dMax=3,show=0,filterToExpl=3)

        #Example 1
data(rossler,package="PoMoS")
reg <- poMoS(as.matrix(rossler),100,20,50,40,nModStart=100,dMax=2,show=0,filterToExpl=NULL)
```

---

rossler *Time series from Rossler (1976) attractor*

Description

This dataset contains simulated variables of the differential equations of the Rossler (1976) attractor, that is:
$dx/dt = -y - z$
$dy/dt = x + ay$
$dz/dt = b + z(x - c)$

Usage

```
rossler
```

Format

A data frame with 199999 observations on 3 variables. The columns correspond to x, y and z, respectively.

Details

The integration was made with a Runge-Kutta 4 algorithm. The following parameter values were used:

a 0.52

b 2

c 4

Besides, the time between two points of the attractor is given by :

dt 0.05

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

References

Rossler O. E. (1976). An equation for contiuous chaos. Physics Letters, 57A(5), pp. 397398.

Examples

```
#Computation of the derivative of z
data(rossler,package="PoMoS")
N <- dim(rossler)[1]
dt <- 0.05
derivRossZ <- (rossler[2:N,3] - rossler[1:(N-1),3])/ dt

#Linear regression between dz/dt and 1, z, x z
linRegResults <- lsfit(cbind(rossler[1:(N-1),3],rossler[1:(N-1),3]*rossler[1:(N-1),1]),
   derivRossZ)
linRegResults$coefficients
#Our results were 2.1242690, -4.0691719, 0.9951174, which is very close to the parameters
#2, -4 and 1 entered in the initial system
```

---

seriesToPoly *Polynomial series construction*

Description

Creates time series corresponding to polynomial terms by multiplying time series among them. One time series can be multiplied several times.

Usage

```
seriesToPoly(series, dMax=NULL, pFilter=NULL)
```

Arguments

series	A matrix containing the time series from which the polynomial series are built. One time series by column.
dMax	The maximum degree of the polynomial. The same value must be used in poMoS
pFilter	A matrix filled representing one polynomial term in each column and the power of each variable in each line (same structure as the output of polyFilter). If this parameter is not set, polyFilter is called, instead. In this case, dMax is required, whereas nVar is deduced from the size of series.

Value

A matrix which have as number of lines as series and which comprehend the polynomial series measures based on it. Each column is relative to one series. To know the expression of a series, use labelPoly.

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

See Also

polyFilter labelPoly

Examples

```
#Both examples give the first values of series 'X_1 X_2^3 X_3'
data(sprottK,package="PoMoS")
sprottK <- as.matrix(sprottK)
dMax <- 5
nVar <- dim(sprottK)[2]

#Example 1
polySeries <- seriesToPoly(sprottK,dMax)
lab <- labelPoly(nVar,dMax)
polySeries[1:100,lab=="X1 X2^3 X3 "]

#Example 2
p <- c(1,3,1)
polySeries <- seriesToPoly(sprottK,pFilter=p)
polySeries[1:100]
```

sprottK *Time series from Sprott K (1994) attractor*

Description

This dataset contains simulated variables of the differential equations of the SprottK attractor, that is: :
$dx/dt = -ay + xz$
$dy/dt = x + by$
$dz/dt = x - z$

Usage

```
sprottK
```

Format

A data frame with 199999 observations on 3 variables. The columns correspond to x, y and z, respectively.

Details

The integration was made with a Runge-Kutta 4 algorithm. The following parameter values were used:

a 0.432

b 0.2

Besides, the time step between two points of the attractor is given by:

dt 0.05

Author(s)

Mangiarotti S., Coudret R., Drapeau L.

References

Sprott J.C., 1994. Some simple chaotic flows. Physical Review E, 50(2), pp. 647-650.

Examples

```
#Computation of the derivative of x
data(sprottK,package="PoMoS")
N <- dim(sprottK)[1]
dt <- 0.05
derivSproX <- (sprottK[2:N,1] - sprottK[1:(N-1),1])/ dt

#Linear regression between dx/dt and a model with regressors 1, x and y
```

```
#which does not corresponds to the initial equation
linRegResults <- lsfit(cbind(sprottK[1:(N-1),1],sprottK[1:(N-1),2]),derivSproX)
normalizedRes <- (linRegResults$residuals - mean(linRegResults$residuals))/sqrt(var(linRegResults$residuals))
ks.test(normalizedRes,"pnorm")
#We reject the normality test, so we cannot use criterion like AIC to find
#the good model, for these series.
```

Index