

Package ‘FitAR’

February 9, 2012

Type Package

Title Subset AR Model Fitting

Version 1.92

Date 2011-05-16

Author A.I. McLeod, Ying Zhang and Changjiang Xu

Maintainer A.I. McLeod <aimcleod@uwo.ca>

Depends R (>= 2.0.0), lattice, leaps, ltsa, bestglm

Description Comprehensive model building function for identification, estimation and diagnostic checking for AR and subset AR models.

Two types of subset AR models are supported. One family of subset AR models, denoted by AR_p, is formed by taking subset of the original AR coefficients and in the other, denoted by AR_z, subsets of the partial autocorrelations are used. The main advantage of the AR_z model is its applicability to very large order models.

Imports lattice, leaps, ltsa, bestglm

LazyLoad yes

LazyData yes

Classification/ACM G.3, G.4, I.5.1

Classification/MSC 62M10, 91B84

License GPL (>= 2)

URL <http://www.stats.uwo.ca/faculty/aim>

Repository CRAN

Date/Publication 2011-05-18 04:53:55

R topics documented:

| | |
|---------------------------|----|
| FitAR-package | 4 |
| AcfPlot | 9 |
| AR1Est | 11 |
| ARSdf | 12 |
| ARToMA | 13 |
| ARToPacf | 14 |
| BackcastResidualsAR | 15 |
| BICqLL | 16 |
| Boot | 18 |
| Boot.FitAR | 19 |
| Boot.ts | 20 |
| BoxCox | 22 |
| BoxCox.Arima | 23 |
| BoxCox.FitAR | 24 |
| BoxCox.numeric | 26 |
| BoxCox.ts | 27 |
| bxcx | 29 |
| Caffeine | 30 |
| ChampernowneD | 31 |
| coef.FitAR | 32 |
| Commodities | 33 |
| cts | 34 |
| DetAR | 35 |
| FastLoglikelihoodAR | 36 |
| FitAR | 37 |
| FitARp | 40 |
| FitARz | 43 |
| fitted.FitAR | 45 |
| FromSymmetricStorageUpper | 46 |
| FXRates | 46 |
| Get1G | 47 |
| GetARMeanMLE | 48 |
| GetB | 49 |
| GetFitAR | 49 |
| GetFitARpLS | 51 |
| GetFitARpMLE | 52 |
| GetFitARz | 54 |
| GetKappa | 55 |
| GetLeapsAR | 55 |
| getRho | 58 |
| getT | 58 |
| glog | 59 |
| InformationMatrixAR | 60 |
| InformationMatrixARp | 61 |
| InformationMatrixARz | 62 |
| InvertibleQ | 63 |

| | |
|------------------------------|-----|
| Jacobian | 64 |
| JacobianK | 65 |
| JarqueBeraTest | 66 |
| LBQPlot | 67 |
| LjungBoxTest | 68 |
| LoglikelihoodAR | 69 |
| Ninemile | 71 |
| PacfDL | 71 |
| PacfPlot | 73 |
| PacfToAR | 74 |
| plot.FitAR | 75 |
| plot.Selectmodel | 76 |
| PlotARSdf | 77 |
| predict.FitAR | 78 |
| print.FitAR | 80 |
| RacfPlot | 81 |
| Readts | 82 |
| residuals.FitAR | 83 |
| sdfplot | 84 |
| sdfplot.ar | 85 |
| sdfplot.Arima | 86 |
| sdfplot.FitAR | 86 |
| sdfplot.numeric | 87 |
| sdfplot.ts | 88 |
| SelectModel | 89 |
| SeriesA | 91 |
| SeriesB | 92 |
| SeriesB2 | 93 |
| SiddiquiMatrix | 93 |
| SimulateGaussianAR | 94 |
| summary.FitAR | 95 |
| TacvfAR | 96 |
| TacvfMA | 97 |
| TimeSeriesPlot | 98 |
| UnitRootTest | 100 |
| USTobacco | 101 |
| VarianceRacfAR | 102 |
| VarianceRacfARp | 103 |
| VarianceRacfARz | 104 |
| Willamette | 105 |

| | |
|---------------|----------------------------------------------------------------------------------------------|
| FitAR-package | <i>Fits AR and subset AR models and provides complete model building capabilities. FitAR</i> |
|---------------|----------------------------------------------------------------------------------------------|

Description

For model estimation the main function is FitAR for which generic methods print, summary, coef, plot and predict are implemented. For model identification, there is a new PacfPlot for subset ARz identification. Subset models may also be selected using AIC, BIC and UBIC criteria with the function SelectModel. SelectModel produces a S3 class object, "SelectModel", for which there is a plot method. The main fitting function is FitAR. New methods and generic functions, BoxCox, Boot and sdfplot are given. Methods for print, summary, coef, residuals, fitted and predict implemented.

Details

| | |
|-----------|------------|
| Package: | FitAR |
| Type: | Package |
| Version: | 1.92 |
| Date: | 2011-05-16 |
| License: | GPL (>= 2) |
| LazyLoad: | yes |
| LazyData: | yes |

To get started please see the documentation and examples given in the functions PacfPlot, SelectModel and FitAR.

R functions for model diagnostic checking, simulation and forecasting are also available. The function plot provides many graphical diagnostic plots.

Model Selection: [TimeSeriesPlot](#), [PacfPlot](#), [SelectModel](#)

Model Estimation: [FitAR](#), [AR1Est](#)

Model Checking: [plot.FitAR](#), [BoxCox](#), [LBQPlot](#), [RacfPlot](#), [JarqueBeraTest](#),

Model Applications: [Boot](#), [SimulateGaussianAR](#)

Methods Functions: [coef](#), [fitted](#), [predict](#), [print](#), [summary](#), [residuals](#)

Useful Utility Functions: [Readts](#), [cts](#)

New Generic and Methods Functions: [Boot](#), [BoxCox](#), [sdfplot](#)

Author(s)

A. I. McLeod and Ying Zhang

Maintainer: aimcleod@uwo.ca

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

McLeod, A.I. and Zhang, Y. (2008a). Faster ARMA Maximum Likelihood Estimation, *Computational Statistics and Data Analysis* 52-4, 2166-2176. DOI link: <http://dx.doi.org/10.1016/j.csda.2007.07.020>.

McLeod, A.I. and Zhang, Y. (2008b). Improved Subset Autoregression: With R Package. *Journal of Statistical Software*.

Changjiang Xu and A. I. McLeod (2010). Bayesian information criterion with Bernoulli prior. Submitted for publication.

Changjiang Xu and A. I. McLeod (2010). Model selection using generalized information criterion. Submitted for publication.

Examples

#Scripts are given below for all Figures and Tables in McLeod and Zhang (2008b).

#

#Figure 1. Plot of lynx time series using plot.ts
plot(lynx)

#Figure 2. Plot of lynx series using TimeSeriesPlot
TimeSeriesPlot(lynx, type="o", pch=16, ylab="# pelts", main="Lynx Trappings")

#Figure 3. Trellis plot for Ninemile series
graphics.off() #clear previous graphics
data(Ninemile)
print(TimeSeriesPlot(Ninemile, SubLength=200))

#Figure 4. Partial autocorrelation plot of lynx series
graphics.off() #clear previous graphics
PacfPlot(log(lynx))

Not run: #takes some time for all these examples
#Figure 5. Using SelectModel to select the best subset ARz or ARp and
comparing BIC and UBIC subset selection.

graphics.off() #clear previous graphics
layout(matrix(1:4,ncol=2),respect=TRUE)
ansBICp<-SelectModel(log(lynx),lag.max=15,Criterion="BIC", ARModel="ARp", Best=3)
ansUBICp<-SelectModel(log(lynx),lag.max=15, ARModel="ARp", Best=3)
ansBICz<-SelectModel(log(lynx),lag.max=15,Criterion="BIC", ARModel="ARz", Best=3)
ansUBICz<-SelectModel(log(lynx),lag.max=15, ARModel="ARz", Best=3)
par(mfg=c(1,1))
plot(ansBICp)
par(mfg=c(2,1))
plot(ansUBICp)
par(mfg=c(1,2))
plot(ansBICz)
par(mfg=c(2,2))

```

plot(ansUBICz)

#Figure 6. Logged spectral density function fitted to square-root of monthly
#       sunspot series using the non-subset AR and subset ARz.
#       AIC and BIC are used for the AR while BIC and UBIC are used
#       for the ARz. Takes about 115 seconds on 3.6 GHz Pentium PC.
graphics.off() #clear previous graphics
layout(matrix(1:4,ncol=2),respect=TRUE)
z<-sqrt(sunspots)
P<-200
pAIC<-SelectModel(z, lag.max=P, ARModel="AR", Best=1, Criterion="AIC")
ARAIC<-FitAR(z, pAIC)
par(mfg=c(1,1))
sdfplot(ARAIC)
title(main="AIC Order Selection")
pBIC<-SelectModel(z, lag.max=P, ARModel="AR", Best=1, Criterion="BIC")
ARBIC<-FitAR(z, pBIC)
par(mfg=c(1,2))
sdfplot(ARBIC)
title(main="BIC Order Selection")
SunspotMonthARzBIC<-SelectModel(z,lag.max=P, ARModel="ARz", Best=1, Criterion="BIC")
ARzBIC<-FitAR(z, SunspotMonthARzBIC)
par(mfg=c(2,1))
sdfplot(ARzBIC)
title(main="BIC Subset Selection")
SunspotMonthARzUBIC<-SelectModel(z,lag.max=P, ARModel="ARz", Best=1)
ARzUBIC<-FitAR(z, SunspotMonthARzUBIC)
par(mfg=c(2,2))
sdfplot(ARzUBIC)
title(main="UBIC Subset Selection")

#Table 3.
#First part of table: AR(1) and AR(2).
#Only timings for GetFitAR and FitAR since the R function ar produces too many
# warnings and an error message as noted in McLeod and Zhang (2008b, p.12).
#The ar function with mle option is not recommended.
start.time<-proc.time()
set.seed(661177723)
NREP<-100 #takes about 156 sec
NREP<-10 #takes about 16 sec
ns<-c(50,100,200,500,1000)
ps<-c(1,2) #AR(p), p=1,2
tmsA<-matrix(numeric(4*length(ns)*length(ps)),ncol=4)
ICOUNT<-0
for (IP in 1:length(ps)){
  p<-ps[IP]
  for (ISIM in 1:length(ns)){
    ICOUNT<-ICOUNT+1
    n<-ns[ISIM]
    ptm <- proc.time()
    for (i in 1:NREP){
      phi<-PacfToAR(runif(p, min=-1, max =1))
      z<-SimulateGaussianAR(phi,n)

```

```

        phiHat<-try(GetFitAR(z,p,MeanValue=mean(z))$phiHat)
    }
    t1<-(proc.time() - ptm)[1]
#
    ptm <- proc.time()
    for (i in 1:NREP){
        phi<-PacfToAR(runif(p, min=-1, max =1))
        z<-SimulateGaussianAR(phi,n)
        phiHat<-try(FitAR(z,p,MeanMLEQ=TRUE)$phiHat)
    }
    t2<-(proc.time() - ptm)[1]
#
    ptm <- proc.time()
    for (i in 1:NREP){
        phi<-PacfToAR(runif(p, min=-1, max =1))
        z<-SimulateGaussianAR(phi,n)
        #uncomment this line and next two lines for ar timings -- expect lots of
        # warnings and an error message!!
        #phiHat<-try(ar(z,aic=FALSE,order.max=p,method="mle")$ar)
        #delete this line and the next one
        phiHat<-NA
    }
    #uncomment this line for ar timings
    #t3<-(proc.time() - ptm)[1]
    t3<-NA #delete this line for ar timings

    tmsA[ICOUNT,]<-c(n,t1,t2,t3)
}
}
rnames<-c(rep("AR(1)", length(ns)),rep("AR(2)", length(ns)) )
cnames<-c("n", "GetFitAR", "FitAR", "ar")
dimnames(tmsA)<-list(rnames,cnames)
tmsA[,-1]<-round(tmsA[,-1]/NREP,2)
end.time<-proc.time()
total.time<-(end.time-start.time)[1]

#Second part of table: AR(20) and AR(40).
#NOTE: ar is not recommended with method="mle" produces numerous warnings
# and also takes a long time!
    start.time<-proc.time()
    set.seed(66117723)
    NREP<-100 #takes 7.5 hours
    NREP<-10 #takes 45 minutes
    ns<-c(1000,2000,5000)
    ps<-c(20,40)
    tmsB<-matrix(numeric(4*length(ns)*length(ps)),ncol=4)
    ICOUNT<-0
    for (IP in 1:length(ps)){
        p<-ps[IP]
        phi<-PacfToAR(0.8/(1:p))
        for (ISIM in 1:length(ns)){
            ICOUNT<-ICOUNT+1
            n<-ns[ISIM]

```

```

    ptm <- proc.time()
    for (i in 1:NREP){
      z<-SimulateGaussianAR(phi,n)
      phiHat<-try(GetFitAR(z,p,MeanValue=mean(z))$phiHat)
    }
    t1<-(proc.time() - ptm)[1]
    ptm <- proc.time()
    for (i in 1:NREP){
      z<-SimulateGaussianAR(phi,n)
      phiHat<-try(FitAR(z,p,MeanMLEQ=TRUE)$phiHat)
    }
    t2<-(proc.time() - ptm)[1]
    ptm <- proc.time()
    for (i in 1:NREP){
      z<-SimulateGaussianAR(phi,n)
      phiHat<-try(ar(z,aic=FALSE,order.max=p,method="mle")$ar)
    }
    t3<-(proc.time() - ptm)[1]
    tmsB[ICOUNT,]<-c(n,t1,t2,t3)
  }
}
rnames<-c( rep("AR(20)", length(ns)), rep("AR(40)", length(ns)) )
cnames<-c("n", "GetFitAR", "FitAR", "ar")
dimnames(tmsB)<-list(rnames,cnames)
tmsB[,-1] <- round(tmsB[,-1]/NREP,2)
end.time<-proc.time()
total.time<-(end.time-start.time)[1]

#Figure 7. Comparing Box-Cox analyses using FitAR and MASS
library(MASS)
graphics.off() #clear previous graphics
layout(matrix(c(1,2,1,2),ncol=2))
pvec<-c(1,2,4,10,11)
out<-FitAR(lynx, ARModel="ARp", pvec)
BoxCox(out)
PMAx<-max(pvec)
Xy <- embed(lynx, PMAx + 1)
y <- Xy[, 1]
X <- (Xy[, -1])[, pvec] #pvec != 1
outlm<-lm(y~X)
boxcox(outlm,lambdaseq(0.0,0.6,0.05))

#Figure 8
graphics.off() #clear previous graphics
BoxCox(AirPassengers) #takes about 30 sec

#Figure 9
graphics.off() #clear previous graphics
data(rivers)
BoxCox(rivers)
title(sub="Length of 141 North American Rivers")

#Figure 10

```

```

graphics.off() #clear previous graphics
data(USTobacco)
TimeSeriesPlot(USTobacco, aspect=1)

#Figure 11
graphics.off() #clear previous graphics
data(USTobacco)
outUST<-arima(USTobacco, c(0,1,1))
BoxCox(outUST)

#Figure 12. Basic diagnostic plots for ARp fitted to the log lynx series
graphics.off() #clear previous graphics
out<-FitAR(log(lynx), ARModel="ARp", c(1,2,4,10,11))
plot(out, terse=TRUE)

#Figure 13. RSF plot for ARp fitted to log lynx series
graphics.off() #clear previous graphics
out<-FitAR(log(lynx), ARModel="ARp", c(1,2,4,10,11))
rfs(out)

#Table 6. Comparison of bootstrap and large-sample sd
#Use bootstrap to compute standard errors of parameters
#takes about 34 seconds on a 3.6 GHz PC
ptm <- proc.time() #user time
set.seed(2491781) #for reproducibility
R<-100 #number of bootstrap iterations
p<-c(1,2,4,7,10,11)
ans<-FitAR(log(lynx),p)
out<-Boot(ans, R)
fn<-function(z) FitAR(z,p)$zetaHat
sdBoot<-sqrt(diag(var(t(apply(out,fn,MARGIN=2))))))
sdLargeSample<-coef(ans)[,2][1:6]
sd<-matrix(c(sdBoot,sdLargeSample),ncol=2)
dimnames(sd)<-list(names(sdLargeSample),c("Bootstrap","LargeSample"))
ptm<-(proc.time()-ptm)[1]
sd

## End(Not run)

```

Description

Produces theoretical correlation plot

Usage

```
AcfPlot(g, LagZeroQ= TRUE, ylab=NULL, main=NULL, ...)
```

Arguments

| | |
|-----------------------|----------------------------------------------------|
| <code>g</code> | vector of autocorrelations at lags 1,...,length(g) |
| <code>LagZeroQ</code> | start plot at lag zero with <code>g[0]=1</code> |
| <code>ylab</code> | vertical axis label |
| <code>main</code> | plot title |
| <code>...</code> | optional graphical parameters |

Value

No value. Plot is produced via plot function.

Author(s)

A.I. McLeod and Y. Zhang

See Also

[acf](#)

Examples

```
#
#Simple example, plot acf for AR(1)
phi<-0.8
maxLag<-20
g<-phi^(1:maxLag)
AcfPlot(g)
AcfPlot(g, LagZeroQ=FALSE)
#
# Plot the sample inverse partial autocorrelations.
# On the basis of this plot, Cleveland (1972) suggested an ARp(1,2,7)
# for this data
"InverseAcf" <-
function(z, p=15){
g<-TacfMA(GetFitARpLS(z-mean(z),1:p)$phiHat, lag.max=p)
g/g[1]
}
#
data(SeriesA)
AcfPlot(InverseAcf(SeriesA),LagZeroQ=FALSE)
```

| | |
|--------|----------------------------------|
| AR1Est | <i>Exact MLE Mean-Zero AR(1)</i> |
|--------|----------------------------------|

Description

This function is used by GetFitAR in the AR(1) case. It is a fast exact solution using the root of a cubic equation.

Usage

```
AR1Est(z, MeanValue = 0)
```

Arguments

| | |
|-----------|-----------------------|
| z | time series or vector |
| MeanValue | known mean |

Details

The exact MLE for mean-zero AR(1) satisfies a cubic equation. The solution of this equation for the MLE given by Zhang (2002) is used. This approach is more reliable as well as faster than the usual approach to the exact MLE using a numerical optimization technique which can occasionally have convergence problems.

Value

MLE for the parameter

Author(s)

A.I. McLeod and Y. Zhang

References

Zhang, Y. (2002). Topics in Autoregression, Ph.D. Thesis, University of Western Ontario.

See Also

[GetFitARz](#)

Examples

```
AR1Est(lynx-mean(lynx))
```

ARSdf

Autoregressive Spectral Density Function

Description

Spectral density function of AR(p) is computed.

Usage

```
ARSdf(phi, pFFT = 8)
```

Arguments

| | |
|------|-------------------------------------------|
| phi | AR Coefficient vector |
| pFFT | FFT with $2^p FFF$ frequencies, default 8 |

Details

The Fast Fourier Transform (FFT) is used to compute the spectral density function.

Value

A vector of the density function values, $(f(1), \dots, f(2^p FFF))$

Author(s)

A.I. McLeod and Y. Zhang

See Also

[spectrum](#), [spec.pgram](#), [spec.ar](#)

Examples

```
ARSdf(0.8)
ARSdf(c(0.1, 0.2))
```

Description

A stationary-causal AR(p) can be written as a general linear process (GLP). This function obtains the moving-average expansion out to the L-th lag, $z[t] = a[t] + \text{psi}[1]*a[t-1] + \dots + \text{psi}[L]*a[t-L]$.

Usage

```
ARToMA(phi, lag.max)
```

Arguments

| | |
|---------|-----------------------|
| phi | AR Coefficient vector |
| lag.max | maximum lag |

Details

The coefficients are computed recursively as indicated in Box and Jenkins (1970).

Value

Vector of length L+1 containing, (1,psi[1],...,psi[L])

Author(s)

A.I. McLeod and Y. Zhang

References

Box and Jenkins (1970), Time Series Analysis, Forecasting & Control

See Also

[InvertibleQ](#)

Examples

```
ARToMA(0.5, 20)  
ARToMA(c(0.2, 0.5), 15)
```

ARToPacf

Reparametrize AR Coefficients In Terms of PACF

Description

Transform AR parameter coefficients into partial autocorrelation function (PACF).

Usage

```
ARToPacf(phi)
```

Arguments

phi vector of AR parameter coefficients

Details

For details see McLeod and Zhang (2006).

Value

Vector of length(phi) containing the parameters in the transformed PACF domain

Warning

No check for invertibility is done for maximum computational efficiency since this function is used extensively in the numerical optimization of the AR loglikelihood function in FitAR. Use `InvertibleQ` to test for invertible AR coefficients.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[InvertibleQ](#), [PacfToAR](#)

Examples

```
somePACF<-c(0.5,0.6,0.7,0.8,-0.9,-0.8)
#PacfToAR() transforms PACF to AR parameter coefficients.
someAR<-PacfToAR(somePACF)
test<-ARToPacf(someAR)
#This should be very small
sum(abs(test-somePACF))
```

BackcastResidualsAR *Innovation Residuals in AR*

Description

Obtains the residuals (estimated innovations). The residuals for $t=1, \dots, p$ are obtained using the backforecasting algorithm of Box and Jenkins (1970).

Usage

```
BackcastResidualsAR(y, phi, Q = 100, demean=TRUE)
```

Arguments

| | |
|--------|-----------------------------------------------------|
| y | a time series or vector |
| phi | AR coefficients, lags 1,...,p |
| Q | for backcasting, the AR is approximated by an MA(Q) |
| demean | subtract sample mean |

Details

The backforecasting algorithm is described in detail in the book of Box and Jenkins (1970). The idea is to compute the expected value of the innovation assuming a high-order MA(q).

Value

Vector of residuals

Note

No check is done that the AR is causal-stationary.

Author(s)

A.I. McLeod and Y. Zhang

References

Box and Jenkins (1970). Time Series Analysis: Forecasting and Control.

See Also

[InvertibleQ](#), [FitAR](#)

Examples

```
#compare residuals obtained using backcasting with fitted parameters and
# the residuals extracted from output of FitAR. They are identical.
p<-11
out<-FitAR(log(lynx), p)
phi<-out$phiHat #fitted parameters
resphi<-BackcastResidualsAR(log(lynx), phi)
sum(abs(resphi-resid(out)))
```

BICqLL

Select best model using BICq

Description

Given the loglikelihoods for a set of models arranged in ascending order of size, the best model is selected using the BICq criterion for a specified size.

Usage

```
BICqLL(logL, n, level = 0.99, mSize = 1:length(logL), mComplex = function(k) k)
```

Arguments

| | |
|----------|--------------------------|
| logL | vector of loglikelihoods |
| n | sample size |
| level | probability |
| mSize | model sizes |
| mComplex | a complexity function |

Details

See reference

Value

| | |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| khat | dataframe with columns: k, a.1, a.2 q.1, q.2, level=level, where k is the optimal model, (a.1,a.2) is the interval for alpha in the GIC, (q.1, q.2) is the interval for q and level is the probability. Each row corresponds to an entry in 'level'. |
| table | This table indicates which models can be selected for some values of alpha or q. |

Note

AIC corresponds to setting level=0.84. BIC corresponds to setting level=pchisq(log(n), 1). So for n=100, 1000; BIC=0.96, 0.97

Author(s)

Changjiang Xu and A. Ian McLeod

References

Changjiang Xu and A. Ian McLeod (2010). Bayesian information criterion with Bernoulli prior. Submitted for publication.

See Also

[SelectModel](#)

Examples

```
#Example 1.
#AR(p) Order selection for 'lynx' series
z <- log(lynx)
n <- length(z)
lag.max <- 20
zta<-ARToPacf(ar.burg(z,aic=FALSE,order.max=lag.max)$ar)
LagsEntering<-1:lag.max
LLapprox <- (-n)*log(cumprod(1-zta[LagsEntering]^2))
ans<-BICqLL(logL=LLapprox, n=n, level=c(0.9, 0.95, 0.99))
ans$khat
ans$table
#if we just want the best model for level=0.99 then,
(BICqLL(logL=LLapprox, n=n, level=0.99)$khat)[[1]]
#aic for comparison
aic<-(-2*LLapprox)+2*LagsEntering
which.min(aic)
plot(LagsEntering, aic)
#
#Example 2. AR(p) Order Selection
#white noise. We do NumRep simulations and
# count the number of overfit models.
set.seed(231789) #make reproducible
n <- 100
lag.max <- 30
LagsEntering<-0:lag.max
NumRep<-25
level<-c(0.99, 0.95, 0.9)
k<-numeric(length(level))
for (i in 1:NumRep){
  z <- rnorm(n)
  zta<-ARToPacf(ar.burg(z,aic=FALSE,order.max=lag.max)$ar)
  LLapprox <- c(0,(-n)*log(cumprod(1-zta[LagsEntering]^2)))
  k<-k+as.numeric(0<(BICqLL(logL=LLapprox, n=n, level=level,mSize=LagsEntering)$khat)[,1])
}
ans<-k
names(ans)<-level
ans
#
```

```

#Example 3. AR(p) best subset. ARz Family.
z <- log(lynx)
n <- length(z)
lag.max <- 20
zta <- ARToPacf(ar.burg(z,aic=FALSE,order.max=lag.max)$ar)
LagsEntering <- order(abs(zta),decreasing=TRUE)
LLapprox <- c(0, -n*log(cumprod(1-zta[LagsEntering]^2)))
kHat <- (BICqLL(logL=LLapprox, n=n, level=0.99)$khat)[[1]]
pvec<-LagsEntering[1:kHat]
pvec
#pvec above shows the lags in order of importance
#
#Example 4. AR(p) best subset. ARp Family.
#could also try z <- sunspot.year
z <- log(lynx)
lag.max <- 15
pvec <- 1:lag.max
n <- length(z)-lag.max
ind <- (lag.max+1):length(z)
y<-z[ind]
X<-matrix(rep(0,n*lag.max), nrow=n, ncol=lag.max)
for (i in 1:lag.max)
  X[,i] <- z[ind-pvec[i]]
outLeaps <- leaps(y=y,x=X,nbest=1,method="r2",strictly.compatible=FALSE)
# approximate likelihood approach
TotSS <- sum((y-mean(y))^2)
RSS <- TotSS*(1-outLeaps$r2)
LogL <- (-n/2)*log(c(TotSS/n, RSS/n))#null model included
ans<-BICqLL(logL=LogL, n=n, level=0.99)
kHat <- (ans$khat)[[1]]-1 #kHat=0 is null model
pvec <-0
if (kHat > 0)
  pvec <- (1:lag.max)[(outLeaps$which)[kHat,]]
pvec

```

 Boot

Generic Bootstrap Function

Description

Generic function to bootstrap a fitted model.

Usage

```
Boot(obj, R=1, ...)
```

Arguments

| | |
|-----|--------------------------------|
| obj | fitted object |
| R | number of bootstrap replicates |
| ... | optional arguments |

Details

At present, the only function implemented is [Boot.FitAR](#).

Value

Parametric bootstrap simulation

Author(s)

A.I. McLeod and Y. Zhang

See Also

[Boot.FitAR](#)

Examples

```
out<-FitAR(SeriesA, c(1,2,7), ARModel="ARp")
Boot(out)
```

Boot.FitAR

Simulate a Fitted AR

Description

Simulate a realization from a fitted AR model. This is useful in the parametric bootstrap. Generic function for "Boot" method.

Usage

```
## S3 method for class 'FitAR'
Boot(obj, R=1, ...)
```

Arguments

| | |
|-----|----------------------------------|
| obj | the output from FitAR |
| R | number of bootstrap replications |
| ... | optional arguments |

Value

A simulated time series with the same length as the original fitted time series is produced when $R=1$. When $R>1$, a matrix with R columns is produced with each column a separate bootstrap realization.

Author(s)

A.I. McLeod and Y. Zhang

See Also

[Boot SimulateGaussianAR](#)

Examples

```
#Plot log(lynx) time series and simulation
#
ans <- FitAR(log(lynx), 8)
z<-Boot.FitAR(ans)
par(mfrow=c(2,1))
TimeSeriesPlot(log(lynx))
title(main="log(lynx) time series")
TimeSeriesPlot(z)
title(main="Simulated AR(8), fitted to log lynx")
#par(mfrow=c(1,1))
#
#Use bootstrap to compute standard errors of parameters
#takes about 18 seconds on a 3.6 GHz PC

## Not run:
ptm <- proc.time() #user time
R<-100 #number of bootstrap iterations
p<-c(1,2,4,7,10,11)
ans<-FitAR(log(lynx),p)
out<-Boot(ans, R)
fn<-function(z) GetFitARz(z,p)$zetaHat
sdBoot<-sqrt(diag(var(t(apply(out,fn,MARGIN=2))))))
sdLargeSample<-coef(ans)[,2][1:6]
sd<-matrix(c(sdBoot,sdLargeSample),ncol=2)
dimnames(sd)<-list(names(sdLargeSample),c("Bootstrap", "LargeSample"))
ptm<-(proc.time()-ptm)[1]
sd

## End(Not run)
```

Description

An AR(p) model is fit to the time series using the AIC and then it is simulated.

Usage

```
## S3 method for class 'ts'  
Boot(obj, R=1, ...)
```

Arguments

| | |
|-----|--------------------------------|
| obj | a time series, class "ts" |
| R | number of bootstrap replicates |
| ... | optional arguments |

Value

A time series or vector.

Note

Parametric and nonparametric time series bootstraps are discussed by Davison and Hinkley (1997, Ch.8.2).

Author(s)

A.I. McLeod and Y. Zhang

References

Davison, A.C. and Hinkley, D.V. (1997), *Bootstrap Methods and Their Application*. Cambridge University Press.

See Also

[Boot.FitAR](#). Nonparametric bootstrap for time series is available in the function `tsboot` in the library `boot`.

Examples

```
layout(matrix(c(1,2,1,2),ncol=2))  
TimeSeriesPlot(SeriesA)  
TimeSeriesPlot(Boot(SeriesA),main="Bootstrap of Series A")
```

`BoxCox`*Generic Box-Cox Analysis Function*

Description

The function is implemented as a generic function with methods for classes "FitAR", "Arima", "ts" and "numeric".

For $\lambda \neq 0$, the Box-Cox transformation is of x is $(x^\lambda - 1)/\lambda$. If the minimum data value is ≤ 0 , a small positive constant, equal to the negative of the minimum plus 0.25, is added to all the data values.

Usage

```
BoxCox(object, ...)
```

Arguments

| | |
|---------------------|--------------------|
| <code>object</code> | model object |
| <code>...</code> | optional arguments |

Value

No value returned. Graphical output is produced as side-effect. The plot shows relative likelihood function as well as the MLE and a confidence interval.

Note

The MASS package has a similar function `boxcox` but this is implemented only for regression and analysis of variance.

Author(s)

A.I. McLeod and Y. Zhang

See Also

[BoxCox.Arima](#), [BoxCox.FitAR](#), [BoxCox.ts](#), [BoxCox.numeric](#)

Examples

```
## Not run: #takes a few seconds
BoxCox(lynx)
out<-FitAR(lynx, c(1,2,4,10,11), ARModel="ARp", MLEQ=FALSE)
BoxCox(out)
out<-FitAR(lynx, c(1,2,4,5,7,8,10,11,12))
BoxCox(out)

## End(Not run)
```

Description

Implements Box-Cox analysis for "Arima" class objects, the output from `arima`, a R built-in function. Variance change in time series is an important topic. In some cases using a Box-Cox transformation will provide a much simpler analysis than the much more complex ARMA-GARCH approach. See US Tobacco series example given below for an example.

Usage

```
## S3 method for class 'Arima'
BoxCox(object, interval = c(-1, 1), type = "BoxCox", InitLambda = "none", ...)
```

Arguments

| | |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <code>object</code> | output from <code>arima</code> , a R built-in function |
| <code>interval</code> | interval to be searched for the optimal transformation |
| <code>type</code> | ignored unless, <code>InitLambda!="none"</code> . Type of transformation, default is "Box-Cox". Otherwise a simple power transformation. |
| <code>InitLambda</code> | default "none". Otherwise a numerical value giving the transformation parameter. |
| <code>...</code> | optional arguments passed to <code>optimize</code> |

Details

If no transformation is used on the data, then the original data is used. But if a transformation has already been used, we need to inverse transform the data to recover the untransformed data.

For $\lambda \neq 0$, the Box-Cox transformation is of x is $(x^\lambda - 1)/\lambda$. If the minimum data value is ≤ 0 , a small positive constant, equal to the negative of the minimum plus 0.25, is added to all the data values.

The log of the Jacobian is $(\lambda - 1) \sum_{t=D+1}^n \log(z_t)$, where λ is the transformation, $n=\text{length}(z)$, z is the vector of data and $D = d + ds*s$, where d is the degree of regular differencing, ds is the degree of seasonal differencing and s is the seasonal period. The correct expression for the loglikelihood function was first given in Hipel and McLeod (1977, eqn. 10). Using the wrong expression for the Jacobian has a disastrous effect in many situations. For example with the international airline passenger time series, the MLE for λ would be about 1.958 instead of close to zero.

If the minimum data value is ≤ 0 , a small positive constant, equal to the negative of the minimum plus 0.25, is added to all the data values.

Value

No value returned. Graphical output is produced as side-effect. The plot shows relative likelihood function as well as the MLE and a confidence interval.

Note

The MASS package has a similar function `boxcox` but this is implemented only for regression and analysis of variance.

Author(s)

A.I. McLeod and Y. Zhang

References

Hipel, K.W. and McLeod, A.I. (1977). Advances in Box-Jenkins Modelling. Part 1, Model Construction. Water Resources Research 13, 567-575.

See Also

[arima](#), [BoxCox](#), [BoxCox.FitAR](#)

Examples

```
## Not run: #not run to save time!
#Tobacco Production
plot(USTobacco)
USTobacco.arima<-arima(USTobacco,order=c(0,1,1))
BoxCox(USTobacco.arima)
#
air.arima<-arima(AirPassengers, c(0,1,1), seasonal=list(order=c(0,1,1), period=12))
BoxCox(air.arima)
#
#In this example, we fit a model to the square-root of the sunspots and
#back transform in BoxCox.
sqrtsun.arima<-arima(sqrt(sunspot.year),c(2,0,0))
BoxCox(sqrtsun.arima, InitLambda=0.5, type="power")
#
#Back transform with AirPassengers
Garima<-arima(log(AirPassengers), c(0,1,1), seasonal=list(order=c(0,1,1),period=12))
BoxCox(Garima, InitLambda=0)

## End(Not run)
```

BoxCox.FitAR

Box-Cox Analysis for "FitAR" Objects

Description

This is a methods function to do a Box-Cox analysis for models fit using FitAR.

Usage

```
## S3 method for class 'FitAR'
BoxCox(object, interval = c(-1, 1), type = "BoxCox", InitLambda = "none", ...)
```

Arguments

| | |
|------------|----------------------------------------------------------------------------------------------------------------------------|
| object | output from FitAR |
| interval | interval to be searched for the optimal transformation |
| type | Ignored unless, InitLambda!="none". Type of transformation, default is "Box-Cox". Otherwise a simple power transformation. |
| InitLambda | default "none". Otherwise a numerical value giving the transformation parameter. |
| ... | optional arguments passed to optimize |

Details

If no transformation is used on the data, then the original data is used. But if a transformation has already been used, we need to inverse transform the data to recover the untransformed data.

For $\lambda \neq 0$, the Box-Cox transformation is of x is $(x^\lambda - 1)/\lambda$. If the minimum data value is ≤ 0 , a small positive constant, equal to the negative of the minimum plus 0.25, is added to all the data values.

Value

No value returned. Graphical output produced as side-effect. The plot shows relative likelihood function as well as the MLE and a confidence interval.

Note

The MASS package has a similar function `boxcox` but this is implemented only for regression and analysis of variance.

Author(s)

A.I. McLeod

References

- Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *Journal of Royal Statistical Society, Series B*, vol. 26, pp. 211-246.
- McLeod, A.I. and Zhang, Y. (2006a). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.
- McLeod, A.I. and Zhang, Y. (2006b, under review). Subset Autoregression Modelling. *Journal of Statistical Software*.

See Also

[BoxCox](#), [BoxCox.Arima](#)

Examples

```
## Not run: #takes a few seconds
#lynx time series. ARp subset model.
out<-FitAR(lynx, c(1,2,4,10,11), ARModel="ARp")
BoxCox(out)
#
#lynx time series. ARz subset model.
p<-SelectModel(lynx, ARModel="ARz", lag.max=25, Best=1)
out<-FitAR(lynx, p)
BoxCox(out)

## End(Not run)
```

BoxCox.numeric

Box-Cox Analysis for a Time Series

Description

An AR(p) model is selected using AIC and then the best Box-Cox transformation is determined. Requires package FitAR.

Usage

```
## S3 method for class 'numeric'
BoxCox(object, interval = c(-1, 1), IIDQ = FALSE, ...)
```

Arguments

| | |
|----------|---------------------------------------------------------------------------------------|
| object | a vector of time series values |
| interval | interval to be searched |
| IIDQ | If true, IID is assumed, ie. p=0. If FALSE, AR(p) is fit with p determined using AIC. |
| ... | optional arguments |

Details

For $\lambda \neq 0$, the Box-Cox transformation is of x is $(x^\lambda - 1)/\lambda$.

If the minimum data value is ≤ 0 , a small positive constant, equal to the negative of the minimum plus 0.25, is added to all the data values. If $\text{length}(\text{object}) < 20$, no AR model is used, that is, p=0.

Value

No value returned. Graphical output produced as side-effect. The plot shows relative likelihood function as well as the MLE and a confidence interval.

Note

The MASS package has a similar function `boxcox` but this is implemented only for regression and analysis of variance.

Author(s)

A.I. McLeod and Y. Zhang

References

Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *Journal of Royal Statistical Society, Series B*, vol. 26, pp. 211-246.

See Also

[BoxCox.FitAR](#), [BoxCox.Arima](#), [BoxCox.ts](#)

Examples

```
## Not run: #takes a few seconds
#annual sunspot series
BoxCox(sunspot.year, IIDQ=FALSE)
#
#non-time series example, lengths of rivers
BoxCox(rivers)

## End(Not run)
```

BoxCox.ts

Box-Cox Analysis for a Time Series

Description

The time series is converted to a vector and `BoxCox.numeric` is used.

Usage

```
## S3 method for class 'ts'
BoxCox(object, interval = c(-1, 1), ...)
```

Arguments

| | |
|-----------------------|--------------------------------|
| <code>object</code> | a vector of time series values |
| <code>interval</code> | interval to be searched |
| <code>...</code> | optional arguments |

Details

For $\lambda \neq 0$, the Box-Cox transformation is of x is $(x^\lambda - 1)/\lambda$. If the minimum data value is ≤ 0 , a small positive constant, equal to the negative of the minimum plus 0.25, is added to all the data values.

Value

No value returned. Graphical output produced as side-effect. The plot shows relative likelihood function as well as the MLE and a confidence interval.

Warning

It is important not to transform the data when fitting it with AR since the optimal transformation would be found for the transformed data – not the original data. Normally this would not be a sensible thing to do.

Note

The MASS package has a similar function `boxcox` but this is implemented only for regression and analysis of variance.

Author(s)

A.I. McLeod

References

Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *Journal of Royal Statistical Society, Series B*, vol. 26, pp. 211-246.

See Also

[BoxCox.FitAR](#), [BoxCox.Arima](#), [BoxCox.numeric](#)

Examples

```
#  
## Not run: #takes a few seconds  
BoxCox(sunspot.year)  
  
## End(Not run)
```

bxcx

*Box-Cox Transformation and its Inverse***Description**

Box-Cox or power transformation or its inverse. For $\lambda \neq 0$, the Box-Cox transformation of x is $(x^\lambda - 1)/\lambda$, whereas the regular power transformation is simply x^λ . When $\lambda = 0$, it is log in both cases. The inverse of the Box-Cox and the power transform can also be obtained.

Usage

```
bxcx(x, lambda, InverseQ = FALSE, type = "BoxCox")
```

Arguments

| | |
|-----------------------|---------------------------------------------|
| <code>x</code> | a vector or time series |
| <code>lambda</code> | power transformation parameter |
| <code>InverseQ</code> | if TRUE, the inverse transformation is done |
| <code>type</code> | either "BoxCox" or "power" |

Value

A vector or time series of the transformed data

Author(s)

A.I. McLeod

References

Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *Journal of Royal Statistical Society, Series B*, vol. 26, pp. 211-246.

See Also

[BoxCox](#)

Examples

```
#lambda=0.5
z<-AirPassengers; lambda<-0.5
y<-bxcx(z, lambda)
z2<-bxcx(y, lambda, InverseQ=TRUE)
sum(abs(z2-z))
#
z<-AirPassengers; lambda<-0.0
y<-bxcx(z, lambda)
z2<-bxcx(y, lambda, InverseQ=TRUE)
sum(abs(z2-z))
```

Caffeine

*Caffeine industrial time series***Description**

Hamilton and Watts (1978) state this series is produced from a cyclic industrial process with a period of 5.

Usage

```
data(Caffeine)
```

Format

The format is: num [1:178] 0.429 0.443 0.451 0.455 0.44 0.433 0.423 0.412 0.411 0.426 ...

Details

The dataset are from the paper by Hamilton and Watts (1978, Table 1). The series is used to illustrate how a multiplicative seasonal ARMA model may be identified using the partial autocorrelations. Chatfield (1979) argues that the inverse autocorrelations are more effective for model identification with this example.

Source

Hamilton, David C. and Watts, Donald G. (1978). Interpreting Partial Autocorrelation Functions of Seasonal Time Series Models. *Biometrika* 65/1, 135-140.

References

Hamilton, David C. and Watts, Donald G. (1978). Interpreting Partial Autocorrelation Functions of Seasonal Time Series Models. *Biometrika* 65/1, 135-140.

Chatfield, C. (1979). Inverse Autocorrelations. *Journal of the Royal Statistical Society. Series A (General)* 142/3, 363–377.

Examples

```
#Example 1
sdfplot(Caffeine)
TimeSeriesPlot(Caffeine)
#
#Example 2
a<-numeric(3)
names(a)<-c("AIC", "BIC", paste(sep=" ", "BIC(q=", paste(sep=" ", c(0.85), ")"))))
z<-Caffeine
lag.max <- ceiling(length(z)/4)
a[1]<-SelectModel(z, lag.max=lag.max, ARModel="AR", Best=1, Criterion="AIC")
a[2]<-SelectModel(z, lag.max=lag.max, ARModel="AR", Best=1, Criterion="BIC")
```

```
a[3]<-SelectModel(z, lag.max=lag.max, ARModel="AR", Best=1, Criterion="BICq", t=0.85)
a
```

ChampernowneD

Champernowne Matrix

Description

Computes sufficient statistics for AR.

Usage

```
ChampernowneD(z, p, MeanZero = FALSE)
```

Arguments

| | |
|----------|------------------------------------------------------------------------------------------------------------------------------------------|
| z | time series data |
| p | order of the AR |
| MeanZero | Assume mean is zero. Default is FALSE so the sample mean is subtracted from the data first. Otherwise no sample mean correction is made. |

Details

This matrix is defined in McLeod & Zhang (2006).

Value

The matrix D defined following eqn. (3) of McLeod & Zhang (2006) is computed.

Note

This function is used by GetFitAR. It may be used to compute the exact loglikelihood for an AR.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[GetFitARz](#), [FastLoglikelihoodAR](#), [FitAR](#)

Examples

```

#compute the exact concentrated loglikelihood function, (McLeod & Zhang, 2006, eq.(6)),
# for AR(p) fitted by Yule-Walker to logged lynx data
#
p<-8
CD<-ChampernowneD(log(lynx), p)
n<-length(lynx)
phi<-ar(log(lynx), order.max=p, aic=FALSE, method="yule-walker")$ar
LoglYW<-FastLoglikelihoodAR(phi,n,CD)
phi<-ar(log(lynx), order.max=p, aic=FALSE, method="burg")$ar
LoglBurg<-FastLoglikelihoodAR(phi,n,CD)
phi<-ar(log(lynx), order.max=p, aic=FALSE, method="ols")$ar
LoglOLS<-FastLoglikelihoodAR(phi,n,CD)
phi<-ar(log(lynx), order.max=p, aic=FALSE, method="mle")$ar
LoglMLE<-FastLoglikelihoodAR(phi,n,CD)
ans<-c(LoglYW,LoglBurg,LoglOLS,LoglMLE)
names(ans)<-c("YW", "Burg", "OLS", "MLE")
ans
#compare the MLE result given by ar with that given by FitAR
FitAR(log(lynx),p)

```

coef.FitAR

*Display Estimated Parameters from Output of "FitAR"***Description**

Method function to display fitted parameters, their standard errors and Z-ratio for AR models fit with FitAR.

Usage

```

## S3 method for class 'FitAR'
coef(object, ...)

```

Arguments

| | |
|--------|---------------------------|
| object | obj the output from FitAR |
| ... | optional parameters |

Value

A matrix is returned. The columns of the matrix are labeled MLE, sd and Z-ratio. The rows labels indicate the AR coefficients which were estimated followed by mu, the estimate of mean.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

Examples

```
# Fit subset AR to SeriesA
outA<-FitAR(SeriesA, c(1,2,7), ARModel="ARz")
coef(outA)
#
outALS<-FitAR(SeriesA, c(1,2,7), ARModel="ARp")
coef(outALS)
```

 Commodities

Commodity prices

Description

Commodity prices on successive business days, Chicago Exchange These data exhibit classic random walk behavior.

Usage

```
data(Commodities)
```

Format

The format is: List of 5 \$ gold:'data.frame': 97 obs. of 3 variables: ..\$ close: num [1:97] 700 671 680 677 690\$ high : num [1:97] 714 698 683 682 692\$ low : num [1:97] 700 669 664 676 684 ... \$ feed:'data.frame': 95 obs. of 3 variables: ..\$ close: num [1:95] 79 79 78.6 79.9 79.3\$ high : num [1:95] 80 79.5 79.2 79.9 79.8\$ low : num [1:95] 79 78.5 78.6 78.8 79.3 ... \$ port:'data.frame': 99 obs. of 3 variables: ..\$ close: num [1:99] 57.7 56.8 57.5 57 59\$ high : num [1:99] 59.9 57.5 58 58.1 59\$ low : num [1:99] 57.2 56.4 55.1 56.8 56.4 ... \$ soy : 'data.frame': 99 obs. of 3 variables: ..\$ close: num [1:99] 766 790 804 794 824\$ high : num [1:99] 788 791 805 808 824\$ low : num [1:99] 764 764 778 792 809 ... \$ us : 'data.frame': 100 obs. of 3 variables: ..\$ close: num [1:100] 91.6 91.6 91.4 91.4 91.2\$ high : num [1:100] 91.9 91.7 91.6 91.4 91.5\$ low : num [1:100] 91.6 91.5 91.3 91.3 91.1 ...

Details

Data from 1981. feed: April; gold: June, pork: March, us: March

Source

I obtained these data from a broker.

Examples

```
dim(Commodities$gold)
dimnames(Commodities$gold)[[2]]
TimeSeriesPlot(Commodities$gold$close)
```

cts

Concatenate Time Series

Description

Creating a ts object by concatenating y onto x, where x is a ts object and y is a vector. If y is a ts object, it is simply converted to a vector and then concatenated on to x and the tsp attribute of y is ignored.

Usage

```
cts(x, y)
```

Arguments

| | |
|---|----------------------------------------|
| x | a time series, a ts object |
| y | a vector which is concatenated on to x |

Details

If y is a ts object, it is first converted to a vector. Then the vector is concatenated on to the time series x. An error is given if x is not a ts object.

Value

A time series which starts at start(x) and has length equal to length(x)+length(y).

Warning

Only two arguments are allowed, otherwise an error message will be given.

Note

The package zoo may also be used to concatenate time series, as in this example,

```
x <- ts(1:3) y <- ts(4:5, start = 4) z <- ts(6:7, start = 7) library("zoo") as.ts(c(as.zooreg(x), y, z))
```

Author(s)

A.I. McLeod

See Also

[ts](#), [start](#), [window](#) [as.zooreg](#)

Examples

```
#Example 1
#Compare cts and c
#In the current version of R (2.6), they produce different results
z1<-window(lynx,end=1900)
z2<-window(lynx,start=1901)
z<-cts(z1,z2)
y<-c(z1,z2)

#See also Example 2 in predict.FitAR documentation

#Example 3.
#Note tsp attribute of second argument is ignored but a warning is given if it is present
# and not aligned with first argument's attribute.
x <- ts(1:3)
z <- ts(6:7, start = 7)
cts(x,z) #warning given
y <- ts(4:5, start = 4)
cts(x,y) #no warning needed in this example.
```

DetAR

Covariance Determinant of AR(p)

Description

Computes the covariance determinant of p successive observations from an AR(p) process with unit innovation variance.

Usage

```
DetAR(phi)
```

Arguments

phi vector of AR coefficients

Details

The AR coefficients are transformed to PACF and then the determinant is computed as a product of PACF terms as given in McLeod and Zhang (2006, eqn. 4).

Value

Determinant

Author(s)

A.I. McLeod and Y. zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[FastLoglikelihoodAR](#)

Examples

```
DetAR(c(0.1,0.1,0.1))
```

FastLoglikelihoodAR *Fast Computation of the Loglikelihood Function in AR*

Description

Computation of the loglikelihood is $O(1)$ flops in repeated evaluations of the loglikelihood holding the data fixed and varying the parameters. This is useful in exact MLE estimation.

Usage

```
FastLoglikelihoodAR(phi, n, CD)
```

Arguments

| | |
|-----|---------------------|
| phi | AR coefficients |
| n | length of series |
| CD | Champernowne matrix |

Details

The details of this computation are described in McLeod and Zhang (2006).

Value

Loglikelihood

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[ChampernowneD](#), [LoglikelihoodAR](#)

Examples

```
#Compute the loglikelihood using the direct method as implemented
# in LoglikelihoodAR and using the fast method
phi<-PacfToAR(rep(0.5,10))
p<-length(phi)
z<-SeriesA-mean(SeriesA)
n<-length(z)
L1<-LoglikelihoodAR(phi, z)
cd<-ChampernowneD(z,p,MeanZero=TRUE)
L2<-FastLoglikelihoodAR(phi,n,cd)
out<-c(L1,L2)
names(out)<-c("direct","fast")
out
```

 FitAR

Fit AR, ARp and ARz

Description

Exact MLE for full AR as well as subset AR. Both subset ARp and subset ARz models are implemented. For subset ARp models the R function `arima` is used. For full AR and subset ARz models, algorithm of McLeod & Zhang (2006) is implemented. The LS algorithm for subset ARp is also available as an option.

Usage

```
FitAR(z, p, lag.max = "default", ARModel = "ARz", ...)
```

Arguments

| | |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>z</code> | time series, vector or ts object. |
| <code>p</code> | <code>p</code> specifies the model. If <code>length(p)</code> is 1, an AR(<code>p</code>) is assumed and if <code>p</code> has length greater than 1, a subset ARp or ARz is assumed - the default is ARz. For example, to fit a subset model with lags 1 and 4 present, set <code>p</code> to <code>c(1,4)</code> or equivalently <code>c(1,0,0,4)</code> . To fit a subset model with just lag 4, you must use <code>p=c(0,0,0,4)</code> since <code>p=4</code> will fit a full AR(4). |
| <code>lag.max</code> | the residual autocorrelations are tabulated for lags 1, ..., <code>lag.max</code> . Also <code>lag.max</code> is used for the Ljung-Box portmanteau test. |
| <code>ARModel</code> | which subset model, ARz or ARp |
| <code>...</code> | optional arguments which are passed to <code>FitARz</code> or <code>FitARp</code> |

Details

The exact MLE for AR(p) and subset ARz use methods described in McLeod and Zhang (2006). In addition the exact MLE for the mean can be computed using an iterative backfitting approach described in McLeod and Zhang (2008).

The subset ARp model can be fit by exact MLE using the R function `arima` or by least-squares.

The default for `lag.max` is `min(300, ceiling(length(z)/5))`

Value

A list with class name "FitAR" and components:

| | |
|-----------------------------|----------------------------------------------------------------------------------|
| <code>loglikelihood</code> | value of the loglikelihood |
| <code>phiHat</code> | coefficients in AR(p) – including 0's |
| <code>sigsqHat</code> | innovation variance estimate |
| <code>muHat</code> | estimate of the mean |
| <code>covHat</code> | covariance matrix of the coefficient estimates |
| <code>zetaHat</code> | transformed parameters, <code>length(zetaHat) = \#</code> coefficients estimated |
| <code>RacfMatrix</code> | residual autocorrelations and sd for lags 1, ..., <code>lag.max</code> |
| <code>LjungBox</code> | table of Ljung-Box portmanteau test statistics |
| <code>SubsetQ</code> | parameters in AR(p) – including 0's |
| <code>res</code> | innovation residuals, same length as <code>z</code> |
| <code>fits</code> | fitted values, same length as <code>z</code> |
| <code>pvec</code> | lags used in AR model |
| <code>demean</code> | TRUE if mean estimated otherwise assumed zero |
| <code>FitMethod</code> | "MLE" or "LS" |
| <code>IterationCount</code> | number of iterations in mean mle estimation |
| <code>convergence</code> | value returned by <code>optim</code> – should be 0 |
| <code>MLEMeanQ</code> | TRUE if mle for mean algorithm used |
| <code>ARModel</code> | "ARp" if FitARp used, otherwise "ARz" |
| <code>tsp</code> | <code>tsp(z)</code> |
| <code>call</code> | result from <code>match.call()</code> showing how the function was called |
| <code>ModelTitle</code> | description of model |
| <code>DataTitle</code> | returns <code>attr(z,"title")</code> |
| <code>z</code> | time series data input |

Note

There are generic `print`, `summary`, `coef` and `resid` functions for class "FitAR".

It is somewhat surprising that in the 'ARp' subset autoregression quite different subsets may be chosen depending on the choice of 'lag.max'. For example, with the 'lynx' taking `lag.max = 15, 20` produces subsets 1, 2, 4, 10, 11 and 1, 2, 10, 11 using the BIC. This also occurs even with the AIC. See sixth example below.

Author(s)

A.I. McLeod

References

McLeod, A.I. and Zhang, Y. (2006). Partial Autocorrelation Parameterization for Subset Autoregression. *Journal of Time Series Analysis*, 27, 599-612.

McLeod, A.I. and Zhang, Y. (2008a). Faster ARMA Maximum Likelihood Estimation, *Computational Statistics and Data Analysis*, 52-4, 2166-2176. DOI link: <http://dx.doi.org/10.1016/j.csda.2007.07.020>.

McLeod, A.I. and Zhang, Y. (2008b, Submitted). Improved Subset Autoregression: With R Package. *Journal of Statistical Software*.

See Also

[FitARp](#), [FitARz](#), [GetFitARz](#), [FitARp](#), [GetFitARpMLE](#), [RacfPlot](#)

Examples

```
#First example: fit exact MLE to AR(4)
set.seed(3323)
phi<-c(2.7607,-3.8106,2.6535,-0.9238)
z<-SimulateGaussianAR(phi,1000)
ans<-FitAR(z,4,MeanMLEQ=TRUE)
ans
coef(ans)

## Not run: #save time building package!
#Second example: compare with sample mean result
ans<-FitAR(z,4)
coef(ans)

#Third example: fit subset ARz and ARp models
z<-log(lynx)
FitAR(z, c(1,2,4,7,10,11))
#now obtain exact MLE for Mean as well
FitAR(z, c(1,2,4,7,10,11), MeanMLE=TRUE)
#subset ARp using exact MLE
FitAR(z, c(1,2,4,7,10,11), ARModel="ARp", MLEQ=TRUE)
#subset ARp using LS
FitAR(z, c(1,2,4,7,10,11), ARModel="ARp", MLEQ=FALSE)
#or
FitAR(z, c(1,2,4,7,10,11), ARModel="ARp")

#Fourth example: use UBIC model selection to fit subset models
z<-log(lynx)
#ARz case
p<-SelectModel(z,ARModel="ARz")[[1]]$p
ans1<-FitAR(z, p)
ans1
```

```

ans1$ARModel

#ARp case
p<-SelectModel(z,ARModel="ARp")[[1]]$p
ans2<-FitAR(z, p, ARModel="ARp")
ans2
ans2$ARModel

#Fifth example: fit a full AR(p) using AIC/BIC methods
z<-log(lynx)
#BIC
p<-SelectModel(z,ARModel="AR")[1,1]
ans1<-FitAR(z, p)
ans1
#AIC
p<-SelectModel(z, ARModel="AR", Criterion="AIC")[1,1]
ans2<-FitAR(z, p)
ans2

## End(Not run)

#Sixth Example: Subset autoregression depends on lag.max!
#Because least-squares is used, P=lag.max observations are
# are deleted. This causes different results depending on lag.max.
#This phenomenon does not happen with "ARz" subset models
#ARp models depend on lag.max
SelectModel(z,lag.max=15,ARModel="ARp", Criterion="BIC", Best=1)
SelectModel(z,lag.max=20,ARModel="ARp", Criterion="BIC", Best=1)
#ARz models do NOT depend in this way on lag.max.
#Obviously if some lags beyond the initial value of lag.max are
# found to be important, then there is a dependence but this
# is not a problem!
SelectModel(z,lag.max=15,ARModel="ARz", Criterion="BIC", Best=1)
SelectModel(z,lag.max=20,ARModel="ARz", Criterion="BIC", Best=1)

```

FitARp

Fit subset ARp Models

Description

The subset ARp is defined as an AR(p) in which some of the ar-coefficients are constrained to zero. This is the usual type of subset AR. In contrast the ARz model constrains some of the partial autocorrelation coefficients to zero.

Usage

```
FitARp(z, p, lag.max = "default", MLEQ = FALSE)
```

Arguments

| | |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| z | time series, vector or ts object |
| p | p specifies the model. If length(p) is 1, an AR(p) is assumed and if p has length greater than 1, a subset ARp is assumed. For example, to fit a subset model with lags 1 and 4 present set p to c(1,4) or equivalently c(1,0,0,4). To fit a subset model with just lag 4, you must use p=c(0,0,0,4) since p=4 will fit a full AR(4). |
| lag.max | the residual autocorrelations are tabulated for lags 1, . . . , lag.max. Also lag.max is used for the Ljung-Box portmanteau test. |
| MLEQ | TRUE, use MLE. FALSE, use LS |

Details

Subset ARp model is fit using exact MLE. The built-in arima function is used for MLE. When MLEQ=FALSE, LS is used. LS is has been widely used in past for subset ARp fitting.

Value

A list with class name "FitAR" and components:

| | |
|----------------|---------------------------------------------------------------------|
| loglikelihood | value of the loglikelihood |
| phiHat | coefficients in AR(p) – including 0's |
| sigsqHat | innovation variance estimate |
| muHat | estimate of the mean |
| covHat | covariance matrix of the coefficient estimates |
| zetaHat | transformed parameters, length(zetaHat) = \# coefficients estimated |
| RacfMatrix | residual autocorrelations and sd for lags 1, . . . , lag.max |
| LjungBox | table of Ljung-Box portmanteau test statistics |
| SubsetQ | parameters in AR(p) – including 0's |
| res | innovation residuals, same length as z |
| fits | fitted values, same length as z |
| pvec | lags used in AR model |
| demean | TRUE if mean estimated otherwise assumed zero |
| FitMethod | "MLE" or "LS" |
| IterationCount | number of iterations in mean mle estimation |
| convergence | value returned by optim – should be 0 |
| MLEMeanQ | TRUE if mle for mean algorithm used |
| ARModel | "ARp" if FitARp used, otherwise "ARz" |
| tsp | tsp(z) |
| call | result from match.call() showing how the function was called |
| ModelTitle | description of model |
| DataTitle | returns attr(z,"title") |
| z | time series data input |

Author(s)

A.I. McLeod

References

McLeod, A.I. and Zhang, Y. (2006). Partial Autocorrelation Parameterization for Subset Autoregression. *Journal of Time Series Analysis*, 27, 599-612.

McLeod, A.I. and Zhang, Y. (2008a). Faster ARMA Maximum Likelihood Estimation, *Computational Statistics and Data Analysis* 52-4, 2166-2176. DOI link: <http://dx.doi.org/10.1016/j.csda.2007.07.020>.

McLeod, A.I. and Zhang, Y. (2008b, Submitted). Improved Subset Autoregression: With R Package. *Journal of Statistical Software*.

See Also

[FitAR](#), [FitARz](#), [GetFitARz](#), [FitARp](#), [GetFitARpMLE](#), [RacfPlot](#)

Examples

```
#First Example: Fit to AR(4)
set.seed(3323)
phi<-c(2.7607,-3.8106,2.6535,-0.9238)
z<-SimulateGaussianAR(phi,1000)
#MLE using arima
ans1<-FitARp(z,4,MLEQ=TRUE)
ans1
coef(ans1)
#OLS
ans2<-FitARp(z,4,MLEQ=FALSE)
ans2
coef(ans2)

## Not run: #save time building package
#Second Example: Fit subset ARp model
z<-log(lynx)
#MLE
FitARp(z, c(1,2,4,7,10,11),MLEQ=TRUE)
#LS
FitARp(z, c(1,2,4,7,10,11),MLEQ=FALSE)

#Third Example: Use UBIC model selection to fit subset models
z<-log(lynx)
p<-SelectModel(z,ARModel="ARp")[[1]]$p
#MLE #error returned by arima
#ans1<-FitARp(z, p, MLEQ=TRUE)
#ans1
#LS
ans2<-FitARp(z, p, MLEQ=FALSE)
ans2

## End(Not run)
```

FitARz

*Subset ARz Model Fitting***Description**

The subset ARz model, defined by constraining partial autocorrelations to zero, is fitted using exact MLE. When $\text{length}(p)=1$, an AR(p) is fit by MLE.

Usage

```
FitARz(z, p, demean = TRUE, MeanMLEQ = FALSE, lag.max = "default")
```

Arguments

| | |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| z | time series, vector or ts object |
| p | p specifies the model. If $\text{length}(p)$ is 1, an AR(p) is assumed and if p has length greater than 1, a subset ARz is assumed. For example, to fit a subset model with lags 1 and 4 present set p to c(1,4) or equivalently c(1,0,0,4). To fit a subset model with just lag 4, you must use $p=c(0,0,0,4)$ since $p=4$ will fit a full AR(4). |
| demean | TRUE, mean estimated. FALSE, mean is zero. |
| MeanMLEQ | use exact MLE for mean parameter |
| lag.max | the residual autocorrelations are tabulated for lags 1, ..., lag.max. Also lag.max is used for the Ljung-Box portmanteau test. |

Details

The model and its properties are discussed in McLeod and Zhang (2006) and McLeod and Zhang (2008).

Value

A list with class name "FitAR" and components:

| | |
|---------------|-------------------------------------------------------------------------------------|
| loglikelihood | value of the loglikelihood |
| phiHat | coefficients in AR(p) – including 0's |
| sigsqHat | innovation variance estimate |
| muHat | estimate of the mean |
| covHat | covariance matrix of the coefficient estimates |
| zetaHat | transformed parameters, $\text{length}(\text{zetaHat}) = \#$ coefficients estimated |
| RacfMatrix | residual autocorrelations and sd for lags 1, ..., lag.max |
| LjungBox | table of Ljung-Box portmanteau test statistics |
| SubsetQ | parameters in AR(p) – including 0's |
| res | innovation residuals, same length as z |

| | |
|-----------------------------|---------------------------------------------------------------------------|
| <code>fits</code> | fitted values, same length as <code>z</code> |
| <code>pvec</code> | lags used in AR model |
| <code>demean</code> | TRUE if mean estimated otherwise assumed zero |
| <code>FitMethod</code> | "MLE" or "LS" |
| <code>IterationCount</code> | number of iterations in mean mle estimation |
| <code>convergence</code> | value returned by optim – should be 0 |
| <code>MLEMeanQ</code> | TRUE if mle for mean algorithm used |
| <code>ARModel</code> | "ARp" if FitARp used, otherwise "ARz" |
| <code>tsp</code> | <code>tsp(z)</code> |
| <code>call</code> | result from <code>match.call()</code> showing how the function was called |
| <code>ModelTitle</code> | description of model |
| <code>DataTitle</code> | returns <code>attr(z,"title")</code> |
| <code>z</code> | time series data input) |

Note

Normally one would use the `FitAR` function which then calls this function for the ARz case.

Author(s)

A.I. McLeod

References

McLeod, A.I. and Zhang, Y. (2006). Partial Autocorrelation Parameterization for Subset Autoregression. *Journal of Time Series Analysis*, 27, 599-612.

McLeod, A.I. and Zhang, Y. (2008a). Faster ARMA Maximum Likelihood Estimation, *Computational Statistics and Data Analysis*, 52-4, 2166-2176. DOI link: <http://dx.doi.org/10.1016/j.csda.2007.07.020>.

McLeod, A.I. and Zhang, Y. (2008b, Submitted). Improved Subset Autoregression: With R Package. *Journal of Statistical Software*.

See Also

[FitAR](#), [FitARp](#), [GetFitARz](#), [GetFitARpMLE](#), [RacfPlot](#)

Examples

```
#First Example: Fit exact MLE to AR(4)
set.seed(3323)
phi<-c(2.7607,-3.8106,2.6535,-0.9238)
z<-SimulateGaussianAR(phi,1000)
ans<-FitARz(z,4,MeanMLEQ=TRUE)
ans
coef(ans)
```

```

## Not run: #save time building package
#Second Example: compare with sample mean result
ans<-FitARz(z,4)
coef(ans)

#Third Example: Fit subset ARz
z<-log(lynx)
FitARz(z, c(1,2,4,7,10,11))
#now obtain exact MLE for Mean as well
FitARz(z, c(1,2,4,7,10,11), MeanMLE=TRUE)

#Fourth Example: Fit subset ARz
somePACF<-c(0.5,0,0,0,-0.9)
someAR<-PacfToAR(somePACF)
z<-SimulateGaussianAR(someAR,1000)
ans=FitARz(z, c(1,5),MeanMLEQ=TRUE)
coef(ans)
GetFitARz(z,c(1,5))#assuming a known zero mean

## End(Not run)

```

fitted.FitAR

Fitted Values from "FitAR" Object

Description

Method function, extracts fitted values from FitAR object.

Usage

```

## S3 method for class 'FitAR'
fitted(object, ...)

```

Arguments

| | |
|--------|-------------------------|
| object | object of class "FitAR" |
| ... | optional arguments |

Value

Vector of fitted values

Author(s)

A.I. McLeod and Y. Zhang

See Also

[FitAR](#)

Examples

```
out<-FitAR(SeriesA, c(1,2,6,7))
fitted(out)
```

FromSymmetricStorageUpper

Converts a Matrix from Symmetric Storage Mode to Regular Format

Description

Utility function.

Usage

```
FromSymmetricStorageUpper(x)
```

Arguments

x a vector which represents a matrix in upper triangular form

Value

symmetric matrix

Author(s)

A.I. McLeod

Examples

```
FromSymmetricStorageUpper(1:5)
```

FXRates

Foreign exchange rates

Description

Daily foreign exchange rates were obtained for: YenUS, DmUS, USGB, CanUS. From 1983-12-13 to 2008-11-12, 6330 values in each series.

Usage

```
data(FXRates)
```

Format

A data frame with 6330 observations on the following 5 variables.

Date a character factor, dates

YenUS Yen/US exchange rate

DmUS Deutsche Mark/US Dollar, exchange rate

USGB US/Great Britain exchange rate

CanUS Canada/US exchange rate

Details

The dates run from "1983-12-13" to "2008-11-24" and were included in the downloaded file. There were 48 missing values out of a total of $4 \times 6330 = 25320$ values. Missing values were replaced with the previous value.

Source

http://www.econstats.com/fx/fx__d1.htm

Examples

```
head(FXRates)
```

Get1G

Internal Utility Function: BLUE Mean

Description

This function is not normally used directly by the user. It is used in the exact mle for mean.

Usage

```
Get1G(phi, n)
```

Arguments

phi a vector of AR coefficients

n length of series

Value

A vector used in the mle computation of the mean.

Author(s)

A.I. McLeod

See Also[GetARMeanMLE](#)**Examples**

```
#Simulate an AR(2) and compute the exact mle for mean
set.seed(7771111)
n<-50
phi<-c(1.8,-0.9)
z<-SimulateGaussianAR(phi, n)
g1<-Get1G(phi, length(z))
sum(g1*z)/sum(g1)
#sample mean
mean(z)
#more directly with getArMu
GetARMeanMLE(z,phi)
```

GetARMeanMLE

Exact MLE for Mean in AR(p)

Description

Details of this algorithm are given in McLeod and Zhang (2007).

Usage

```
GetARMeanMLE(z, phi)
```

Arguments

`z` vector of length `n` containing the time series
`phi` vector of AR coefficients

Value

Estimate of mean

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[mean](#)

Examples

```

#Simulate a time series with mean zero and compute the exact
#mle for mean and compare with sample average.
## Not run: #save time building package!
set.seed(3323)
phi<-c(2.7607,-3.8106,2.6535,-0.9238)
z<-SimulateGaussianAR(phi,1000)
ans1<-mean(z)
ans2<-GetARMeanMLE(z,phi)
# define a direct MLE function
"DirectGetMeanMLE" <-
function(z, phi){
  GInv<-solve(toeplitz(TacvAR(phi, length(z)-1)))
  g1<-colSums(GInv)
  sum(g1*z)/sum(g1)
}
ans3<-DirectGetMeanMLE(z,phi)
ans<-c(ans1,ans2,ans3)
names(ans)<-c("mean", "GetARMeanMLE", "DirectGetMeanMLE")
ans

## End(Not run)

```

GetB*Internal Utility Function*

Description

The user would not normally use this function. The function is needed for exact mle for mean. Used in Get1G which is called from GetARMeanMLE.

Usage

```
GetB(phi)
```

Arguments

phi vector of AR coefficients

GetFitAR*MLE for AR, ARp and ARz*

Description

Obtains the exact MLE for AR(p) or subset AR models ARp or ARz. This function is used by FitAR. One might prefer to use GetFitAR for applications such as bootstrapping since it is faster than FitAR.

Usage

```
GetFitAR(z, p, ARModel = "ARz", ...)
```

Arguments

| | |
|---------|---------------------------------------------------------------|
| z | time series |
| p | model order or subset lags |
| ARModel | either "ARp" or "ARz" corresponding to GetFitARp or GetFitARz |
| ... | optional arguments which are passed to GetFitARp or GetFitARz |

Details

This is just a shell which simply invokes either GetFitARp or GetFitARz

Value

| | |
|---------------|----------------------------------|
| loglikelihood | value of maximized loglikelihood |
| zetaHat | estimated zeta parameters |
| phiHat | estimated phi parameters |
| convergence | result from optim |

Author(s)

A.I. McLeod

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

McLeod, A.I. and Zhang, Y. (2008a). Faster ARMA Maximum Likelihood Estimation, *Computational Statistics and Data Analysis* 52-4, 2166-2176. DOI link: <http://dx.doi.org/10.1016/j.csda.2007.07.020>.

McLeod, A.I. and Zhang, Y. (2008b, Submitted). Improved Subset Autoregression: With R Package. *Journal of Statistical Software*.

See Also

[FitAR](#)

Examples

```
#compare results from GetFitAR and FitAR
z<-log(lynx)
z<-z - mean(z)
GetFitAR(z, c(1,2,8))
out<-FitAR(log(lynx), c(1,2,8))
out
coef(out)
```

Description

For ARp subset models, the least squares estimates are computed. The exact loglikelihood is then determined. The estimated parameters are checked to see if they are in the AR admissible region.

Usage

```
GetFitARpLS(z, pvec)
```

Arguments

| | |
|------|---------------------------------------------------------------|
| z | vector or ts object, the time series |
| pvec | lags included in subset AR. If pvec = 0, white noise assumed. |

Details

The R function `lsfit` is used.

Value

a list with components:

| | |
|---------------|--------------------------------------------------------------------|
| loglikelihood | the exact loglikelihood |
| phiHat | estimated AR parameters |
| constantTerm | constant term in the linear regression |
| pvec | lags of estimated AR coefficient |
| res | the least squares regression residuals |
| InvertibleQ | True, if the estimated parameters are in the AR admissible region. |
| yX | the y vector and X matrix used for the regression fitting |

Note

This is a helper function for `FitARp` which is invoked by the main package function `FitAR`. Normally the user would `FitAR` since this function provides generic print, summary, resid and plot methods but `GetFitARpLS` is sometimes useful in iterative computations like bootstrapping since it is faster.

Author(s)

A.I. McLeod

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

McLeod, A.I. and Zhang, Y. (2008a). Faster ARMA Maximum Likelihood Estimation, *Computational Statistics and Data Analysis* 52-4, 2166-2176. DOI link: <http://dx.doi.org/10.1016/j.csda.2007.07.020>.

McLeod, A.I. and Zhang, Y. (2008b, Submitted). Improved Subset Autoregression: With R Package. *Journal of Statistical Software*.

See Also

[FitAR](#), [FitARz](#), [GetFitARz](#), [FitARp](#), [GetFitARpMLE](#), [RacfPlot](#)

Examples

```
#Fit subset AR using LS
#normally use FitAR
ans<-FitAR(SeriesA, c(1,2,7), ARModel="ARp", MLEQ=FALSE)
#could also use FitARp
ans<-FitARp(SeriesA, c(1,2,7))
#for some applications GetFitARpLS is simpler and faster
ansLS<-GetFitARpLS(SeriesA, c(1,2,7))
ansLS
```

GetFitARpMLE

Exact MLE for subset ARp Models

Description

Uses built-in function `arima` to fit subset ARp model, that is, the subset model is formed by constraining some coefficients to zero.

Usage

```
GetFitARpMLE(z, pvec)
```

Arguments

`z` time series
`pvec` lags included in AR model. If `pvec = 0`, white noise model assumed.

Details

Due to the optimization algorithms used by `arima`, this method is not very reliable. The optimization may simply fail. Example 1 shows it working but in Example 2 below it fails.

Value

a list with components:

| | |
|---------------|--------------------------------------------------------------------|
| loglikelihood | the exact loglikelihood |
| phiHat | estimated AR parameters |
| constantTerm | constant term in the linear regression |
| pvec | lags of estimated AR coefficient |
| res | the least squares regression residuals |
| InvertibleQ | True, if the estimated parameters are in the AR admissible region. |

Author(s)

A.I. McLeod

References

McLeod, A.I. and Zhang, Y. (2006). Partial Autocorrelation Parameterization for Subset Autoregression. *Journal of Time Series Analysis*, 27, 599-612.

McLeod, A.I. and Zhang, Y. (2008a). Faster ARMA Maximum Likelihood Estimation, *Computational Statistics and Data Analysis* 52-4, 2166-2176. DOI link: <http://dx.doi.org/10.1016/j.csda.2007.07.020>.

McLeod, A.I. and Zhang, Y. (2008b, Submitted). Improved Subset Autoregression: With R Package. *Journal of Statistical Software*.

See Also

[FitAR](#), [FitARz](#), [GetFitARz](#), [FitARp](#), [RacfPlot](#)

Examples

```
#Example 1. MLE works
z<-log(lynx)
p<-c(1,2,4,7,10,11)
GetFitARpMLE(z, p)
#
#Example 2. MLE fails with error.
p<-c(1,2,9,12)
## Not run: GetFitARpMLE(z, p)
```

GetFitARz

*Exact MLE for AR(p) and Subset ARz – Short Version***Description**

Obtain the exact MLE for AR(p) or subset ARz model. This function is used by `FitAR` and `FitARz`. One might prefer to use `GetFitARz` for applications such as bootstrapping since it is faster.

Usage

```
GetFitARz(z, pvec, MeanValue=0, ...)
```

Arguments

| | |
|------------------------|----------------------------------------------------------------------------------|
| <code>z</code> | time series |
| <code>pvec</code> | lags included in AR model. If <code>pvec = 0</code> , white noise model assumed. |
| <code>MeanValue</code> | by default it is assumed the mean of <code>z</code> is 0 |
| <code>...</code> | optional arguments passed through to <code>optim</code> |

Details

The built-in function `optim` is used to obtain the MLE estimates for an AR or subset AR. First "BFGS" is tried. This usually works fine. In the rare cases where convergence is not obtained, "Nelder-Mead" is used. A warning message is given if this happens.

Value

| | |
|----------------------------|----------------------------------|
| <code>loglikelihood</code> | value of maximized loglikelihood |
| <code>zetaHat</code> | estimated zeta parameters |
| <code>phiHat</code> | estimated phi parameters |
| <code>convergence</code> | result from <code>optim</code> |
| <code>pvec</code> | lags of estimated AR coefficient |
| <code>algorithm</code> | "BFGS" or "Nelder-Mead" |

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

McLeod, A.I. and Zhang, Y. (2008a). Faster ARMA Maximum Likelihood Estimation, *Computational Statistics and Data Analysis* 52-4, 2166-2176. DOI link: <http://dx.doi.org/10.1016/j.csda.2007.07.020>.

McLeod, A.I. and Zhang, Y. (2008b, Submitted). Improved Subset Autoregression: With R Package. *Journal of Statistical Software*.

See Also

[FitAR](#), [FitARz](#), [FitARp](#), [GetFitARpMLE](#), [RacfPlot](#)

Examples

```
#compare results from GetFitARz and FitAR
z<-log(lynx)
z<-z - mean(z)
GetFitARz(z, c(1,2,8))
out<-FitAR(log(lynx), c(1,2,8), ARModel="ARz")
out
coef(out)
```

 GetKappa

Internal Utility Function

Description

Used by Get1G.

Usage

```
GetKappa(phi)
```

Arguments

phi ARCoefficients

 GetLeapsAR

Select lags for Best Subset ARp Model

Description

The subset ARp model is the usual subset model, for example see Tong (1977). This function is used by SelectModel for model identification for ARp models.

Usage

```
GetLeapsAR(z, lag.max = 15, Criterion = "UBIC", Best = 3, Candidates=5, t="default", ExactQ=FALSE)
```

Arguments

| | |
|-------------------------|---------------------------------------------------------------------------------------------------------------|
| <code>z</code> | ts object or vector containing time series |
| <code>lag.max</code> | maximum order of the AR |
| <code>Criterion</code> | default UBIC, other choices are "AIC", "BIC", "EBIC", "BICq", "GIC" |
| <code>Best</code> | the number of based selected. Ignore with "GIC". |
| <code>Candidates</code> | number of models initially selected using the approximate criterion |
| <code>t</code> | tuning parameter, EBIC, BICq, GIC |
| <code>ExactQ</code> | exhaustive numeration using exact likelihood. Still under under development. NOT AVAILABLE IN THIS VERSION |

Details

The R function `leaps` in the R package `leaps` is used to compute the subset regression model with the smallest residual sum of squares containing 1, ..., `lag.max` parameters. The mean is always included, so the only parameters considered are the phi coefficients. After the best models containing 1, ..., `lag.max` parameters are selected the models are individually refit to determine the exact likelihood function for each selected model. Based on this likelihood the UBIC/BIC/AIC is computed and then the best models are selected. The UBIC criterion was developed by Chen and Chen (2007). The EBIC using a tuning parameter, G , where $0 \leq G \leq 1$. The BICq takes a tuning parameter, Q , where $0 < Q < 1$. The GIC takes a tuning parameter, p , where $0 < p < 0.25$.

Value

When 'Criterion' is one of UBIC, AIC, BIC, EBIC, BICq, a list with components:

| | |
|-------------------|----------------------------------------------------------------------------------------|
| <code>p</code> | lags present in model |
| <code>UBIC</code> | approximate UBIC (Chen & Chen, 2007), if <code>Criterion=="UBIC"</code> |
| <code>AIC</code> | approximate AIC (McLeod and Zhang, 2006a, eqn. 15), if <code>Criterion=="AIC"</code> |
| <code>BIC</code> | approximate BIC (McLeod and Zhang, 2006a, eqn. 15), if <code>Criterion=="BIC"</code> |
| <code>EBIC</code> | approximate EBIC (McLeod and Zhang, 2006a, eqn. 15), if <code>Criterion=="EBIC"</code> |
| <code>BICq</code> | approximate BICq, if <code>Criterion=="BICq"</code> |
| <code>GIC</code> | approximate GIC, if <code>Criterion=="GIC"</code> |

Warning

AIC and BIC values produced are not comparable to AIC and BIC produced by `SelectModel` for ARz models. However comparable AIC/BIC values are produced when the selected models are fit by `FitAR`.

Note

Requires `leaps` package. Since the least-squares is used, the number of observations depends on 'lag.max'. Hence different subsets may be chosen depending on the 'lag.max'. See example below.

Author(s)

A.I. McLeod

References

Tong, H. (1977) Some comments on the Canadian lynx data. *Journal of the Royal Statistical Society A* 140, 432-436.

Chen, J. and Chen, Z. (2008). Extended Bayesian Information Criteria for Model Selection with Large Model Space. *Biometrika*.

Changjiang Xu and A. I. McLeod (2010). Bayesian information criterion with Bernoulli prior. Submitted for publication.

Changjiang Xu and A. I. McLeod (2010). Model selection using generalized information criterion. Submitted for publication.

See Also

[SelectModel](#), [GetFitARpLS](#), [leaps](#)

Examples

```
#Example 1: Simple Example
#for the log(lynx) Tong (1977) selected an ARp(1,2,4,10,11)
#using the AIC and a subset selection algorithm. Our more exact
#approach shows that the ARp(1,2,3,4,10,11) has slightly lower
#AIC (using exact likelihood evaluation).
z<-log(lynx)
GetLeapsAR(z, lag.max=11)
GetLeapsAR(z, lag.max=11, Criterion="BIC")

#Example 2: Subset autoregression depends on lag.max!
#Because least-squares is used, P=lag.max observations are
# are deleted. This causes different results depending on lag.max.
#This phenomenon does not happen with "ARz" subset models
#ARp models depend on lag.max
GetLeapsAR(z, lag.max=15, Criterion="BIC")
GetLeapsAR(z, lag.max=20, Criterion="BIC")

#Example 3: Comparing GIC with BIC, AIC, UBIC and BICq
z <- log(lynx)
GetLeapsAR(z, lag.max=15, Criterion="BIC", Best=1)
GetLeapsAR(z, lag.max=15, Criterion="AIC", Best=1)
GetLeapsAR(z, lag.max=15, Criterion="UBIC", Best=1)
GetLeapsAR(z, lag.max=15, Criterion="BICq", Best=1, t=0.25)
GetLeapsAR(z, lag.max=15, Best=1, Criterion="GIC", t=0.01)
ans<-GetLeapsAR(z, lag.max=15, Best=3, Criterion="GIC", t=0.001)
plot(ans)
```

| | |
|--------|------------------------------------------------|
| getRho | <i>Normalized rho unit root test statistic</i> |
|--------|------------------------------------------------|

Description

Utility function used by UnitRootTest

Usage

```
getRho(ans)
```

Arguments

ans output from FitAR

Value

Value of the test statistic

Author(s)

A.I. McLeod

See Also

[getT UnitRootTest](#)

Examples

```
z <- cumsum(rnorm(100))
ans <- FitAR(z, p=1)
getRho(ans)
```

| | |
|------|---------------------------------------|
| getT | <i>t-statistic for unit root test</i> |
|------|---------------------------------------|

Description

Utility function used by UnitRootTest

Usage

```
getT(ans)
```

Arguments

ans output from FitAR

Value

Value of the test statistic

Author(s)

A.I. McLeod

See Also

[getRho UnitRootTest](#)

Examples

```
z <- cumsum(rnorm(100))
ans <- FitAR(z, p=1)
getT(ans)
```

glog

glog transformation

Description

The glog is a better behaved log transformation when some data values are zero or just near zero.

Usage

```
glog(x, a = 1, InverseQ = FALSE)
```

Arguments

| | |
|----------|----------------------------|
| x | numeric vector of data |
| a | additive constant, often 1 |
| InverseQ | inverse glog |

Details

Basic properties of the glog transformation are illustrated in the Mathematica notebook `glog.nb` and its pdf version `glog.pdf` which are available in the package directory `doc`.

Value

transformed data

Author(s)

A.I. McLeod

References

W. Huber, A. von Heydebreck, H. Sultmann, A. Poustka, and M. Vingron. Variance stabilization applied to microarray data calibration and to quantification of differential expression. *Bioinformatics*, 18: S96-S10 2002.

See Also

[bxcx](#)

Examples

```
#usual log transformation doesn't work
all(is.finite(log(sunspot.month)))
#either shifted log
all(is.finite(log(sunspot.month+1)))
#or glog works
all(is.finite(glog(sunspot.month)))
#but glog may be better, especially for values <1 but >=0
```

InformationMatrixAR *Information Matrix for AR(p)*

Description

The Fisher large-sample information matrix per observation for the p coefficients in an AR(p) is computed.

Usage

```
InformationMatrixAR(phi)
```

Arguments

`phi` vector of length p corresponding to the AR(p) coefficients

Details

The Fisher information matrix is computed as the covariance matrix of an AR(p) process with coefficients given in the argument `phi` and with unit innovation variance. The `TacvfAR` function is used to compute the necessary autocovariances. `FitAR` uses `InformationMatrixAR` to obtain estimates of the standard errors for the estimated parameters in the case of the full AR(p) model.

Value

a p -by- p Toeplitz matrix, $p = \text{length}(\text{phi})$

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[FitAR](#), [InformationMatrixARp](#), [TacfAR](#), [InformationMatrixARz](#)

Examples

```
InformationMatrixAR(c(1.8,-0.6))
```

InformationMatrixARp *Fisher Information Matrix Subset Case, ARp*

Description

The large-sample information matrix per observation is computed in a subset AR with the usual parameterization, that is, a subset of the AR coefficients.

Usage

```
InformationMatrixARp(phi, lags)
```

Arguments

| | |
|------|-----------------------------------------|
| phi | vector of coefficients in the subset AR |
| lags | vector indicating lags present in phi |

Details

The subset information matrix is obtained simply by selecting the appropriate rows and columns from the full information matrix. This function is used by `FitARp` to obtain the estimated standard errors of the parameter estimates.

Value

a p-by-p Toeplitz matrix, $p = \text{length}(\text{phi})$

Author(s)

A.I. McLeod & Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[InformationMatrixAR](#), [FitARp](#), [InformationMatrixARz](#)

Examples

```
#variances of parameters in a subset ARp(1,2,6)
fi<-InformationMatrixARp(c(0.36,0.23,0.23),c(1,2,6))
sqrt(diag(solve(fi*197)))
```

InformationMatrixARz *Fisher Information Matrix Subset Case, ARz*

Description

Computes the large-sample Fisher information matrix per observation for the AR coefficients in a subset AR when parameterized by the partial autocorrelations.

Usage

```
InformationMatrixARz(zeta, lags)
```

Arguments

| | |
|------|---------------------------------------------------------------------------------------------|
| zeta | vector of coefficients, ie. partial autocorrelations at lags specified in the argument lags |
| lags | lags in subset model, same length as zeta argument |

Details

The details of the computation are given in McLeod and Zhang (2006, eqn 13). FitAR uses InformationMatrixARz to obtain estimates of the standard errors of the estimated parameters in the subset AR model when partial autocorrelation parameterization is used.

Value

a p-by-p Toeplitz matrix, p=length(zeta)

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[FitAR](#), [InformationMatrixAR](#), [InformationMatrixARp](#)

Examples

```
#Information matrix for ARz(1,4) with parameters 0.9 and 0.9.
InformationMatrixARz(c(0.9, 0.9), lags=c(1,4))
```

 InvertibleQ

Test if Invertible or Stationary-casual

Description

Tests if the polynomial

$$1 - \phi(1)B \dots - \phi(p)B^p,$$

where $p=\text{length}[\text{phi}]$ has all roots outside the unit circle. This is the invertibility condition for the polynomial.

Usage

```
InvertibleQ(phi)
```

Arguments

phi a vector of AR coefficients

Details

The PACF is computed for lags 1, ..., p using eqn. (1) in McLeod and Zhang (2006). The invertibility condition is satisfied if and only if all PACF values are less than 1 in absolute value.

Value

TRUE, if invertibility condition is satisfied. FALSE, if not invertible.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[ARToPacf](#)

Examples

```
#simple examples
InvertibleQ(0.5)
#find the area of the invertible region for AR(2).
#We assume that the parameters must be less than 2 in absolute value.
#From the well-known diagram in the book of Box and Jenkins (1970),
#this area is exactly 4.
NSIM<-10^4
phi1<-runif(NSIM, min=-2, max=2)
phi2<-runif(NSIM, min=-2, max=2)
k<-sum(apply(matrix(c(phi1,phi2),ncol=2), MARGIN=1, FUN=InvertibleQ))
area<-16*k/NSIM
area
```

Jacobian

Jacobian AR-coefficients to Partial Autocorrelations

Description

This is more or less an internal routine used by `InformationMatrixZeta` but it is described in more details since it may be useful in other computations.

Usage

```
Jacobian(zeta)
```

Arguments

`zeta` partial autocorrelation parameters

Details

The computation is described in detail in McLeod and Zhang (2006, Section 2.2)

Value

square matrix of order `length(zeta)`

Author(s)

A.I. McLeod

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also[InformationMatrixARz](#)**Examples**

```
#In McLeod and Zhang (2006, p.603) a symbolic example is given
# for the AR(4).
#
Jacobian(rep(0.8,4))
```

JacobianK

Internal Utility Function

Description

The matrix defined in eqn. (10) of McLeod and Zhang (2006). Used by the function [Jacobian](#).

Usage

```
JacobianK(zeta, k)
```

Arguments

| | |
|------|--------------------------|
| zeta | partial autocorrelations |
| k | k-th Jacobian |

Value

Matrix

Author(s)

A.I. McLeod

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also[Jacobian](#)**Examples**

```
JacobianK(rep(0.8,4),3)
```

| | |
|----------------|-----------------------------------|
| JarqueBeraTest | <i>Jarque-Bera Normality Test</i> |
|----------------|-----------------------------------|

Description

A powerful omnibus test for normality.

Usage

```
JarqueBeraTest(z)
```

Arguments

z vector of data

Details

This test is derived as a Lagrange multiplier test for normal distribution in the family of Pearson distributions (Jarque and Bera, 1987).

Value

LM value of the LM statistic
pvalue p-value

Author(s)

A.I. McLeod

References

Jarque, C.M. and Bera, A.K. (1987). A Test for Normality of Observations and Regression Residuals. *International Statistical Review* 55, 163-172

Examples

```
#some normal data  
z<-rnorm(100)  
JarqueBeraTest(z)  
#some skewed data  
z<-rexp(100)  
JarqueBeraTest(z)  
#some thick tailed data  
z<-rt(100,5)  
JarqueBeraTest(z)
```

LBQPlot

Plot Ljung-Box Test P-value vs Lag

Description

The Ljung-Box portmanteau p-value is plotted vs lag.

Usage

```
LBQPlot(res, lag.max = 30, StartLag = k + 1, k = 0, SquaredQ = FALSE)
```

Arguments

| | |
|----------|--------------------------------------------------------------------------------------------------------------------|
| res | residuals |
| lag.max | maximum lag |
| StartLag | starting lag |
| k | number of AR parameters fit |
| SquaredQ | default, SquaredQ = FALSE, regular autocorrelations. If SquaredQ = TRUE use autocorrelations of squared residuals. |

Value

Plot is produced as a side-effect. No output

Note

This function is normally invoked when plot.FitAR is used.

Author(s)

A.I. McLeod and Y. Zhang

References

Ljung, G.M. and Box, G.E.P. (1978) On a measure of lack of fit in time series models. *Biometrika* 65, 297-303.

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[plot.FitAR](#), [FitAR](#)

Examples

```
#fit subset AR and plot diagnostic check
out<-FitAR(SeriesA, c(1,2,7), ARModel="ARp")
res<-resid(out)
LBQPlot(res)
#note that plot produces LBQPlot and RacfPlot
plot(out)
```

LjungBoxTest

Ljung-Box Test for Randomness

Description

The Ljung-Box Portmanteau test for the goodness of fit of ARIMA models is implemented.

Usage

```
LjungBoxTest(res, k=0, lag.max=30, StartLag=1, SquaredQ=FALSE)
```

Arguments

| | |
|----------|-----------------------------------------------------------------------|
| res | residuals |
| k | number of ARMA parameters, default k = 0 |
| lag.max | maximum lag, default MaxLag = 30 |
| StartLag | test is done for lags m=StartLag:MaxLag, default StartLag = 1 |
| SquaredQ | if TRUE, use squared residuals for ARCH test, default Squared = FALSE |

Details

This test is described in detail in Wei (2006, p.153, eqn. 7.5.1). The df are given by h-k, where h is the lag, running from StartLag to lag.max, when h-k < 1, it is reset to 1. This is ok, since the test is conservative in this case.

A powerful test for ARCH and other nonlinearities is obtained by using squared values of the series to be tested (McLeod & Li, 1983). Note that if Squared=TRUE is used the data "res" is centered by sample mean correction before squaring.

Value

A matrix with columns labelled m, Qm, pvalue, where m is the lag and Qm is the Ljung-Box Portmanteau statistic and pvalue its p-value.

Note

This test may also be used to test a time series for randomness taking k = 0.

Author(s)

A.I. McLeod

References

W.W.S. Wei (2006, 2nd Ed.), *Time Series Analysis: Univariate and Multivariate Methods*.

A.I. McLeod. & W.K. Li (1983), Diagnostic checking ARMA time series models using squared-residual autocorrelations, *Journal of Time Series Analysis* **4**, 269–273.

See Also

[Box.test](#)

Examples

```
#test goodness-of-fit of AR(2) model fit to log(lynx)
data(lynx)
z<-log(lynx)
ans<-FitAR(z, 1:2)
#notice that the test is also available as a component of the output of FitAR
ans$LjungBox
#a plot of the test is produced
plot(ans)
#doing the test manually
res<-resid(ans)
LjungBoxTest(res, k=2, lag.max=20, StartLag=5)

#test for subset case
z<-log(lynx)
pvec<-SelectModel(z, ARModel="ARz", Criterion="BIC", lag.max=10, Best=1)
ans<-FitAR(z, pvec)
plot(ans)
res<-resid(ans)
LjungBoxTest(res, k=length(pvec), lag.max=20, StartLag=11)
#test for ARCH effect,
LjungBoxTest(res,SquaredQ=TRUE)
```

 LoglikelihoodAR

Exact Loglikelihood for AR

Description

The exact loglikelihood function, defined in eqn. (6) of McLeod & Zhang (2006) is computed. Requires $O(n)$ flops, $n = \text{length}(z)$.

Usage

```
LoglikelihoodAR(phi, z, MeanValue = 0)
```

Arguments

| | |
|-----------|---------------------------------------------------------------------------------------|
| phi | AR parameters |
| z | time series data, not assumed mean corrected |
| MeanValue | usually this is mean(z) but it could be another value for example the MLE of the mean |

Details

Eqn (6) of McLeod and Zhang (2006) may be written

$$-(n/2) \log(\hat{\sigma}_a^2) - (1/2) \log(g_p),$$

where $\hat{\sigma}_a^2$ is the residual variance and g_p is the covariance determinant.

Value

The value of the loglikelihood is returned

Warning

No check is done for stationary-causal process

Note

For MLE computation it is better to use [FastLoglikelihoodAR](#) since for repeated likelihood evaluations this requires only $O(1)$ flops vs $O(n)$ flops, where $n = \text{length}(z)$.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[FastLoglikelihoodAR](#)

Examples

```
#Fit a subset model to Series A and verify the loglikelihood
out<-FitAR(SeriesA, c(1,2,7))
out
#either using print.default(out) to see the components in out
#or applying LoglikelihoodAR () by first obtaining the phi parameters as out$phiHat.

#
LoglikelihoodAR(out$phiHat, SeriesA, MeanValue=mean(SeriesA))
```

Ninemile

Douglas Fir Treerings, Nine Mile Canyon, Utah, 1194-1964

Description

A treering time series comprises of 771 values showing a periodicity of around 10 years.

Usage

```
data(Ninemile)
```

Format

ts object with title attribute

Source

Hipel, K.W. and McLeod, A.I. (2006). Time Series Modelling of Water Resources and Environmental Systems.

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

Examples

```
ans<-FitAR(Ninemile, c(1,2,9))
summary(ans)
```

PacfDL

Partial Autocorrelations via Durbin-Levinson

Description

Given autocovariances, the partial autocorrelations and/or autoregressive coefficients in an AR may be determined using the Durbin-Levinson algorithm. If the autocovariances are sample autocovariances, this is equivalent to using the Yule-Walker equations. But as noted below our function is more general than the built-in R functions.

Usage

```
PacfDL(c, LinearPredictor = FALSE)
```

Arguments

`c` autocovariances at lags $0, 1, \dots, p = \text{length}(c) - 1$
`LinearPredictor` if TRUE, AR coefficients are also determined using the Yule-Walker method

Details

The Durbin-Levinson algorithm is described in many books on time series and numerical methods, for example Percival and Walden (1993, eqn 403).

Value

If `LinearPredictor = FALSE`, vector of length $p = \text{length}(c) - 1$ containing the partial autocorrelations at lags $1, \dots, p$. Otherwise a list with components:

`Pacf` vector of partial autocorrelations
`ARCoefficients` vector of AR coefficients
`ResidualVariance` residual variance for AR(p)

Warning

Stationarity is not tested.

Note

Sample partial autocorrelations can also be computed with the `acf` function and Yule-Walker estimates can be computed with the `ar` function. Our function `PacfDL` provides more flexibility since then input `c` may be any valid autocovariances not just the usual sample autocovariances. For example, we can determine the minimum mean square error one-step ahead linear predictor of order p for theoretical autocovariances from a fractional arma or other linear process.

Author(s)

A.I. McLeod and Y. Zhang

References

Percival, D.B. and Walden, A.T. (1993). Spectral Analysis For Physical Applications, Cambridge University Press.

See Also

[acf](#), [ar](#)

Examples

```

#first define a function to compute the Sample Autocovariances
sacvf<-function(z, lag.max){
  c(acf(z, plot=FALSE, lag.max=lag.max)$acf)*(length(z)-1)/length(z)
}
#now compute PACF and also fit AR(7) to SeriesA
ck<-sacvf(SeriesA, 7)
PacfDL(ck)
PacfDL(ck, LinearPredictor = TRUE)
#compare with built-in functions
pacf(SeriesA, lag.max=7, plot=FALSE)
ar(SeriesA, lag.max=7, method="yw")
#fit an optimal linear predictor of order 10 to MA(1)
g<-TacfMA(0.8,5)
PacfDL(g, LinearPredictor=TRUE)
#
#Compute the theoretical pacf for MA(1) and plot it
ck<-c(1,-0.4,rep(0,18))
AcfPlot(PacfDL(ck)$Pacf)
title(main="Pacf of MA(1), r(1)=-0.4")

```

PacfPlot

*Plot Partial Autocorrelations and Limits***Description**

The sample partial autocorrelations and their individual 95 percent confidence intervals are plotted under the assumption the model is contained in an AR(P), where P is a specified maximum order.

Usage

```
PacfPlot(z, lag.max = 15, ...)
```

Arguments

| | |
|---------|---------------------------------------------|
| z | time series |
| lag.max | maximum lag, P |
| ... | optional parameters passed through to plot. |

Details

The Burg algorithm is used to estimate the PACF.

Value

No value is returned. Graphical output is produced as side-effect.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[ar.burg pacf](#)

Examples

```
#For the log(lynx) series and taking lag.max=15, the PacfPlot and
# the minimum BIC subset selection produce the same result.
z<-log(lynx)
PacfPlot(z)
SelectModel(z,lag.max=15,ARModel="ARz",Best=1,Criterion="BIC")
```

PacfToAR

Transform from PACF Parameters to AR Coefficients

Description

Transforms AR partial autocorrelation function (PACF) parameters to AR coefficients based on the Durbin-Levinson recursion.

Usage

```
PacfToAR(zeta)
```

Arguments

zeta vector of AR PACF parameters

Details

See Mcleod and Zhang (2006)

Value

Vector of AR coefficients

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[InvertibleQ](#), [PacfToAR](#)

Examples

```
somePACF<-c(0.5,0.6,0.7,0.8,-0.9,-0.8)
someAR<-PacfToAR(somePACF)
test<-ARToPacf(someAR)
#this should be very small
sum(abs(test-somePACF))
```

plot.FitAR

Plot Method for "FitAR" Object

Description

Diagnostic plots: portmanteau p-values; residual autocorrelation plot; normal probability plot and Jarque-Bera test; spectral density function

Usage

```
## S3 method for class 'FitAR'
plot(x, terse=TRUE, ...)
```

Arguments

| | |
|-------|-----------------------------------------------------------------------|
| x | object of class "FitAR" |
| terse | if TRUE, only one graph is produced, otherwise many diagnostic plots. |
| ... | optional arguments |

Value

No value is returned. Plots are produced as side-effect.

Note

When terse=FALSE, numerous graphs are produced. Turn on recording to be able to page back and forth between them.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[summary.FitAR](#), [FitAR](#), [JarqueBeraTest](#) [RacfPlot](#) [LBQPlot](#)

Examples

```
obj<-FitAR(SeriesA, c(1,2,6,7))
plot(obj)
```

plot.Selectmodel *Subset AR Graph for "Selectmodel" Object*

Description

A graphical depiction is given of the output from SelectModel.

Usage

```
## S3 method for class 'Selectmodel'
plot(x, ...)
```

Arguments

```
x                    out from SelectModel
...                   optional arguments
```

Details

The relative plausibility of Model A vs the best Model B, is defined as $R = e^{(AIC_B - AIC_A)/2}$. Values of R less than 1 R is defined similarly if the BIC/UBIC criterion is used.

Value

No value. Plot produced as side-effect.

Author(s)

A.I. McLeod

See Also

[SelectModel](#)

Examples

```
#takes about 10 seconds
## Not run:
out<-SelectModel(log(Willamette),lag.max=150,ARModel="AR",Best=5,Criterion="AIC")
plot(out)

## End(Not run)
```

PlotARSdf

Plot AR or ARMA Spectral Density

Description

Constructs a plot of the AR spectral density function.

Usage

```
PlotARSdf(phi = NULL, theta = NULL, units = "radial", logSdf = FALSE, InnovationVariance = 1, main = NULL)
```

Arguments

| | |
|--------------------|-----------------------------------------------|
| phi | AR Coefficients |
| theta | MA Coefficients |
| units | default is "radial" |
| logSdf | default is FALSE otherwise log sdf is plotted |
| InnovationVariance | innovation variance, default is 1 |
| main | optional plot title |
| sub | optional subtitle |
| lwd | optional lwd for plot, default lwd=3. |
| col | optional col for plot. Default "blue". |
| plotQ | True, plot otherwise not |
| ... | optional arguments |

Details

The spectral density function is symmetric and defined in $(-\pi, \pi)$ but plotted over $(0, \pi)$. If units are not "radial", it is plotted over $(0, 0.5)$.

Value

Plot is produced using plot. Matrix with 2 columns containing the frequencies and spectral density is returned invisibly.

See Also[ARSdf](#)**Examples**

```
#AR(1)
PlotARSdf(0.8)
#MA(1)
PlotARSdf(theta=0.8)
#ARMA(1,1)
PlotARSdf(0.9,0.5)
#white noise
PlotARSdf()
```

predict.FitAR

Predict Subset AR Model

Description

After fitting we predict at origin times $n, n+1, \dots, n+m$, where m is the length of the vector `newdata` and for lead time series as specified by `n.ahead`.

Usage

```
## S3 method for class 'FitAR'
predict(object, n.ahead = 1, newdata = numeric(0), ...)
```

Arguments

| | |
|----------------------|------------------------|
| <code>object</code> | 'FitAR' object |
| <code>n.ahead</code> | lead time |
| <code>newdata</code> | new time series values |
| <code>...</code> | optional arguments |

Details

The prediction algorithm described in McLeod, Yu and Zinovi (2008) is used.

Value

A list with components

| | |
|--------------------------|--------------------------------------------------------------------------------------|
| <code>Forecasts</code> | matrix with $m+1$ rows and <code>maxLead</code> columns with the forecasts |
| <code>SDForecasts</code> | matrix with $m+1$ rows and <code>maxLead</code> columns with the sd of the forecasts |

Author(s)

A.I. McLeod

References

McLeod AI, Yu H, Zinovi K (2008). Linear Time Series Modeling with R Package. Journal of Statistical Software, 23/5, 1-26.

See Also

[TrenchForecast](#)

Examples

```
## Not run: #these examples take about a minute
#Example 1.
#Compare the predictions for the monthly sunspots using the ARZ
# fitted using the UBIC and BIC.
# This computation takes about 3-4 minutes.

'getRMSE' <-
function(obj, zTOT, n.ahead = 12, newdata=numeric(0)){
  ans<-predict(obj, n.ahead=n.ahead, newdata=newdata)
  ansf<-ans$Forecasts
  nL<-as.numeric(colnames(ansf))
  nO<-as.numeric(rownames(ansf))
  err<-ansf-zTOT[-1+outer(nO,nL,FUN="+")]
  s<-apply(err, MARGIN=2, FUN=rmse)
  s
}

'rmse' <-
function(x){
  y<-x[!is.na(x)]
  sqrt(sum(y^2)/length(y))
}

zTOT <- sqrt(sunspots)
nTOT <- length(zTOT)
nOUT <- 12*3 #using last 3 years for out-of-sample forecasts
ind<- (1:nTOT)<(nTOT-nOUT+1)
newdata<-zTOT[!ind]
z<-zTOT[ind]
lag.max<-12*11 #using lags up to last 11 years in subset model
nahead<-4 #forecasts for 1 to 4 months ahead
pUBIC <- SelectModel(z, ARModel="ARz", lag.max=lag.max, Best=1)
zUBIC <- FitAR(z, pUBIC, ARModel="ARz")
pBIC <- SelectModel(z, ARModel="ARz",lag.max=lag.max,Best=1,Criterion="BIC")
zBIC <- FitAR(z, pBIC, ARModel="ARz")
fubic<-getRMSE(zUBIC, zTOT, n.ahead=nahead, newdata=newdata)
fbic<-getRMSE(zBIC, zTOT, n.ahead=nahead, newdata=newdata)
m<-matrix(c(fubic,fbic), ncol=2)
dimnames(m)<-list(1:nahead, c("fubic","fbic"))
m
#
#Example 2.
```

```

#Compute predictions and plot observed - predicted
z <- sqrt(sunspots)
pUBIC <- SelectModel(z, ARModel="ARz", lag.max=240, Best=1)
zUBIC <- FitAR(z, pUBIC, ARModel="ARz")
out<-predict(zUBIC, n.ahead=24)
zf<-out$Forecasts
zsd<-out$SDForecasts
y<-cts(z, zf)
plot(window(y,start=1980), type="n", ylab="sqrt sunspot number")
y1<-window(y, start=1980, end=1983)
lines.ts(y1,col="blue",type="o", lwd=2, pch=16)
y2<-window(y, start=c(1983,1))
lines.ts(y2,col="red",type="o",lwd=2, pch=16)
legend(1984,12, legend=c("observed", "forecast"),col=c("red","blue"),lwd=c(2,2),pch=c(16,16))

## End(Not run)

```

print.FitAR

Print Method for "FitAR" Object

Description

A terse summary is given.

Usage

```

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

```

Arguments

| | |
|-----|-------------------------|
| x | object of class "FitAR" |
| ... | optional arguments |

Value

A terse summary is displayed

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also[summary.FitAR](#)**Examples**

```
data(SeriesA)
FitAR(SeriesA, c(1,2,6,7))
```

RacfPlot*Residual Autocorrelation Plot*

Description

Residual autocorrelation plot for "FitAR" objects. This plot is useful for diagnostic checking models fit with the function [FitAR](#).

Usage

```
RacfPlot(obj, lag.max = 1000, SquaredQ=FALSE, ylab="")
```

Arguments

| | |
|----------|----------------------------------------------------------------------------------------|
| obj | output from FitAR |
| lag.max | maximum lag. Set to 1000 since minimum of this value and the value in the obj is used. |
| SquaredQ | default is FALSE. For squared residual autocorrelations, set to TRUE |
| ylab | y-axis label |

Details

The standard deviations of the residual autocorrelations are obtained from McLeod (1978, eqn.16) or McLeod and Zhang (2006, eqn.16). Simultaneous confidence bounds are shown and constructed using the Bonferonni approximation as suggested by Hosking and Ravishanker (1993)

Value

Plot is produced as a side-effect. No output

Note

This function is normally invoked when plot.FitAR is used.

Author(s)

A.I. McLeod and Y. Zhang

References

Hosking, J.R.M. and Ravishanker, N. (1993) Approximate simultaneous significance intervals for residual autocorrelations of autoregressive-moving average time series models. *Journal of Time Series Analysis* 14, 19-26.

McLeod, A.I. (1978), On the distribution and applications of residual autocorrelations in Box-Jenkins modelling, *Journal of the Royal Statistical Society B* 40, 296-302.

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

See Also

[plot.FitAR](#), [FitAR](#),

Examples

```
#fit subset AR and plot diagnostic check
data(SeriesA)
out<-FitAR(SeriesA, c(1,2,7), ARModel="ARp")
RacfPlot(out)
#note that plot produces LBQPlot and RacfPlot
plot(out)
#check squared residuals
RacfPlot(out, SquaredQ=TRUE)
```

Readts

Input a Time Series

Description

This function inputs time series stored in ASCII in a format that the first line in the file is a title, next few lines, beginning with a \#, are comments, and the remaining lines contain the data. Here is an example:

```
Changes In Global Temperature, Annual, 1880-1985 #Surface air "temperature
change" for the globe, 1880-1985. #Degrees Celsius. "Temperature change" actually
means temperature #Surface Air Temperature", 'Journal of Geophysical Research',
Vol. 92, -.40 -.37 -.43 .... .27 .42 .02 .30 .09 .05
```

Usage

```
Readts(file = "", freq = 1, start = 1, VerboseQ=TRUE)
```

Arguments

| | |
|----------|------------------------------------------------------------------|
| file | location for input file |
| freq | tsp parameter, =1, annual, =12 monthly etc |
| start | tsp parameter |
| VerboseQ | normally prompt for arguments but set VerboseQ=FALSE to automate |

Value

ts object with attribute 'title'

Author(s)

A.I. McLeod

See Also

[scan](#), [ts](#),

Examples

```
#You will need to change save the data given above in a file
#and change the directory as appropriate
#z<-Readts(file="d:/datasets/mhsets/annual/globtp.1", start=1880, VerboseQ=FALSE)
```

| | |
|-----------------|----------------------------------------------|
| residuals.FitAR | <i>Extract Residuals from "FitAR" Object</i> |
|-----------------|----------------------------------------------|

Description

Method function.

Usage

```
## S3 method for class 'FitAR'
residuals(object, ...)
```

Arguments

| | |
|--------|-------------------------|
| object | object of class "FitAR" |
| ... | optional arguments |

Value

Vector of residuals

Author(s)

A.I. McLeod

See Also[FitAR](#),**Examples**

```
out<-FitAR(SeriesA, c(1,2,6,7))
resid(out)
```

sdfplot*Autoregressive Spectral Density Estimation*

Description

Generic function. Methods are available for "FitAR", "ar", "Arima", "ts" and "numeric".

Usage

```
sdfplot(obj, ...)
```

Arguments

| | |
|-----|--------------------|
| obj | input object |
| ... | optional arguments |

Value

Plot is produced using plot. Matrix with 2 columns containing the frequencies and spectral density is returned invisibly.

Author(s)

A.I. McLeod

See Also[sdfplot](#), [FitAR](#)**Examples**

```
#Example 1
#Use AIC to select best subset model to fit to lynx data and
#plot spectral density function
pvec<-SelectModel(SeriesA, ARModel="ARp", lag.max=10, Best=1)
ans<-FitAR(SeriesA, pvec)
sdfplot(ans)
#
#Example 2
#Fit ARMA and plot sdf
```

```

ans<-arima(SeriesA, c(1,0,1))
sdfplot(ans)
#
#Example 3
#Fit ARz model using AIC to monthly sunspots and plot spectral density
#Warning: this may take 10 minutes or so.
## Not run:
pvec<-SelectModel(sunspots, lag.max=200, ARModel="ARz", Criterion="AIC", Best=1)
ans<-FitAR(sunspots, pvec)
sdfplot(ans)

## End(Not run)

```

sdfplot.ar

Autoregressive Spectral Density Estimation for "ar"

Description

Method for class "ar" for sdfplot.

Usage

```

## S3 method for class 'ar'
sdfplot(obj, ...)

```

Arguments

| | |
|-----|-----------------------------------|
| obj | class "ar" object, output from ar |
| ... | optional arguments |

Value

Plot is produced using plot. Matrix with 2 columns containing the frequencies and spectral density is returned invisibly.

Author(s)

A.I. McLeod

See Also

[ARSdf](#), [sdfplot](#), [sdfplot.FitAR](#), [sdfplot.ts](#)

Examples

```

#Fit AR(p) using AIC model selection and Burg estimates. Plot spectral density
#estimate
ans<-ar(lynx, lag.max=20)
sdfplot(ans)

```

sdfplot.Arima *Spectral Density of Fitted ARIMA Model*

Description

Method for class "Arima" for sdfplot.

Usage

```
## S3 method for class 'Arima'
sdfplot(obj, ...)
```

Arguments

obj object class "Arima"
 ... optional arguments

Value

Plot is produced using plot. Matrix with 2 columns containing the frequencies and spectral density is returned invisibly.

Author(s)

A.I. McLeod

See Also

[ARSdf](#), [sdfplot](#), [sdfplot.FitAR](#), [sdfplot.ts](#)

Examples

```
sdfplot(SeriesA,c(1,0,1))
```

sdfplot.FitAR *Autoregressive Spectral Density Estimation for "FitAR"*

Description

Methods function for sdfplot. Autoregressive spectral density function estimation using the result output from FitAR.

Usage

```
## S3 method for class 'FitAR'
sdfplot(obj, ...)
```

Arguments

obj object, class"FitAR"
 ... optional arguments

Value

Plot is produced using plot. Matrix with 2 columns containing the frequencies and spectral density is returned invisibly.

Author(s)

A.I. McLeod

See Also

[sdfplot](#), [FitAR](#)

Examples

```
#Use AIC to select best subset model to fit to lynx data and
#plot spectral density function
pvec<-SelectModel(SeriesA, ARModel="ARp", lag.max=10, Best=1)
ans<-FitAR(SeriesA, pvec)
sdfplot(ans)
#
#plot sdf and put your own title
z<-c(SeriesA)
pvec<-SelectModel(z, ARModel="ARp", lag.max=10, Best=1)
ans<-FitAR(z, pvec)
sdfplot(ans)
title(main="Example SDF")
```

sdfplot.numeric

Autoregressive Spectral Density Estimation for "numeric"

Description

Method function for vectors, class "numeric"

Usage

```
## S3 method for class 'numeric'
sdfplot(obj, ...)
```

Arguments

obj object, class"numeric", a vector
 ... optional arguments

Value

Plot is produced using plot. Matrix with 2 columns containing the frequencies and spectral density is returned invisibly.

Author(s)

A.I. McLeod

See Also

[sdfplot](#)

Examples

```
sdfplot(lynx)
```

sdfplot.ts

Autoregressive Spectral Density Estimation for "ts" Object

Description

Methods function for "ts".

Usage

```
## S3 method for class 'ts'  
sdfplot(obj, ...)
```

Arguments

| | |
|-----|--------------------|
| obj | object, class"ts" |
| ... | optional arguments |

Value

Plot is produced using plot. Matrix with 2 columns containing the frequencies and spectral density is returned invisibly.

Author(s)

A.I. McLeod

See Also

[sdfplot](#)

Examples

```
data(SeriesA)  
sdfplot(SeriesA)
```

| | |
|-------------|-----------------------------------------|
| SelectModel | <i>Select Best AR, ARz or ARp Model</i> |
|-------------|-----------------------------------------|

Description

The AIC/BIC/UBIC/EBIC/BICq criterion is used to select the best fitting AR or subset AR model. When Best > 1, the result may be plotted using plot.

Usage

```
SelectModel(z, lag.max = 15, ARModel = c("AR", "ARz", "ARp"), Criterion = "default", Best = 3, Candidates
```

Arguments

| | |
|------------|--------------------------------------------------------------------------------------------------------------------------------|
| z | time series data |
| lag.max | maximum order of autoregression |
| ARModel | "AR" for full AR(p) or "ARp"/"ARz" corresponding to subset models |
| Criterion | default is "BIC" for order selection and "BICq" for subset selection. Options: "AIC", "BIC", "UBIC", "EBIC", "BICq" and "GIC". |
| Best | final number of models to be selected |
| Candidates | number of models initially selected using the approximate criterion |
| t | tuning parameter, EBIC, BICq, GIC |

Details

McLeod and Zhang (2006) outline an approximate AIC/BIC selection algorithm. This algorithm is a refinement of that method. The refinement consists of automatically look for the best k candidates, where k = Candidates. Then the exact likelihood is evaluated for all k candidates. Out of these k candidates, the best q = Best are then selected. This two-step procedure is needed because if k is too low, the approximate AIC/BIC rankings may not agree with the exact rankings. This strategy is used for model selection for AR, ARz and ARp models. A plot method is available to graph the output. The UBIC and EBIC developed by Chen and Chen (2007) are an extension of the BIC criterion for subset selection. In the non-subset case UBIC is equivalent to BIC. The EBIC using a tuning parameter, G, where $0 \leq G \leq 1$. The BICq takes a tuning parameter, Q, where $0 < Q < 1$. The GIC takes a tuning parameter, p, where $0 < p < 0.25$.

Value

When Best = 1, a vector is returned indicated the lag or lags included in the model. The null model is indicated by returning 0 for the lag. An object with class "Selectmodel" is returned when Best > 1. If ARModel = "AR", a matrix is return whose first column shows p and second AIC or BIC. Otherwise for subset selection, the result is a list with q components, where q=BEST. When Criterion = "UBIC", the components in this list are:

| | |
|------|--------------------------------------------|
| p | lags present, a 0 indicates the null model |
| UBIC | exact UBIC |

similarly for the AIC/BIC case.

The components are arranged in order of the criterion used.

When ARModel = "ARp" or "ARz", an attribute "model" indicating "ARp" or "ARz" is included.

Warning

Setting Candidates too low can result in anomalous results. For example if Candidates = 1, we find that the top ranking model may depend on how large Best is set. This phenomenon is due to the fact that among the best AIC/BIC models there is sometimes very little difference in their AIC/BIC scores. Since the initial ranking of the Candidates is done using the approximate likelihood, the final ranking using the exact likelihood may change.

Note

For white noise, the best model is the null model, containing no lags. This is indicating by setting the model order, $p = 0$.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial Autocorrelation Parameterization for Subset Autoregression. *Journal of Time Series Analysis*, 27, 599-612.

Chen, J. and Chen, Z. (2008). Extended Bayesian Information Criteria for Model Selection with Large Model Space. *Biometrika*.

See Also

[plot.Selectmodel](#), [PacfPlot](#), [PacfPlot](#), [FitAR](#)

Examples

```
#Example 1: Find an ARp subset model for lynx data using BIC
z<-log(lynx)
out<-SelectModel(z, ARModel="ARp", Criterion="BIC", Best=5)
plot(out)
#
#Example 2: Find an ARz subset model for lynx data using BIC
out<-SelectModel(z, ARModel="ARz", Criterion="BIC", Best=5)
plot(out)
#
#Example 3: Select an AR(p) model
out<-SelectModel(z, ARModel="AR", Criterion="BIC", Best=5)
out
plot(out)
out<-SelectModel(z, ARModel="AR", Criterion="BIC", Best=1)
#
#Example 4: Fit subset models to lynx series
```

```

z<-log(lynx)
#requires library leaps. Should be automatically when FitAR package is loaded.
#first fit ARp
pvec <- SelectModel(z, lag.max=11, ARModel="ARp", Criterion="AIC", Best=1)
ans1 <- FitAR(z, pvec, ARModel="ARp", MLEQ=FALSE)
#now fit ARz
pvec <- SelectModel(z, lag.max=11, ARModel="ARz", Criterion="AIC", Best=1)
ans2<-FitAR(z, pvec, ARModel="ARz")
#compare
summary(ans1)
summary(ans2)
#Use UBIC
pvec <- SelectModel(z, ARModel="ARp",lag.max=11,Best=1)
ans3<-FitAR(z, pvec, ARModel="ARp")
pvec <- SelectModel(z, ARModel="ARz",lag.max=11,Best=1)
ans4<-FitAR(z, pvec, ARModel="ARz")
#compare
summary(ans3)
summary(ans4)
#
#Example 5: lynx data subset AR models
#The AIC and BIC choose the same models as the GIC with t=0.1 and t=0.01 respectively.
#An even more parsimonious model is chosen with t=0.001
SelectModel(z, lag.max=15, ARModel="ARp", Criterion="GIC", Best=1, Candidates=5, t=0.1)
SelectModel(z, lag.max=15, ARModel="ARp", Criterion="GIC", Best=1, Candidates=5, t=0.01)
SelectModel(z, lag.max=15, ARModel="ARp", Criterion="GIC", Best=1, Candidates=5, t=0.001)
ans<-SelectModel(z, lag.max=15, ARModel="ARp", Criterion="GIC", Best=3, Candidates=5, t=0.001)
plot(ans)

```

SeriesA

*Series A, Chemical Process Concentration Readings***Description**

Chemical process concentration readings for every 2 hours.

Usage

```
data(SeriesA)
```

Format

ts object with attribute "title"

Details

Box and Jenkins (1970) fit an ARMA(1,1) and ARIMA(0,1,1) to this series. Cleveland(1971) suggested a subset AR(1,2,7). McLeod and Zhang (2006) fit a subset ARz(1,2,6,7) parameterized with the partial autocorrelations.

Source

Box and Jenkins (1970). Time Series Analysis: Forecasting and Control.

References

Cleveland, W.S. (1971) The inverse autocorrelations of a time series and their applications. *Technometrics* 14, 277-298.

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, 27, 599-612.

Examples

```
data(SeriesA)
#fit subset models
FitAR(SeriesA, c(1,2,7), ARModel="ARp")
FitAR(SeriesA, c(1,2,6,7), ARModel="ARz")
```

SeriesB

Series B

Description

closing price of common stock, daily, May 17 1961 to November 2 1962

Usage

```
data(SeriesB)
```

Format

The format is: num [1:369] 460 457 452 459 462 459 463 479 493 490 ...

Source

Box and Jenkins (1970). Time Series Analysis.

Examples

```
data(SeriesB)
plot(SeriesB)
```

| | |
|----------|-------------------------------------|
| SeriesB2 | <i>IBM Stock Prices, 2nd series</i> |
|----------|-------------------------------------|

Description

Closing price of common stock, daily, June 29 1959 to June 30 1960

Usage

```
data(SeriesB2)
```

Format

The format is: num [1:255] 445 448 450 447 451 453 454 454 459 440 ...

Source

Box and Jenkins (1970). Time Series Analysis.

Examples

```
data(SeriesB2)
TimeSeriesPlot(SeriesB2)
```

| | |
|----------------|--------------------------------------------------------|
| SiddiquiMatrix | <i>Covariance Matrix of MLE Parameters in an AR(p)</i> |
|----------------|--------------------------------------------------------|

Description

A direct method of computing the inverse of the covariance matrix of p successive observations in an AR(p) with unit innovation variance given by Siddiqui (1958) is implemented. This matrix, divided by $n = \text{length of series}$, is the covariance matrix for the MLE estimates in a regular AR(p).

Usage

```
SiddiquiMatrix(phi)
```

Arguments

phi coefficients in a regular AR(p)

Value

Matrix, covariance matrix of MLE estimates

Note

No check on whether the parameters are in the stationary region is done. It has been shown a necessary and sufficient condition for the parameters to be in the stationary region is that this matrix should be positive-definite (Pagano, 1973). But computationally it is probably better to test for stationarity by using [ARToPacf](#) to transform to the PACF and then check that the absolute value of all partial autocorrelations are less than 1.

Author(s)

A.I. McLeod

References

Siddiqui, M.M. (1958) On the inversion of the sample covariance matrix in a stationary autoregressive process. *Annals of Mathematical Statistics* 29, 585-588.

Pagano, M. (1973), When is an autoregressive scheme stationary? *Communications in Statistics A* 1, 533-544.

See Also

[FitAR](#)

Examples

```
#compute the inverse directly and by Siddiqui's method and compare:
phi<-PacfToAR(rep(0.8,5))
A<-SiddiquiMatrix(phi)
B<-solve(toeplitz(TacvfAR(phi, lag.max=length(phi)-1)))
max(abs(A-B))
```

SimulateGaussianAR *Autoregression Simulation*

Description

Simulate a mean-zero stationary Gaussian AR(p) time series.

Usage

```
SimulateGaussianAR(phi, n = 100, InnovationVariance = 1)
```

Arguments

| | |
|--------------------|-----------------------------------|
| phi | vector containing AR coefficients |
| n | length of time series |
| InnovationVariance | innovation variance |

Details

The p initial values are simulated using the appropriate multivariate distribution as was suggested in McLeod (1975). The R function `rnorm()` is used.

Value

A vector of length n , the simulated series

Note

If the process is non-stationary, then random initial values are used determined by the first p innovations.

Author(s)

A.I. McLeod

References

McLeod, A.I. (1975), Derivation of the theoretical autocorrelation function of autoregressive moving-average time series, *Applied Statistics* **24**, 255–256. Percival, D.B. and Walden, A.T. (1993), *Spectral Analysis for Physical Applications*.

See Also

[Boot.FitAR](#)

Examples

```
#Percival and Walden (1993, p.46) illustrated a time series with a
#very peaked spectrum with the AR(4) with coefficients
#c(2.7607,-3.8106,2.6535,-0.9238) with NID(0,1) innovations.
#
z<-SimulateGaussianAR(c(2.7607,-3.8106,2.6535,-0.9238),1000)
library(lattice)
TimeSeriesPlot(z)
```

summary.FitAR

Summary Method for "FitAR" Object

Description

summary for "FitAR" object.

Usage

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

Arguments

| | |
|--------|--------------------|
| object | "FitAR" object |
| ... | optional arguments |

Value

A printed summary is given

Author(s)

A.I. McLeod

See Also

[print.FitAR](#), [FitAR](#)

Examples

```
data(SeriesA)
out<-FitAR(SeriesA, c(1,2,6,7))
summary(out)
```

TacvfAR

Theoretical Autocovariance Function of AR

Description

The theoretical autocovariance function of an AR(p) with unit variance is computed. This algorithm has many applications. In this package it is used for the computation of the information matrix, in simulating p initial starting values for AR simulations and in the computation of the exact mle for the mean.

Usage

```
TacvfAR(phi, lag.max = 20)
```

Arguments

| | |
|---------|----------------------------------------------|
| phi | vector of AR coefficients |
| lag.max | computes autocovariances lags 0,1,...,maxlag |

Details

The algorithm given by McLeod (1975) is used.

The built-in R function ARMAacf could also be used but it is quite complicated and apart from the source code, the precise algorithm used is not described. The only reference given for ARMAacf is the Brockwell and Davis (1991) but this text does not give any detailed exact algorithm for the general case.

Another advantage of TacvfAR over ARMAacf is that it will be easier for to translate and implement this algorithm in other computing environments such as MatLab etc. since the code is entirely written in R.

Value

Vector of length = (lag.max+1) containing the autocovariances at lags 0,...,lag.max is returned.

Author(s)

A.I. McLeod

References

McLeod, A.I. (1975), Derivation of the theoretical autocorrelation function of autoregressive moving-average time series. Applied Statistics, 24, 255-256.

See Also

[ARMAacf](#), [InformationMatrixAR](#), [GetARMeanMLE](#), [SimulateGaussianAR](#)

Examples

```
#calculate and plot the autocorrelations from an AR(2) model
# with parameter vector c(1.8,-0.9).
g<-TacvfAR(c(1.8,-0.9),20)
AcfPlot(g/g[1], LagZeroQ=FALSE)
```

TacvfMA

Theoretical Autocovariances for Moving Average Process

Description

The theoretical autocovariance function of a MA(q) with unit variance is computed.

Usage

```
TacvfMA(theta, lag.max = 20)
```

Arguments

theta q parameters in MA(q)
lag.max number of lags required.

Details

The first $q+1$ values are determined using a matrix multiplication - avoiding a loop. The remaining values set to zero.

Value

Vector of length $q+1$ containing the autocovariances at lags 0,1,...,lag.max

Note

See Details in [TacfFAR](#) for why we prefer to use this algorithm instead of [ARMAacf](#)

Author(s)

A.I.McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006), Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, **27**, 599-612.

See Also

[ARMAacf](#), [TacfFAR](#)

Examples

```
TacfFMA(c(1.8,-0.9), 10)
```

| | |
|----------------|-------------------------------------------------------------------------------|
| TimeSeriesPlot | <i>Multi-Panel or Single-Panel Time Series Plot with Aspect-Ratio Control</i> |
|----------------|-------------------------------------------------------------------------------|

Description

Cleveland (1993) pointed out that the aspect-ratio is important in graphically showing the rate-of-change or shape information. For many time series, it is preferably to set this ratio to 0.25 than the default. In general, Cleveland (1993) shows that the best choice of aspect-ratio is often obtained by if the average apparent absolute slope in the graph is about 45 deg. But for many stationary time series, this would result in an aspect-ratio which would be too small. As a compromise we have chosen a default of 0.25 but the user can select other choices.

Usage

```
TimeSeriesPlot(z, SubLength = Inf, aspect = 0.25, type="l", xlab = "Observation Number",  
ylab=NULL, main=NULL, ...)
```

Arguments

| | |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>z</code> | ts object or vector, time series data |
| <code>SubLength</code> | maximum number of data points per panel. Default <code>SubLength=Inf</code> and regular graphics. For trellis graphics, set <code>SubLength</code> to a finite value. |
| <code>aspect</code> | optional setting for the aspect-ratio |
| <code>type</code> | plot type, default <code>type="l"</code> join points with lines |
| <code>xlab</code> | label for horizontal axis |
| <code>ylab</code> | optional label for vertical axis |
| <code>main</code> | optional title |
| <code>...</code> | optional arguments passed to <code>xypLOT</code> |

Details

If `z` has attribute "title" containing a character string, this is used on the plot. Time series input using the function [Readts](#) always have this attribute set.

Value

If `SubLength` is finite, the lattice package is used and a graphic object of class `trellis` is produced. Otherwise, the standard R graphics system is used and the plot is produced as a side-effect and there is no output.

Note

Requires `lattice` library

Author(s)

A.I. McLeod

References

W.S. Cleveland (1993), *Visualizing Data*.

See Also

[plot.ts](#), [Readts](#)

Examples

```
#from built-in datasets
TimeSeriesPlot(AirPassengers)
title(main="Monthly number of trans-Atlantic airline passengers")
#
#compare plots for lynx series
plot(lynx)
TimeSeriesPlot(lynx, type="o", pch=16, ylab="# pelts", main="Lynx Trappings")
#
#lattice style plot
data(Ninemile)
TimeSeriesPlot(Ninemile, SubLength=200)
```

UnitRootTest

*Unit Root Test***Description**

Unit root test. Test $H_0: \rho=1$ vs. $H_1: \rho<1$ in the model $z[t] = \text{const} + \rho*z[t-1] + a[t]$.

Usage

```
UnitRootTest(z, method = c("MLE", "ExactMLE", "LS", "All"), statistic = c("Z", "T"), NumBoot = 1000, PValueMethod)
```

Arguments

| | |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>z</code> | time series |
| <code>method</code> | estimation methods |
| <code>statistic</code> | normalized rho or t-statistic |
| <code>NumBoot</code> | number of bootstrap iterations |
| <code>PValueMethod</code> | p-value can be estimated either as $(k+1)/(N+1)$ as recommended by Davison and Hinkley (p. 148) or as k/N as in Efron and Tibsharini (p. 221, Algorithm 16.1). |

Details

Bootstrap unit root tests

Value

one-sided P-value

Author(s)

A.I. McLeod

References

- Davison, A.C. and Hinkley, D.V. (1997). *Bootstrap Methods and their Application*. Cambridge.
- Efron, B. and Tibshirani, R. (1993). *An Introduction to the Bootstrap*. Chapman/Hall.
- Yu, H., Zhang, Y. and McLeod, A.I. (2009). *Unit Root Bootstrap Tests with Exact Maximum Likelihood*.

See Also

[PP.test](#)

Examples

```
## Not run: #takes about 10 seconds
z<-cumsum(rnorm(100))
UnitRootTest(z)

## End(Not run)
```

USTobacco

U.S. Tobacco Production, 1871-1984

Description

Annual U.S. tobacco production, 1871-1984, in millions of pounds.

Usage

```
data(USTobacco)
```

Format

The format is: Time-Series [1:114] from 1871 to 1984: 327 385 382 217 609 466 621 455 472 469
... - attr(*, "title")= chr "Tobacco production,US, 1871-1984"

Details

Wei (2006, p.120, Example 6.6) fits an ARIMA(0,1,1) to the logarithms. But a more accurate Box-Cox analysis indicates a square-root transformation should be used. A more complex ARIMA-GARCH model is also suggested by Wei (2006).

Source

Wei, W.W.S. (2006, Series W6, p.570), *Time Series Analysis: Univariate and Multivariate Methods*. 2nd Ed., New York: Addison-Wesley.

Examples

```
#From a plot of the series, we see that the variance is increasing with level.
#From the acf of the first differences an ARIMA(0,1,1) is suggested.
data(USTobacco)
# layout(matrix(c(1,2,1,2),ncol=2))
plot(USTobacco)
lines(lowess(time(USTobacco), USTobacco), lwd=2, col="blue")
acf(diff(USTobacco, differences=1))
```

VarianceRacfAR

Covariance Matrix Residual Autocorrelations for AR

Description

Computes the variance-covariance matrix for the residual autocorrelations in an AR(p).

Usage

```
VarianceRacfAR(phi, MaxLag, n)
```

Arguments

| | |
|--------|------------------------------------------------------------------------------------------------|
| phi | vector of AR coefficients |
| MaxLag | covariance matrix for residual autocorrelations at lags 1 ..., m, where m = MaxLag is computes |
| n | length of time series |

Details

The covariance matrix for the residual autocorrelations is derived in McLeod (1978, eqn. 15) for the general ARMA case. With this function one can obtain the standard deviations of the residual autocorrelations which can be used for diagnostic checking with [RacfPlot](#).

Value

The m-by-m covariance matrix of residual autocorrelations at lags 1, ..., m, where m = MaxLag.

Note

The derivation assumes normality of the innovations, mle estimation of the parameters and a known mean-zero time series. It is easily seen that the same result still holds for IID innovations with mean zero and finite variance, any first-order efficient estimates of the parameters including the AR coefficients and mean.

Author(s)

A.I. McLeod

References

McLeod, A.I. (1978), On the distribution and applications of residual autocorrelations in Box-Jenkins modelling, *Journal of the Royal Statistical Society B*, **40**, 296–302

See Also

[VarianceRacfARp](#), [VarianceRacfARz](#), [RacfPlot](#)

Examples

```
VarianceRacfAR(0.5,5,100)
```

| | |
|-----------------|------------------------------------------------------------|
| VarianceRacfARp | <i>Covariance Matrix Residual Autocorrelations for ARp</i> |
|-----------------|------------------------------------------------------------|

Description

The ARp subset model is defined by taking a subset of the parameters in the regular AR(p) model. With this function one can obtain the standard deviations of the residual autocorrelations which can be used for diagnostic checking with [RacfPlot](#).

Usage

```
VarianceRacfARp(phi, lags, MaxLag, n)
```

Arguments

| | |
|--------|---------------------------------------------------------------------------------------------|
| phi | vector of AR coefficients |
| lags | lags in subset AR |
| MaxLag | covariance matrix for residual autocorrelations at lags 1,...,m, where m=MaxLag is computes |
| n | length of time series |

Details

The covariance matrix for the residual autocorrelations is derived in McLeod (1978, eqn. 15) for the general ARMA case. McLeod (1978, eqn. 35) specializes this result to the subset case.

Value

The m-by-m covariance matrix of residual autocorrelations at lags 1, . . . , m, where m = MaxLag.

Author(s)

A.I. McLeod

References

McLeod, A.I. (1978), On the distribution and applications of residual autocorrelations in Box-Jenkins modelling, *Journal of the Royal Statistical Society B*, **40**, 296-302.

See Also

[VarianceRacfAR](#), [VarianceRacfARz](#), [RacfPlot](#)

Examples

```
#the standard deviations of the first 5 residual autocorrelations
#to a subset AR(1,2,6) model fitted to Series A is
v<-VarianceRacfARp(c(0.36,0.23,0.23),c(1,2,6), 5, 197)
sqrt(diag(v))
```

VarianceRacfARz

Covariance Matrix Residual Autocorrelations for ARz

Description

The ARz subset model is defined by taking a subset of the partial autocorrelations (zeta parameters) in the AR(p) model. With this function one can obtain the standard deviations of the residual autocorrelations which can be used for diagnostic checking with [RacfPlot](#).

Usage

```
VarianceRacfARz(zeta, lags, MaxLag, n)
```

Arguments

| | |
|--------|---------------------------------------------------------------------------------------------|
| zeta | zeta parameters (partial autocorrelations) |
| lags | lags in model |
| MaxLag | covariance matrix for residual autocorrelations at lags 1,...,m, where m=MaxLag is computes |
| n | length of time series |

Details

The covariance matrix of the residual autocorrelations in the subset ARz case is derived in McLeod and Zhang (2006, eqn. 16)

Value

The m-by-m covariance matrix of residual autocorrelations at lags 1, . . . , m, where m = MaxLag.

Author(s)

A.I. McLeod and Y. Zhang

References

McLeod, A.I. and Zhang, Y. (2006). Partial autocorrelation parameterization for subset autoregression. *Journal of Time Series Analysis*, **27**, 599-612.

See Also

[VarianceRacfAR](#), [VarianceRacfARz](#), [RacfPlot](#)

Examples

```
#the standard deviations of the first 5 residual autocorrelations
#to a subset AR(1,2,6) model fitted to Series A is
v<-VarianceRacfARp(c(0.36,0.23,0.23),c(1,2,6), 5, 197)
sqrt(diag(v))
```

Willamette

Willamette Riverflow Time Series

Description

Monthly flows of the Willamette River, Salem, Oregon, Oct. 1950 - Aug. 1983. Percival and Walden (1993, Ch. 10.15) fit high-order AR models to estimate the spectral density function.

Usage

```
data(Willamette)
```

Format

The format is: Time-Series [1:395] from 1951 to 1984: 8.95 9.49 10.19 10.96 11.08 ... - attr(*, "title")= chr "Willamette river, Monthly, Salem, Oregon, Oct. 1950 - Aug. 1983"

Source

<http://faculty.washington.edu/dbp/sapabook.html>

References

Percival, D.B. and Walden, A.T. (1993), *Spectral Analysis for Physical Applications*. Cambridge University Press.

Examples

```
#Percival and Walden (1993) fit an AR(27).
#Compare spectral densities with subset AR's.
data(Willamette)
pmax<-27
sdfplot(FitAR(log(Willamette), pmax))
p<-SelectModel(log(Willamette), ARModel="ARz", lag.max=pmax, Best=1)
sdfplot(FitAR(log(Willamette), p))
p<-SelectModel(log(Willamette), ARModel="ARp", lag.max=pmax, Best=1)
sdfplot(FitAR(log(Willamette), p), ARModel="ARp", MLEQ=FALSE)
```

Index

- *Topic **array**
 - FromSymmetricStorageUpper, 46
- *Topic **datasets**
 - Caffeine, 30
 - Commodities, 33
 - FXRates, 46
 - Ninemile, 71
 - SeriesA, 91
 - SeriesB, 92
 - SeriesB2, 93
 - USTobacco, 101
 - Willamette, 105
- *Topic **htest**
 - getRho, 58
 - getT, 58
 - JarqueBeraTest, 66
 - UnitRootTest, 100
- *Topic **manip**
 - glog, 59
- *Topic **package**
 - FitAR-package, 4
- *Topic **ts**
 - AcfPlot, 9
 - AR1Est, 11
 - ARSdf, 12
 - ARToMA, 13
 - ARToPacf, 14
 - BackcastResidualsAR, 15
 - BICqLL, 16
 - Boot, 18
 - Boot.FitAR, 19
 - Boot.ts, 20
 - BoxCox, 22
 - BoxCox.Arima, 23
 - BoxCox.FitAR, 24
 - BoxCox.numeric, 26
 - BoxCox.ts, 27
 - bxcx, 29
 - ChampernowneD, 31
 - coef.FitAR, 32
 - cts, 34
 - DetAR, 35
 - FastLoglikelihoodAR, 36
 - FitAR, 37
 - FitAR-package, 4
 - FitARp, 40
 - FitARz, 43
 - fitted.FitAR, 45
 - FromSymmetricStorageUpper, 46
 - Get1G, 47
 - GetARMeanMLE, 48
 - GetB, 49
 - GetFitAR, 49
 - GetFitARpLS, 51
 - GetFitARpMLE, 52
 - GetFitARz, 54
 - GetKappa, 55
 - GetLeapsAR, 55
 - getRho, 58
 - getT, 58
 - InformationMatrixAR, 60
 - InformationMatrixARp, 61
 - InformationMatrixARz, 62
 - InvertibleQ, 63
 - Jacobian, 64
 - JacobianK, 65
 - LBQPlot, 67
 - LjungBoxTest, 68
 - LoglikelihoodAR, 69
 - PacfDL, 71
 - PacfPlot, 73
 - PacfToAR, 74
 - plot.FitAR, 75
 - plot.Selectmodel, 76
 - PlotARSdf, 77
 - predict.FitAR, 78
 - print.FitAR, 80
 - RacfPlot, 81

- Readts, 82
 - residuals.FitAR, 83
 - sdfplot, 84
 - sdfplot.ar, 85
 - sdfplot.Arima, 86
 - sdfplot.FitAR, 86
 - sdfplot.numeric, 87
 - sdfplot.ts, 88
 - SelectModel, 89
 - SiddiquiMatrix, 93
 - SimulateGaussianAR, 94
 - summary.FitAR, 95
 - TacvFAR, 96
 - TacvFMA, 97
 - TimeSeriesPlot, 98
 - UnitRootTest, 100
 - VarianceRacFAR, 102
 - VarianceRacFARp, 103
 - VarianceRacFARz, 104
- acf, 10, 72
 AcfPlot, 9
 ar, 72
 ar.burg, 74
 AR1Est, 4, 11
 arima, 24
 ARMAacf, 97, 98
 ARSdf, 12, 78, 85, 86
 ARToMA, 13
 ARToPacf, 14, 63, 94
 as.zooreg, 34
- BackcastResidualsAR, 15
 BICqLL, 16
 Boot, 4, 18, 20
 Boot.FitAR, 19, 19, 21, 95
 Boot.ts, 20
 Box.test, 69
 BoxCox, 4, 22, 24, 25, 29
 BoxCox.Arima, 22, 23, 25, 27, 28
 BoxCox.FitAR, 22, 24, 24, 27, 28
 BoxCox.numeric, 22, 26, 28
 BoxCox.ts, 22, 27, 27
 bxcx, 29, 60
- Caffeine, 30
 ChampernowneD, 31, 37
 coef, 4
 coef.FitAR, 32
- Commodities, 33
 cts, 4, 34
- DetAR, 35
- FastLoglikelihoodAR, 31, 36, 36, 70
 FitAR, 4, 15, 31, 37, 42, 44, 45, 50–53, 55, 61, 63, 67, 76, 81, 82, 84, 87, 90, 94, 96
 FitAR-package, 4
 FitARp, 39, 40, 42, 44, 51–53, 55, 62
 FitARz, 39, 42, 43, 52, 53, 55
 fitted, 4
 fitted.FitAR, 45
 FromSymmetricStorageUpper, 46
 FXRates, 46
- Get1G, 47
 GetARMeanMLE, 48, 48, 97
 GetB, 49
 GetFitAR, 49
 GetFitARpLS, 51, 57
 GetFitARpMLE, 39, 42, 44, 52, 52, 55
 GetFitARz, 11, 31, 39, 42, 44, 52, 53, 54
 GetKappa, 55
 GetLeapsAR, 55
 getRho, 58, 59
 getT, 58, 58
 glog, 59
- InformationMatrixAR, 60, 62, 63, 97
 InformationMatrixARp, 61, 61, 63
 InformationMatrixARz, 61, 62, 62, 65
 InvertibleQ, 13–15, 63, 75
- Jacobian, 64, 65
 JacobianK, 65
 JarqueBeraTest, 4, 66, 76
- LBQPlot, 4, 67, 76
 leaps, 57
 LjungBoxTest, 68
 LoglikelihoodAR, 37, 69
 lsfit, 51
- mean, 48
- Ninemile, 71
- pacf, 74
 PacFDL, 71, 72

- PacfPlot, 4, 73, 90
- PacfToAR, 14, 74, 75
- plot.FitAR, 4, 67, 75, 82
- plot.Selectmodel, 76, 90
- plot.ts, 99
- PlotARSdf, 77
- PP.test, 101
- predict, 4
- predict.FitAR, 78
- print, 4
- print.FitAR, 80, 96

- RacfPlot, 4, 39, 42, 44, 52, 53, 55, 76, 81, 102–105
- Readts, 4, 82, 99
- residuals, 4
- residuals.FitAR, 83

- scan, 83
- sdfplot, 4, 84, 84–88
- sdfplot.ar, 85
- sdfplot.Arima, 86
- sdfplot.FitAR, 85, 86, 86
- sdfplot.numeric, 87
- sdfplot.ts, 85, 86, 88
- SelectModel, 4, 17, 57, 76, 89
- SeriesA, 91
- SeriesB, 92
- SeriesB2, 93
- SiddiquiMatrix, 93
- SimulateGaussianAR, 4, 20, 94, 97
- spec.ar, 12
- spec.pgram, 12
- spectrum, 12
- start, 34
- summary, 4
- summary.FitAR, 76, 81, 95

- TacvfAR, 61, 96, 98
- TacvfMA, 97
- TimeSeriesPlot, 4, 98
- TrenchForecast, 79
- ts, 34, 83

- UnitRootTest, 58, 59, 100
- USTobacco, 101

- VarianceRacfAR, 102, 104, 105
- VarianceRacfARp, 103, 103

- VarianceRacfARz, 103, 104, 104, 105
- Willamette, 105
- window, 34