Package ‘fracdiff’

February 19, 2015

Version 1.4-2

Date 2012-12-01

Title Fractionally differenced ARIMA aka ARFIMA(p,d,q) models

Author S original by Chris Fraley, U.Washington, Seattle. R port by Fritz Leisch at TU Wien; since 2003-12: Martin Maechler; fdGPH(), fdSperio(), etc by Valderio Reisen and Artur Lemonte.

Maintainer Martin Maechler <maechler@stat.math.ethz.ch>

Description Maximum likelihood estimation of the parameters of a fractionally differenced ARIMA(p,d,q) model (Haslett and Raftery, Appl.Statistics, 1989).

Suggests longmemo, urca

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-12-02 07:08:12

NeedsCompilation yes

R topics documented:

confint.fracdiff ......................................................... 2
diffseries .............................................................. 3
fdGPH ................................................................. 4
fdSperio ............................................................... 5
fracdiff ................................................................. 6
fracdiff-methods ...................................................... 8
fracdiff.sim .......................................................... 9
fracdiff.var ......................................................... 11

Index 13
Description

Computes confidence intervals for one or more parameters in a fitted fracdiff model, see \texttt{fracdiff}.

Usage

\texttt{CC sS method for class 'fracdiff'
confint(object, parm, level = 0.95, ...)}

Arguments

- \texttt{object}: an object of class \texttt{fracdiff}, typically result of \texttt{fracdiff(..)}.
- \texttt{parm}: a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
- \texttt{level}: the confidence level required.
- \texttt{...}: additional argument(s) for methods.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2 in \% (by default 2.5\% and 97.5\%).

Author(s)

Spencer Graves posted the initial version to R-help.

See Also

the generic \texttt{confint}; \texttt{fracdiff} model fitting.

Examples

\begin{verbatim}
set.seed(101)
ts2 <- fracdiff.sim(5000, ar = .2, ma = -.4, d = .3)
mFD <- fracdiff( ts2$series, nar = length(ts2$ar), nma = length(ts2$ma))
coef(mFD)
confint(mFD)
\end{verbatim}
**diffseries**  

*Fractionally Differenciate Data*

**Description**

Differenciates the time series data using the approximated binomial expression of the long-memory filter and an estimate of the memory parameter in the ARFIMA(p,d,q) model.

**Usage**

```r
diffseries(x, d)
```

**Arguments**

- `x`: numeric vector or univariate time series.
- `d`: number specifying the fractional difference order.

**Value**

the fractionally differenced series `x`.

**Author(s)**

Valderio A. Reisen <valderio@cce.ufes.br> and Artur J. Lemonte

**References**

See those in `fdSperio`; additionally


**See Also**

`fracdiff.sim`

**Examples**

```r
memory.long <- fracdiff.sim(80, d = 0.3)
mGPH <- fdGPH(memory.long$series)
r <- diffseries(memory.long$series, d = mGPH$d)
#acf(r) # shouldn't show structure - ideally
```
Description

Estimate the fractional (or “memory”) parameter \( d \) in the ARFIMA(p,d,q) model by the method of Geweke and Porter-Hudak (GPH). The GPH estimator is based on the regression equation using the periodogram function as an estimate of the spectral density.

Usage

\[
\text{fdGPH}(x, \text{bandw.exp} = 0.5)
\]

Arguments

- \( x \) univariate time series
- \( \text{bandw.exp} \) the bandwidth used in the regression equation

Details

The function also provides the asymptotic standard deviation and the standard error deviation of the fractional estimator.

The bandwidth is \( bw = \text{trunc}(n^{\text{bandw.exp}}) \), where \( 0 < \text{bandw.exp} < 1 \) and \( n \) is the sample size. Default \( \text{bandw.exp} = 0.5 \).

Value

- \( d \) GPH estimate
- \( \text{sd.as} \) asymptotic standard deviation
- \( \text{sd.reg} \) standard error deviation

Author(s)

Valderio A. Reisen and Artur J. Lemonte

References

see those in \textit{fdsperio}.

See Also

\textit{fdsperio, fracdiff}

Examples

\[
\text{memory.long} \leftarrow \text{fracdiff.sim}(1500, d = 0.3)
\]
\[
\text{fdGPH(memory.long$series)}
\]
**Description**

This function makes use Reisen (1994) estimator to estimate the memory parameter \( d \) in the ARFIMA(\( p,d,q \)) model. It is based on the regression equation using the smoothed periodogram function as an estimate of the spectral density.

**Usage**

\[
\text{fdSperio}(x, \text{bandw.exp} = 0.5, \beta = 0.9)
\]

**Arguments**

- **x**: univariate time series data.
- **bandw.exp**: numeric: exponent of the bandwidth used in the regression equation.
- **beta**: numeric: exponent of the bandwidth used in the lag Parzen window.

**Details**

The function also provides the asymptotic standard deviation and the standard error deviation of the fractional estimator.

The bandwidths are \( bw = \text{trunc}(n^{\text{bandw.exp}}) \), where \( 0 < \text{bandw.exp} < 1 \) and \( n \) is the sample size. Default \( \text{bandw.exp} = 0.5 \);
and \( bw2 = \text{trunc}(n^\beta) \), where \( 0 < \beta < 1 \) and \( n \) is the sample size. Default \( \beta = 0.9 \).

**Value**

A list with components

- **d**: Sperio estimate
- **sd.as**: asymptotic standard deviation
- **sd.reg**: standard error deviation

**Author(s)**

Valderio A. Reisen <valderio@cc.e.ufes.br> and Artur J. Lemonte

**References**


fracdiff

ML Estimates for Fractionally-Differenced ARIMA (p,d,q) models

Description

Calculates the maximum likelihood estimators of the parameters of a fractionally-differenced ARIMA (p,d,q) model, together (if possible) with their estimated covariance and correlation matrices and standard errors, as well as the value of the maximized likelihood. The likelihood is approximated using the fast and accurate method of Haslett and Raftery (1989).

Usage

fracdiff(x, nar = 0, nma = 0, ar = rep(NA, max(nar, 1)), ma = rep(NA, max(nma, 1)), dtol = NULL, drange = c(0, 0.5), h, M = 100, trace = 0)

Arguments

x
number of autoregressive parameters p.

nar
number of moving average parameters q.

nma
initial autoregressive parameters.

ar
initial moving average parameters.

ma
interval of uncertainty for d. If dtol is negative or NULL, the fourth root of machine precision will be used. dtol will be altered if necessary by the program.

drange
interval over which the likelihood function is to be maximized as a function of d.

h
size of finite difference interval for numerical derivatives. By default (or if negative),

\[ h = \min(0.1, \epsilon_5 \times (1 + \text{abs}(\text{cllf}))), \]

where \( \text{cllf} := \log \text{max likelihood (as returned)} \) and \( \epsilon_5 := \text{sqrt(Machine\$double.neg.\epsilon)} \) (typically 1.05e-8).

This is used to compute a finite difference approximation to the Hessian, and hence only influences the cov, cor, and std.error computations; see also fracdiff.var.

M
number of terms in the likelihood approximation (see Haslett and Raftery 1989).

trace
optional integer, specifying a trace level. If positive, currently the “outer loop” iterations produce one line of diagnostic output.

See Also

fdGPH, fracdiff

Examples

memory.long <- fracdiff.sim(1500, d = 0.3)
spm <- fdSperio(memory.long$series)
str(spm, digits=6)
**Details**

The `fracdiff` package has — for historical reason, namely, S-plus `arima()` compatibility — used an unusual parametrization for the MA part, see also the ‘Details’ section in `arima` (in standard R’s `stats` package). The ARMA (i.e., \( d = 0 \)) model in `fracdiff()` and `fracdiff.sim()` is

\[
X_t - a_1 X_{t-1} - \cdots - a_p X_{t-p} = e_t - b_1 e_{t-1} - \ldots - b_q e_{t-q},
\]

where \( e_t \) are mean zero i.i.d., for `fracdiffHI`’s estimation, \( e_t \sim \mathcal{N}(0, \sigma^2) \). This model indeed has the signs of the MA coefficients \( b_j \) inverted, compared to other parametrizations, including Wikipedia’s [http://en.wikipedia.org/wiki/Autoregressive-moving-average_model](http://en.wikipedia.org/wiki/Autoregressive-moving-average_model) and the one of `arima`.

Note that NA’s in the initial values for ar or ma are replaced by 0’s.

**Value**

an object of S3 class “fracdiff”, which is a list with components:

- `log.likelihood`: logarithm of the maximum likelihood
- `d`: optimal fractional-differencing parameter
- `ar`: vector of optimal autoregressive parameters
- `ma`: vector of optimal moving average parameters
- `covariance.dpq`: covariance matrix of the parameter estimates (order : d, ar, ma).
- `stderr.dpq`: standard errors of the parameter estimates c(d, ar, ma).
- `correlation.dpq`: correlation matrix of the parameter estimates (order : d, ar, ma).
- `h`: interval used for numerical derivatives, see h argument.
- `dtol`: interval of uncertainty for d; possibly altered from input dtol.
- `M`: as input.
- `hessian.dpq`: the approximate Hessian matrix \( H \) of 2nd order partial derivatives of the likelihood with respect to the parameters; this is (internally) used to compute `covariance.dpq`, the approximate asymptotic covariance matrix as \( C = (-H)^{-1} \).

**Method**

The optimization is carried out in two levels:
an outer univariate unimodal optimization in d over the interval `drange` (typically \([0,.5]\)), using Brent’s `fmin` algorithm, and
an inner nonlinear least-squares optimization in the AR and MA parameters to minimize white noise variance (uses the MINPACK subroutine `lmDER`), written by Chris Fraley (March 1991).

**Note**

Ordinarily, `nar` and `nma` should not be too large (say < 10) to avoid degeneracy in the model. The function `fracdiff.sim` is available for generating test problems.
References


See Also

coeff.fracdiff and other methods for “fracdiff” objects; fracdiff.sim

Examples

ts.test <- fracdiff.sim( 5000, ar = .2, ma = -.4, d = .3)
fds <- fracdiff( ts.test$series,  
nar = length(ts.test$ar), nma = length(ts.test$ma))

## Confidence intervals

confint(fds)

## with iteration output

fd2 <- fracdiff(ts.test$series, nar = 1, nma = 1, trace = 1)
all.equal(fd, fd2)

fracdiff-methods  Many Methods for “fracdiff” Objects

Description

Many “accessor” methods for fracdiff objects, notably summary, coef, vcov, and logLik; further print() methods were needed.

Usage

## S3 method for class 'fracdiff'

coeff(object, ...)

## S3 method for class 'fracdiff'

logLik(object, ...)

## S3 method for class 'fracdiff'

print(x, digits = getOption("digits"), ...)

## S3 method for class 'fracdiff'

summary(object, symbolic.cor = FALSE, ...)

## S3 method for class 'summary.fracdiff'

print(x, digits = max(3, getOption("digits") - 3),  
correlation = FALSE, symbolic.cor = x$symbolic.cor,  
signif.stars = getOption("show.signif.stars"), ...)
## S3 method for class 'fracdiff'
vcov(object, ...)

### Arguments

- `x`, `object`: object of class `fracdiff`.
- `digits`: the number of significant digits to use when printing.
- `...`: further arguments passed from and to methods.
- `correlation`: logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
- `symbolic.cor`: logical. If TRUE, print the correlations in a symbolic form (see `symnum`) rather than as numbers.
- `signif.stars`: logical. If TRUE, “significance stars” are printed for each coefficient.

### Author(s)

Martin Maechler

### See Also

`fracdiff` to get “fracdiff” objects, `confint.fracdiff` for the `confint` method; further, `fracdiff.var`.

### Examples

```r
set.seed(7)
ts4 <- fracdiff.sim(10000, ar = c(0.6, -0.05, -0.2), ma = -0.4, d = 0.2)
modFD <- fracdiff(ts4$series, nar = length(ts4$ar), nma = length(ts4$ma))
## -> warning (singular Hessian) % FIXME ???
coef(modFD) # the estimated parameters
vcov(modFD)
smFD <- summary(modFD)
smFD
coef(smFD) # gives the whole table
AIC(modFD) # AIC works because of the logLik() method
```

---

**fracdiff.sim**  
Simulate fractional ARIMA Time Series

### Description

Generates simulated long-memory time series data from the fractional ARIMA(p,d,q) model. This is a test problem generator for `fracdiff`.

Note that the MA coefficients have inverted signs compared to other parametrizations, see the details in `fracdiff`.
Usage

\texttt{fracdiff.sim(n, ar = NULL, ma = NULL, d,}
\quad \texttt{rand.gen = rnorm, innov = rand.gen(n+q, ...),}
\quad \texttt{n.start = NA, backComp = TRUE, allow.0.nstart = FALSE,}
\quad \texttt{start.innov = rand.gen(n.start, ...),}
\quad \ldots, \texttt{mu = 0)}

Arguments

- \texttt{n} length of the time series.
- \texttt{ar} vector of autoregressive parameters; empty by default.
- \texttt{ma} vector of moving average parameters; empty by default.
- \texttt{d} fractional differencing parameter.
- \texttt{rand.gen} a function to generate the innovations; the default, \texttt{rnorm} generates white \texttt{N(0,1)} noise.
- \texttt{innov} an optional times series of innovations. If not provided, \texttt{rand.gen()} is used.
- \texttt{n.start} length of “burn-in” period. If \texttt{NA}, the default, the same value as in \texttt{arima.sim} is computed.
- \texttt{backComp} logical indicating if back compatibility with older versions of \texttt{fracdiff.sim} is desired. Otherwise, for \texttt{d = 0}, compatibility with \texttt{R}'s \texttt{arima.sim} is achieved.
- \texttt{allow.0.nstart} logical indicating if \texttt{n.start = 0} should be allowed even when \( p + q > 0 \). This not recommended unless for producing the same series as with older versions of \texttt{fracdiff.sim}.
- \texttt{start.innov} an optional vector of innovations to be used for the burn-in period. If supplied there must be at least \texttt{n.start} values.
- \texttt{...} additional arguments for \texttt{rand.gen()}. Most usefully, the standard deviation of the innovations generated by \texttt{rnorm} can be specified by \texttt{sd}.
- \texttt{mu} time series mean (added at the end).

Value

- a list containing the following elements:
  - \texttt{series} time series
  - \texttt{ar}, \texttt{ma}, \texttt{d}, \texttt{mu}, \texttt{n.start}
    same as input

See Also

\texttt{fracdiff}, also for references; \texttt{arima.sim}
Examples

```r
## Pretty (too) short to "see" the long memory
fracdiff.sim(100, ar = .2, ma = .4, d = .3)

## longer with "extreme" ar:
 r <- fracdiff.sim(n=1500, ar=-0.9, d= 0.3)
plot(as.ts(r$series))

## Show that MA coefficients meaning is inverted
## compared to stats :: arima :

AR <- 0.7
MA <- -0.5
n.st <- 2

AR <- c(0.7, -0.1)
MA <- c(-0.5, 0.4)
n <- 512 ; sd <- 0.1
n.st <- 10

set.seed(101)
Y1 <- arima.sim(list(ar = AR, ma = MA), n = n, n.start = n.st, sd = sd)
plot(Y1)

# For our fracdiff, reverse the MA sign:
set.seed(101)
Y2 <- fracdiff.sim(n = n, ar = AR, ma = - MA, d = 0,
    n.start = n.st, sd = sd)$series
lines(Y2, col=adjustcolor("red", 0.5))

## .. no, you don't need glasses ;-) Y2 is Y1 shifted slightly

##' rotate left by k (k < 0: rotate right)
rot <- function(x, k) {
  stopifnot(k == round(k))
  n <- length(x)
  i <- (n-k+1):n
  x[c(i, (1:k))]
}
k <- n.st - 2
Y2.s <- rot(Y2, k)
head.matrix(cbind(Y1, Y2.s))
plot(Y1, Y2.s); i <- (n-k+1):n
text(Y1[i], Y2.s[i], adj = c(0,0) -- 1, col=2)

## With backComp = FALSE, get *the same* as arima.sim():
set.seed(101)
Y2. <- fracdiff.sim(n = n, ar = AR, ma = - MA, d = 0,
    n.start = n.st, sd = sd, backComp = FALSE)$series
stopifnot( all.equal( c(Y1), Y2., tol= 1e-15))
```

Recompute Covariance Estimate for fracdiff

---

fracdiff.var

Recompute Covariance Estimate for fracdiff
Description

Allows the finite-difference interval to be altered for recomputation of the covariance estimate for `fracdiff`.

Usage

`fracdiff.var(x, fracdiff.out, h)`

Arguments

- `x`: a univariate time series or a vector. Missing values (NAs) are not allowed.
- `fracdiff.out`: output from `fracdiff` for time series `x`.
- `h`: finite-difference interval for approximating partial derivatives with respect to the `d` parameter.

Value

an object of S3 class "fracdiff", i.e., basically a list with the same elements as the result from `fracdiff`, but with possibly different values for the hessian, covariance, and correlation matrices and for standard error, as well as for `h`.

See Also

`fracdiff`, also for references.

Examples

```r
## Generate a fractionally-differenced ARIMA(1,d,1) model:
> ts.test <- fracdiff.sim(10000, ar = .2, ma = .4, d = .3)
## estimate the parameters in an ARIMA(1,d,1) model for the simulated series
> fd.out <- fracdiff(ts.test$ser, nar = 1, nma = 1)

## Modify the covariance estimate by changing the finite-difference interval
> (fd.o2 <- fracdiff.var(ts.test$series, fd.out, h = .0001))
## looks identical as print(fd.out),
## however these (e.g.) differ:
> vcov(fd.out)
> vcov(fd.o2)
```
Index

*Topic models
  confint.fracdiff, 2
  fracdiff-methods, 8
*Topic print
  fracdiff-methods, 8
*Topic ts
  diffseries, 3
  fdGPH, 4
  fdsPerio, 5
  fracdiff, 6
  fracdiff.sim, 9
  fracdiff.var, 11

arima, 7
arima.sim, 10

class, 7, 12
coeff, 8
coeff.fracdiff, 8
coeff.fracdiff(fracdiff-methods), 8
confint, 2, 9
confint.fracdiff, 2, 9
diffseries, 3

fdGPH, 4, 6
fdsPerio, 3, 4, 5
fitted.fracdiff(fracdiff-methods), 8
fracdiff, 2, 4, 6, 6-10, 12
fracdiff-methods, 8
fracdiff.sim, 3, 7, 8, 9
fracdiff.var, 6, 9, 11

logLik, 8
logLik.fracdiff(fracdiff-methods), 8

print, 8
print.fracdiff(fracdiff-methods), 8
print.summary.fracdiff
  (fracdiff-methods), 8

residuals.fracdiff(fracdiff-methods), 8
rnorm, 10

summary, 8
summary.fracdiff(fracdiff-methods), 8
symnum, 9
vcov, 8
vcov.fracdiff(fracdiff-methods), 8