

# Package ‘FGN’

February 14, 2012

**Version** 1.5

**Date** 2011-02-11

**Title** Fractional Gaussian Noise, estimation and simulaton

**Author** A.I. McLeod

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

**Depends** R (>= 2.1.0), ltsa

**Description** MLE for H parameter in FGN; MLE for regression with FGN error; simulation of FGN; print, summary, plot, coef,residuals, Boot methods.

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

**Classification/MSC** 62M10, 91B84

**Imports** ltsa

**LazyLoad** yes

**LazyData** yes

**License** GPL (>= 2)

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

**Repository** CRAN

**Date/Publication** 2011-02-12 13:30:09

## R topics documented:

FGN-package . . . . .	2
Boot . . . . .	3
Boot.FitFGN . . . . .	4
coef.FitFGN . . . . .	5
FGNAcf . . . . .	6
FGNLL . . . . .	7

FitFGN . . . . .	8
FitRegressionFGN . . . . .	10
GetFitFGN . . . . .	11
HurstK . . . . .	13
NileFlowCMS . . . . .	14
NileMin . . . . .	14
plot.FitFGN . . . . .	16
predict.FitFGN . . . . .	17
print.FitFGN . . . . .	18
residuals.FitFGN . . . . .	19
SimulateFGN . . . . .	20
summary.FitFGN . . . . .	21

<b>Index</b>	<b>23</b>
--------------	-----------

---

FGN-package	<i>Fractional Gaussian Noise, estimation and simulation</i>
-------------	---

---

## Description

FGN (Fractional Gaussian Noise) model fitting, simulation, bootstrapping, forecasting. Minimum Nile River flow, annual, 663 values, 622 AD to 1284 AD. Annual Nile river flow at Aswan, 1871-1945 from original source.

## Details

Package:	FGN
Type:	Package
Version:	1.5
Date:	2011-02-11
License:	GPL (>= 2)
LazyLoad:	yes
LazyData:	yes

This package provides a comprehensive approach to fitting FGN.

## Author(s)

A. I. McLeod, Hao Yu and Zinovi Krougly.

Maintainer: aimcleod@uwo.ca

## References

Hipel, K.W. and McLeod, A.I., (2005). Time Series Modelling of Water Resources and Environmental Systems. Electronic reprint of our book originally published in 1994. <http://www.stats.uwo.ca/faculty/aim/1994Book/>.

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

### See Also

[HurstK](#), [FitFGN](#), [FitRegressionFGN](#), [SimulateFGN](#), [print.FitFGN](#), [summary.FitFGN](#), [predict.FitFGN](#), [plot.FitFGN](#), [residuals.FitFGN](#)

### Examples

```
#Compare HurstK and MLE for H
#Hurst K for Nile Minima
data(NileMin)
HurstK(NileMin)
out<-FitFGN(NileMin)
summary(out)
plot(out)
coef(out)
```

---

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

### Value

Parametric bootstrap simulation

### Author(s)

A.I. McLeod

### References

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**[Boot.FitFGN](#)**Examples**

```
data(NileMin)
out<-FitFGN(NileMin)
Boot(out, R=3)
```

---

`Boot.FitFGN`*Simulate Fitted FGN Model*

---

**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 'FitFGN'
Boot(obj, R = 1, ...)
```

**Arguments**

<code>obj</code>	the output from FitAR
<code>R</code>	number of bootstrap replications
<code>...</code>	optional arguments

**Details**

The method of Davies and Harte (1987) is used if it is applicable, otherwise the Durbin-Levinson recursion is used.

**Value**

If  $R=1$ , a simulated time series with the same length as the original fitted time series is produced. Otherwise if  $R>1$ , a matrix with  $R$  columns and number of rows equal to the length of the series containing  $R$  replications of the bootstrap.

**Author(s)**

A.I. McLeod

**References**

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**

[SimulateFGN](#), [DHSimulate](#) [DLSimulate](#)

**Examples**

```
#Example 1
#Fit a FGN model and determine the bootstrap sd of H
#Measure cpu time. With R=250, it takes about 23 sec
#on 3.6 GHz Pentium IV.
data(NileMin)
outNileMin<-FitFGN(NileMin)
start<-proc.time()[1]
R<-25
Hs<-numeric(R)
Z<-Boot(outNileMin, R=R)
for (i in 1:R)
  Hs[i]<-GetFitFGN(Z[,i])$H
BootSD<-sd(Hs) #this is the bootstrap sd
end<-proc.time()[1]
totTim<-end-start
```

---

coef.FitFGN

*Display estimated parameters from FitFGN*


---

**Description**

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

**Usage**

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

**Arguments**

```
object      obj the output from FitFGN
...         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

**References**

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**Examples**

```
data(NileMin)
out<-FitFGN(NileMin)
coef(out)
```

---

FGNAcf

*Autocorrelation of FGN*

---

**Description**

The FGN time series is an example of a time series exhibiting long-range dependence and characterized by the fact that its autocorrelation function exhibits hyperbolic decay rather than exponential decay found in stationary ARMA time series. The FGN and other alternatives are discussed in Hipel and McLeod (2005).

**Usage**

```
FGNAcf(k, H)
```

**Arguments**

k	lag or lags - may be vector
H	Hurst parameter

**Value**

value of the autocorrelation at lag(s) k

**Note**

The parameter H should be in (0,1). An error message is given if it is not.

**Author(s)**

A.I. McLeod

**References**

Hipel, K.W. and McLeod, A.I., (2005). Time Series Modelling of Water Resources and Environmental Systems. Electronic reprint of our book originally published in 1994. <http://www.stats.uwo.ca/faculty/aim/1994Book/>.

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**[FGNLL](#), [acf](#)**Examples**

```
#compute the acf at lags 0,1,...,10 when H=0.7
FGNAcf(0:10, 0.7)
```

---

**FGNLL***Concentrated Loglikelihood Function for H*

---

**Description**

The concentrated loglikelihood, that is, the loglikelihood function maximized over the innovation variance parameter, is computed.

**Usage**

```
FGNLL(H, z)
```

**Arguments**

H	parameter
z	data vector, assumed to be mean corrected

**Value**

the value of the loglikelihood

**Author(s)**

A.I. McLeod

**References**

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**[FitFGN](#), [DLLoglikelihood](#)

**Examples**

```
#compute loglikelihood for NileFlowCMS with H=0.9
data(NileFlowCMS)
z<-NileFlowCMS
z<-z-mean(z)
FGNLL(0.9, z)

#simulate Gaussian white noise and tabulate the loglikelihood for H=0.40, 0.45, 0.50, 0.55, 0.60
set.seed(4321)
h<-c(0.40, 0.45, 0.50, 0.55, 0.60)
z<-rnorm(500, 100, 50)
z<-z-mean(z)
LL<-numeric(length(h))
for (i in 1:length(h))
LL[i]<-FGNLL(h[i],z)
matrix(c(h,LL),ncol=2)
```

---

FitFGN

*MLE estimation for FGN*


---

**Description**

Exact MLE estimation for FGN

**Usage**

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

**Arguments**

<code>z</code>	time series, vector or ts object.
<code>demean</code>	if True, subtract mean. Otherwise assume it is zero.
<code>MeanMLEQ</code>	if True, an iterative algorithm is used for exact simultaneous MLE estimation of the mean and other parameters.
<code>lag.max</code>	the residual autocorrelations are tabulated for lags 1, ..., lag.max. Also lag.max is used for the Ljung-Box portmanteau test.

**Details**

The exact loglikelihood function is maximized numerically using `optimize`. The standard error for the H parameter is estimated (McLeod, Yu and Krougly, 2007).

**Value**

A list with class name "FitAR" and components:

loglikelihood	value of the loglikelihood
H	estimate of H parameter
SEH	SE of H estimate
sigsqHat	innovation variance estimate
muHat	estimate of the mean
SEmu	SE of mean
Rsqr	R-squared, coefficient of forecastability
LjungBox	table of Ljung-Box portmanteau test statistics
res	normalized residuals, same length as z
demean	TRUE if mean estimated otherwise assumed zero
IterationCount	number of iterations in mean mle estimation
MLEMeanQ	TRUE if mle for mean algorithm used
tsp	tsp(z)
call	result from match.call() showing how the function was called
DataTitle	returns attr(z,"title")

**Author(s)**

A.I. McLeod

**References**

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**

[GetFitFGN](#), [FitRegressionFGN](#), [Boot.FitFGN](#), [coef.FitFGN](#), [plot.FitFGN](#), [print.FitFGN](#), [summary.FitFGN](#), [HurstK](#)

**Examples**

```
data(NileMin)
out<-FitFGN(NileMin)
summary(out)
plot(out)
coef(out)
```

---

FitRegressionFGN      *Regression with FGN Errors*

---

**Description**

Fits a multiple linear regression with FGN errors

**Usage**

FitRegressionFGN(X, y)

**Arguments**

X	design matrix, must include column of 1's if constant term is present
y	the response variable, a time series

**Details**

An iterative algorithm is used to compute the exact MLE.

**Value**

a list with 3 elements:

Loglikelihood	value of the maximized loglikelihood
H	MLE for H
alpha	MLE for regression coefficients corresponding to columns of X

**Note**

It is assumed that X is not collinear.

**Author(s)**

A.I. McLeod

**References**

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**

[FitFGN](#), [lsfit](#)

**Examples**

```
#simulate FGN with mean zero and H=0.2 and fit exact mle for H and mean
H<-0.2
z<-SimulateFGN(512, H)
mean(z)
X<-matrix(rep(1,length(z)), ncol=1)
ans<-FitRegressionFGN(X,z)
ans
```

```
#fit a step intervention model to the Nile annual riverflow data
data(NileFlowCMS)
n<-length(NileFlowCMS)
X<-matrix(c(rep(1,n),rep(0,32),rep(1,n-32)),ncol=2)
ans<-FitRegressionFGN(X,NileFlowCMS)
ans
```

---

 GetFitFGN

*Fit FGN Time Series Model*


---

**Description**

Exact maximum likelihood estimation of the parameter  $H$  in fractional Gaussian noise (FGN). This is a utility function used by [FitFGN](#) but it is also useful in simulation experiments since it is faster than using [FitFGN](#). See example below.

**Usage**

```
GetFitFGN(z, MeanZeroQ = FALSE)
```

**Arguments**

<code>z</code>	time series data vector
<code>MeanZeroQ</code>	optional argument, default is <code>MeanZeroQ=FALSE</code> . Set to <code>TRUE</code> if the mean is known to be zero

**Details**

The function `optimize` is used. It is very rare but it has been observed that `optimize` can incorrectly choose an endpoint. If this happens a warning is given and `optim` is used.

**Value**

a list with two elements:

<code>Loglikelihood</code>	value of the maximized loglikelihood
<code>H</code>	MLE for $H$

**Author(s)**

A.I. McLeod

**References**

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**

[optimize](#), [optim](#), [Boot.FitFGN](#), [FitFGN](#), [FitRegressionFGN](#)

**Examples**

```
#Example 1
#fit Gaussian White Noise, H=0.5
z<-rnorm(500, 100, 10)
GetFitFGN(z)

#Example 2
#estimate H for NileMin series
data(NileMin)
GetFitFGN(NileMin)

#Example 3
#Timing comparison for GetFitFGN and FitFGN
ns<-c(500,1000) #may extend this to other n's
H<-0.8
nR<-10
tim1<-tim2<-numeric(length(ns))
for (i in 1:length(ns)){
  n <- ns[i]
  t1<-t2<-0
  s1<-proc.time()[1]
  for (iR in 1:nR){
    z<-SimulateFGN(n, H)
    H1<-GetFitFGN(z)
  }
  e1<-proc.time()[1]
  t1<-t1+(e1-s1)
  s2<-proc.time()[1]
  for (iR in 1:nR){
    z<-SimulateFGN(n, H)
    H2<-FitFGN(z)
  }
  e2<-proc.time()[1]
  t2<-t2+(e2-s2)
  tim1[i]<-t1
  tim2[i]<-t2
}
tb<-matrix(c(tim1,tim2),ncol=2)
dimnames(tb)<-list(ns,c("GetFitFGN","FitFGN"))
```

---

HurstK	<i>Hurst K Coefficient</i>
--------	----------------------------

---

**Description**

The Hurst K provides a non-parametric estimate for the Hurst H coefficient

**Usage**

```
HurstK(z)
```

**Arguments**

z                    time series vector

**Details**

There are many alternative non-parametric estimators for H. Some of the popular ones are discussed in Hipel and McLeod (2005).

**Value**

an estimate of H

**Author(s)**

A.I. McLeod

**References**

Hipel, K.W. and McLeod, A.I., (2005). Time Series Modelling of Water Resources and Environmental Systems. Electronic reprint of our book originally published in 1994. <http://www.stats.uwo.ca/faculty/aim/1994Book/>.

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**

[FitFGN](#)

**Examples**

```
# the Hurst coefficient for NID series is 0.5
z<-rnorm(1000)
HurstK(z)
#Hurst K for Nile Minima
data(NileMin)
HurstK(NileMin)
```

---

 NileFlowCMS

*Annual flow of Nile River at Aswan, 1871-1945*


---

**Description**

This is average annual flow of the Nile River below the Aswan Dam. The units are CMS (cubic meters per second).

**Usage**

```
data(NileFlowCMS)
```

**Format**

The format is: Time-Series [1:75] from 1870 to 1944: 3958 3370 3485 3438 3702 ...

**Source**

Hipel, K.W. and McLeod, A.I., (2005). Time Series Modelling of Water Resources and Environmental Systems. Electronic reprint of our book originally published in 1994. <http://www.stats.uwo.ca/faculty/aim/1994Book/>.

**Examples**

```
#Plot the time series
data(NileFlowCMS)
ts.plot(NileFlowCMS)

#Hurst K estimate
HurstK(NileFlowCMS)
```

---

 NileMin

*Nile Annual Minima, 622 AD to 1284 AD*


---

**Description**

Annual Minimum flow of Nile River. See below for details.

**Usage**

```
data(NileMin)
```

**Format**

The format is: Time-Series [1:663] from 622 to 1284: 11.57 10.88 11.69 11.69 9.84 ... - attr(\*, "title")= "Nile River minima series"

## Details

The minimum annual level of the Nile has been recorded over many centuries and was given by Toussoun (1925). The data over the period 622 AD to 1284 AD is considered more homogenous and reliable than the full dataset and has been analyzed by Beran (1994) and Percival and Walden (2000). The full dataset is available StatLib Datasets hipel-mcleod archive – file: Minimum.

## Source

Toussoun, O. (1925). Memoire sur l’Histoire du Nil. In Memoires a l’Institut d’Egypte, 18, 366-404.

## References

Beran, J. (1994). Statistics for Long-Memory Processes. Chapman and Hall, New York.

Percival, D.B. and Walden, A.T. (2000) Wavelet Methods for Time Series Analysis. Cambridge University Press.

## Examples

```
#Compute Hurst's K estimate of H
data(NileMin)
HurstK(NileMin)

#Script for comparing FGN/ARMA forecast performance
#
data(NileMin)
outNileMin<-FitFGN(NileMin)
set.seed(12177)
z<-Boot(outNileMin)
n<-length(z)
K<-100 #number of out-of-sample data values
z1<-z[1:(length(z)-K)] #training data
z2<-z[-(1:(length(z)-K))] #testing data
#
#FGN fit to z1 and forecast using z2
maxLead<-3
n1<-length(z1)
outz1<-FitFGN(z1)
H<-outz1$H
mu<-outz1$muHat
rFGN<-var(z1)*FGNAcf(0:(n + maxLead - 1), H)
F<-TrenchForecast(c(z1,z2), rFGN, mu, n1, maxLead=maxLead)$Forecasts
nF<-nrow(F)
err1<-z2-F[,1][-nF]
err2<-z2[-1]-F[,2][-c(nF,(nF-1))]
err3<-z2[-c(1,2)]-F[,3][-c(nF,(nF-1),(nF-2))]
rmse1<-sqrt(mean(err1^2))
rmse2<-sqrt(mean(err2^2))
rmse3<-sqrt(mean(err3^2))
FGNrmse<-c(rmse1,rmse2,rmse3)
#
```

```

#ARMA(p,q) fit to z1 and forecast using z2
p<-2
q<-1
ansz1<-arima(z1, c(p,0,q))
phi<-theta<-numeric(0)
if (p>0) phi<-coef(ansz1)[1:p]
if (q>0) theta<-coef(ansz1)[(p+1):(p+q)]
zm<-coef(ansz1)[p+q+1]
sigma2<-ansz1$sigma2
vz<-tacvfARMA(phi=phi, theta=theta, sigma2=sigma2, maxLag=0)
r<-vz*ARMAacf(ar=phi, ma=theta, lag.max=n + maxLead -1)
F<-TrenchForecast(c(z1,z2), r, zm, n1, maxLead=3)$Forecasts
err1<-z2-F[,1][-nF]
err2<-z2[-1]-F[,2][-c(nF,(nF-1))]
err3<-z2[-c(1,2)]-F[,3][-c(nF,(nF-1),(nF-2))]
rmse1<-sqrt(mean(err1^2))
rmse2<-sqrt(mean(err2^2))
rmse3<-sqrt(mean(err3^2))
ARMARMse<-c(rmse1,rmse2,rmse3)
#
#tabulate result
tb<-matrix(c(FGNrmse,ARMARMse),ncol=2)
dimnames(tb)<-list(c("lead1","lead2","lead3"),c("FGN","ARMA"))

```

---

plot.FitFGN

*Plot Method for "FitFGN" Object*


---

## Description

Diagnostic plots of the residual autocorrelations and Ljung-Box test.

## Usage

```

## S3 method for class 'FitFGN'
plot(x, maxLag=30, ...)

```

## Arguments

x	object of class "FitFGN"
maxLag	maximum lag in residual acf plot
...	optional arguments

## Details

The top plot shows the residual autocorrelations and their 5% significance limits. The bottom plot shows the p-values of the Ljung-Box test for various lags.

**Value**

No value is returned. A plots are produced as side-effect. The plot is a two-panel display showing the residual autocorrelations and the p-values for the Ljung-Box test.

**Author(s)**

A.I. McLeod

**References**

Ljung, G.M., Box, G.E.P. (1978). On a Measure of Lack of Fit in Time Series Models. *Biometrika* 65, 297-303.

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, *Journal of Statistical Software*.

**See Also**

[summary.FitFGN](#), [FitFGN](#)

**Examples**

```
data(NileMin)
obj<-FitFGN(NileMin, c(1,2,6,7))
plot(obj)
```

---

predict.FitFGN

*Forecasts from a fitted FGN model*

---

**Description**

The exact finite-sample minimum mean square error forecasts are computed using the Trench algorithm.

**Usage**

```
## S3 method for class 'FitFGN'
predict(object, n.ahead = 1, ...)
```

**Arguments**

object	"FitFGN" object produced by FitFGN
n.ahead	forecasts are done for lead times 1,...,n.ahead
...	optional arguments, are ignored

**Value**

A list with components

Forecasts        matrix with m+1 rows and maxLead columns with the forecasts

SDForecasts     matrix with m+1 rows and maxLead columns with the sd of the forecasts

**Author(s)**

A.I. McLeod

**References**

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**

[FitFGN](#), [TrenchForecast](#) [PredictionVariance](#) [predict.Arima](#)

**Examples**

```
data(NileMin)
out<-FitFGN(NileMin)
predict(out, n.ahead=15)
```

---

`print.FitFGN`

*Print Method for "FitFGN" Object*

---

**Description**

A terse summary is given.

**Usage**

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

**Arguments**

x                object of class "FitFGN"  
 ...              optional arguments

**Value**

A terse summary is displayed

**Author(s)**

A.I. McLeod

**References**

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, Journal of Statistical Software.

**See Also**[summary.FitFGN](#), [FitFGN](#)**Examples**

```
data(NileMin)
FitFGN(NileMin)
```

---

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

---

**Description**

Method function.

**Usage**

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

**Arguments**

object	object of class "FitFGN"
...	optional arguments

**Value**

Vector of standardized prediction residuals

**Author(s)**

A.I. McLeod

**See Also**[FitFGN](#)

**Examples**

```
data(NileMin)
out<-FitFGN(NileMin)
qqnorm(resid(out))
```

---

SimulateFGN

*Simulates FGN*

---

**Description**

A fractional Gaussian noise time series is simulated.

**Usage**

```
SimulateFGN(n, H)
```

**Arguments**

n	length of time series
H	Hurst coefficient

**Details**

The FFT is used so it is most efficient if you select n to be a power of 2.

**Value**

vector of length containing the simulated time series

**Author(s)**

A.I. McLeod

**References**

Davies, R. B. and Harte, D. S. (1987). Tests for Hurst Effect. *Biometrika* 74, 95–101.

McLeod, A.I., Yu, Hao, Krougly, Zinovi L. (2007). Algorithms for Linear Time Series Analysis, *Journal of Statistical Software*.

**See Also**

[DLSimulate](#)

**Examples**

```

#Example 1
#simulate a process with H=0.2 and plot it
z<-SimulateFGN(100, 0.2)
ts.plot(z)
#
#Example 2
#simulate FGN and compare theoretical and sample autocovariances
H<-0.7
n<-8192
z<-SimulateFGN(n, H)
#autocovariances
sacvf<-acf(z, plot=FALSE,type="covariance")$acf
tacf<-FGNAcf(0:(n-1), H)
tb<-matrix(c(tacf[1:10],sacvf[1:10]),ncol=2)
dimnames(tb)<-list(0:9, c("Tacf","Sacvf"))
tb

```

summary.FitFGN

*Summary Method for "FitFGN" Object***Description**

summary for "FitFGN" object.

**Usage**

```

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

```

**Arguments**

object	"FitFGN" object
...	optional arguments

**Value**

A printed summary is given

**Author(s)**

A.I. McLeod

**See Also**

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

**Examples**

```
data(NileMin)
out<-FitFGN(NileMin)
summary(out)
```

# Index

- \*Topic **datagen**
    - SimulateFGN, 20
  - \*Topic **datasets**
    - NileFlowCMS, 14
    - NileMin, 14
  - \*Topic **nonlinear**
    - FitRegressionFGN, 10
  - \*Topic **nonparametric**
    - HurstK, 13
  - \*Topic **package**
    - FGN-package, 2
  - \*Topic **regression**
    - FitRegressionFGN, 10
  - \*Topic **ts**
    - Boot, 3
    - Boot.FitFGN, 4
    - coef.FitFGN, 5
    - FGN-package, 2
    - FGNAcf, 6
    - FGNLL, 7
    - FitFGN, 8
    - FitRegressionFGN, 10
    - GetFitFGN, 11
    - HurstK, 13
    - plot.FitFGN, 16
    - predict.FitFGN, 17
    - print.FitFGN, 18
    - residuals.FitFGN, 19
    - SimulateFGN, 20
    - summary.FitFGN, 21
- acf, 7
- Boot, 3
- Boot.FitFGN, 4, 4, 9, 12
- coef.FitFGN, 5, 9
- DHSimulate, 5
- DLLLoglikelihood, 7
- DLSimulate, 5, 20
- FGN-package, 2
- FGNAcf, 6
- FGNLL, 7, 7
- FitFGN, 3, 7, 8, 10–13, 17–19, 21
- FitRegressionFGN, 3, 9, 10, 12
- GetFitFGN, 9, 11
- HurstK, 3, 9, 13
- lsfit, 10
- NileFlowCMS, 14
- NileMin, 14
- optim, 12
- optimize, 12
- plot.FitFGN, 3, 9, 16
- predict.Arima, 18
- predict.FitFGN, 3, 17
- PredictionVariance, 18
- print.FitFGN, 3, 9, 18, 21
- residuals.FitFGN, 3, 19
- SimulateFGN, 3, 5, 20
- summary.FitFGN, 3, 9, 17, 19, 21
- TrenchForecast, 18