Package ‘RGENERATE’

February 11, 2017

Maintainer Emanuele Cordano <emanuele.cordano@gmail.com>
License GPL (>= 2)
Title Tools to Generate Vector Time Series
Type Package
Author Emanuele Cordano
Description A method 'generate()' is implemented in this package for the random
generation of vector time series according to models obtained by 'RMAWGEN',
'vars' or other packages. This package was created to generalize the
algorithms of the 'RMAWGEN' package for the analysis and generation of any
environmental vector time series.
Repository CRAN
URL https://github.com/ecor/RGENERATE
Date 2017-02-06
Version 1.3.5
Depends R (>= 2.10), RMAWGEN
RoxygenNote 5.0.1
NeedsCompilation no
Date/Publication 2017-02-11 15:17:56

R topics documented:

  gapFilling ........................................... 2
generate .............................................. 3

Index

  7
gapFilling

Description

gapFilling
It fills in a gap of a data frame by using `generate` method

Usage

gapFilling(x = NULL, ...)

## Default S3 method:
gapFilling(x, objectForGeneration = NULL, ...)

## S3 method for class 'data.frame'
gapFilling(x, objectForGeneration = NULL,
max.filling = 2, nofill.code = -9999, ...)

Arguments

x object with gaps to fill
...
  further argument for `generate` method
objectForGeneration object used for `generate` method
max.filling integer values: gap are filled if the previous max.filling values are not NA or nofill.code
nofill.code Alternative value to NA which indicates the gaps which are not filled

Examples

```
set.seed(122)
NSTEP <- 1000
x <- rnorm(NSTEP)
y <- x+rnorm(NSTEP)
z <- c(rnorm(1),y[-1]+rnorm(NSTEP-1))
df <- data.frame(x=x,y=y,z=z)
var <- VAR(df,type="none")

dfobs <- df
dfobs[20:30,2] <- NA
n <- nrow(df)
gp <- gapFilling(x=dfobs,objectForGeneration=var,max.filling=2)
```
### Description

It generates a multivariate random series according to the model \( x \).

### Usage

```r
generate(x = NULL, ...)
```

#### Default S3 method:

```r
generate(x, FUN = rnorm, n = 100, K = 3, names = NULL,
          cov = NULL, gap.filling = NULL, ...)
```

#### S3 method for class 'varest'

```r
generate(x, FUN = rnorm, n = 100, names = NULL,
          noise = NULL, exogen = NULL, xprev = NULL, gap.filling = NULL, ...)
```

#### S3 method for class 'varest2'

```r
generate(x, FUN = rnorm, n = 100, names = NULL,
          noise = NULL, exogen = NULL, xprev = NULL, extremes = TRUE,
          type = 3, gap.filling = NULL, GPCA.row.gap.filling.option = TRUE, ...)
```

#### S3 method for class 'matrix'

```r
generate(x, FUN = rnorm, n = 100, noise = NULL,
          xprev = NULL, names = NULL, gap.filling = NULL,
          type = c("autoregression", "covariance"), ...)
```

#### S3 method for class 'list'

```r
generate(x, factor.series = names(x), n = NA, ...)
```

#### S3 method for class 'MonthlyList'

```r
generate(x, origin, n, ...)
```

### Arguments

- **x**  
  null object or the model used for random generation, e.g., a VAR model as a \texttt{varest-class} or \texttt{varest2-class} object. Default is NULL.
... further arguments for FUN

FUN random function of the probability distribution used for noise random generation. Default is rnorm. See https://CRAN.R-project.org/view=Distributions

n number of generations requested

K number of the variables to be generated simultaneously, i.e. the K parameters of a VAR. It is automatically detected by x, names or cov, if one of these is not NULL.

names null object or string vectors or names of the variables to be generated simultaneously. Default is NULL.

cov null object or covariance matrix of the random variables to be generated simultaneously. Default is NULL, not used in case this information can be detected from x.

gap.filling data frame with time series with gabs (NA values) to be filled. Default is NULL and not considered, otherwise the method returns this data frame with NA row replaced with generated (e.g auto-regressed) values.

noise null object or a generic external noise for x model residuals, e.g. standard white noise, for random generation with the model x. Default is NULL. If NULL the noise is automatically calculated.

exogen null object or amatrix or data frame with exogeneous variables (predictors) id requested by x. Default is NULL.

xprev null object or initial condition of the multivariate random process to be generated. Default is NULL.

extremes see inv_GPCA

type character string used in some method implementations. See inv_GPCA. In the matrix implementation, default is "autoregression", i.e. the matrix is used as a vector auto-regression coefficient, if it is "covariance" the method generated a sample with covariance matrix given by x.

GPCA.row.gap.filling.option logical value. Default is TRUE. In case of GPCArest2-class objects, If gap.filling contains both NA and finite values in the same row, this row will contains all NA values after GPCA. In this case all row values are generated through auto-regression. If GPCA.row.gap.filling.option all insterred non-NA gap.filling values are replaced before returning the function value. Otherwise, in the rows with NAs all values are re-generated. The option TRUE is not safe in case the gaps are vary long becouse the generated values is used for subsequent auto-regression.

factor.series factor series used by 'factor.series'

origin start date for generation. See adddate

Value

a matrix or a data frame object

See Also
getVARmodel
library(rgenerate)

set.seed(122)
NSTEP <- 1000
x <- rnorm(NSTEP)
y <- x+rnorm(NSTEP)
z <- c(rnorm(1), y[-1]+rnorm(NSTEP-1))
df <- data.frame(x=x, y=y, z=z)
var <- VAR(df, type="none")
gg <- generate(var, n=20)

cov <- cov(gg)

### Generate an auto-regressive time-series with a generic matrix

A <- diag(c(1, -1, 1))
mgg <- generate(A, n=100)

### Gap Filling Examples

dfobs <- df
dfobs[20:30, ] <- NA
n <- nrow(df)
dffill <- generate(gpcavar, n=n, exogen=exogen, gap.filling=dfobs, names=names(dfobs))

qqplot(dfobs$y, dffill$y)
abline(0,1)

### Gap filling with matrix

mgg_n <- mag
mgg_n[20:30, 2] <- NA
mkg_nfill <- generate(A, gap.filling=mgg_n)

print(mgg_n[1:31,])
print(mgg_nfill[1:31,])
dfobs2 <- df
dfobs2[20:30,2] <- NA
n <- nrow(df)
dffill2 <- generate(gpcavar,n=n,exogen=exogen,gap.filling=dfobs2,names=names(dfobs2))

qqplot(dfobs$y,dffill$y)
abline(0,1)

### generation with 'generation.matrix'
### and matrix 'x' is a covariance matrix
covariance <- array(0.5,c(3,3))

diag(covariance) <- 1

set.seed(127)
ngns <- 1000
gg1 <- generate(FUN=rnorm,n=ngns,cov=covariance)
set.seed(127)
gg2 <- generate(covariance,type="covariance",n=ngns)

### generate with a list of covariance matrix
ndim <- 5
dim <- c(ndim,ndim)
CS1 <- array(0.3,dim)
CS2 <- array(0.5,dim)
CS3 <- array(0.7,dim)
CS4 <- array(0.1,dim)

diag(CS1) <- 1
diag(CS2) <- 1
diag(CS3) <- 1
diag(CS4) <- 1

list <- list(CS1=CS1,CS2=CS2,CS3=CS3,CS4=CS4)

series <- rep(1:4,times=4,each=100)
series <- sprintf("CS%d",series)
names_A <- sprintf("%s%d",1:ndim)
ggs <- generate(list,factor.series=series,FUN=rnorm,type="covariance",names=names_A)

for (i in c("CS1","CS3")) {
  cov(ggs[[i]])
}
Index

adddate, 4

gapFilling, 2
generate, 2, 3
getVARmodel, 4

inv_GPCA, 4

rnorm, 4