Package ‘EvalEst’

May 2, 2015

Version 2015.4-2
Title Dynamic Systems Estimation - Extensions
Description Provides functions for evaluating (time series) model
 estimation methods. These facilitate Monte Carlo experiments of repeated
 simulations and estimations. Also provides methods for
 looking at the distribution of the results from these experiments,
 including model roots (which are an equivalence class invariant).
Depends R (>= 2.5.0), tfplot, dse (>= 2007.10-1)
Imports setRNG, tframe (>= 2007.5-3)
LazyLoad yes
License GPL-2
Gilbert
Author Paul Gilbert <pgilbert.tt9z@ncf.ca>
Maintainer Paul Gilbert <pgilbert.tt9z@ncf.ca>
URL http://tsanalysis.r-forge.r-project.org/
NeedsCompilation no
Repository CRAN
Date/Publication 2015-05-02 16:31:02

R topics documented:

coeff.TSmodelEstEval ............................................ 2
distribution ..................................................... 3
distribution.coefEstEval ...................................... 4
EstEval .......................................................... 5
generateSSmodel ................................................ 6
genMineData ...................................................... 7
MonteCarloSimulations ......................................... 8
nseries.MonteCarloSimulations .............................. 10
print.estimatedModels ........................................ 11
Specific Methods for \texttt{coef}

Description

See the generic function description.

Usage

```r
## S3 method for class 'TSmodelEstEval'
coef(object, criterion.args=NULL, ...)
## S3 method for class 'TSestModelEstEval'
coef(object, criterion.args=NULL, ...)
```

Arguments

- \texttt{object} an object (model) from which to extract coefficients parameters.
- \texttt{criterion.args} arguments to be passed to this method when it is called by \texttt{EstEval}.
- \texttt{...} (further arguments, currently disregarded).

Details

The methods \texttt{***.ee} are intended mainly to be called from \texttt{EstEval} as criterion for evaluating an estimation method. See \texttt{coef}.

See Also

\texttt{EstEval coef}
distribution

Generate distribution plots of Monte Carlo simulations.

Usage

distribution(obj, ...)  
## S3 method for class 'TSdata'
distribution(obj, ..., bandwidth=0.2,  
  select.inputs = seq(length=nseriesInput(obj)),  
  select.outputs= seq(length=nseriesOutput(obj)))
## Default S3 method:
distribution(obj, ..., bandwidth=0.2, series=NULL)

## S3 method for class 'MonteCarloSimulations'
distribution(obj,  
  series=seq(dim(obj$simulations)[2]),  
  x.sections=TRUE, periods=1:3, graphs.per.page=5, ...)

Arguments

obj          The result of MonteCarloSimulations.
bandwidth    passed to density or ksmooth.
series       The series which should be plotted. The default gives all series.
select.inputs  series to be plotted. (passed to selectSeries)
select.outputs series to be plotted. (passed to selectSeries)
x.sections    If TRUE then kernel density estimates are plotted for periods indicated by periods. If FALSE then a time series plots of the mean and estimates 1 and 2 standard deviations from the mean. Periods is ignored if x.sections is FALSE.
periods       The periods at which the distribution should be calculated and plotted. The default gives the first three.
graphs.per.page integer indicating number of graphs to place on a page.
...           (further arguments, currently disregarded).
select        integer vector indicating roots to be plotted. If select is not NULL then roots are sorted by magnitude and only the indicated roots are plotted. For example, select=c(1,2) will plot only the two largest roots.

Details

Kernel estimates of the densities (series by series, not joint densities) are estimated using ksmooth (if available) or density (if available) to produces density plots. Output graphics can be paused between pages by setting par(ask=TRUE).
distribution.coefEstEval

Value
None

See Also
tfplot.MonteCarloSimulations

Examples

data('eg1.DSE.data.diff', package="dse")
model <- estVARXls(eg1.DSE.data.diff)
z <- MonteCarloSimulations(model)
distribution(z)

---

distribution.coefEstEval

Plot distribution of estimates

Description

Plot distribution of estimates.

Usage

## S3 method for class 'coefEstEval'
distribution(obj, ..., Sort=FALSE, bandwidth=0.2, 
graphs.per.page=5)

## S3 method for class 'rootsEstEval'
distribution(obj, ..., mod=TRUE, invert=FALSE, Sort=FALSE, 
bandwidth=0.2, select=NULL)

Arguments

obj an object as returned by EstEval.
Sort if Sort is true then sort is applied. This helps (a bit) with estimation methods like 
black.box which may not return parameters of the same length or in the same 
order.
bandwidth passed to density or ksmooth.
graphs.per.page integer indicating number of graphs to place on a page.
... other objects to be plotted (not working for some methods).
invert logical indicating if the inverse of roots should be plotted
mod logical indicating if the modulus of roots should be plotted
select integer vector indicating roots to be plotted. If select is not NULL then roots 
are sorted by magnitude and only the indicated roots are plotted. For example, 
select=c(1,2) will plot only the two largest roots.
EstEval

Details
ksmooth is applied if available to get a smoothed estimate of the distribution of the estimates. If
ksmooth is not available then density is applied if it is available.

Value
None

See Also
EstEval

Examples

data("eg1.DSE.data.diff", package="dse")
model <- estVARXls(TSdata(output=outputData(eg1.DSE.data.diff)), max.lag=2)
# now use this as the true model
z <- EstEval(model,
estimation="estVARXls", estimation.args=list(max.lag=2))
distribution(z)
tfplot(z)

---

EstEval

Evaluate an estimation method

Description
Evaluate an estimation method.

Usage

EstEval(model, replications=100, rng=NULL, quiet=FALSE,
simulation.args=NULL,
estimation=NULL, estimation.args=NULL,
criterion ="coef", criterion.args =NULL)

is.EstEval(obj)

Arguments

model A TSmodel.
replications The number of simulations.
rng The RNG and starting seed.
quiet If TRUE then no information is printed during estimation.
simulation.args A list of any arguments to pass to simulate.
estimation A character string indicating the estimation routine to use.
Randomly generate a state space model.

generateSSmodel

estimation.args
A list of any arguments to pass to the estimation routine.

criterion
A function to apply to the results of estimation to extract the information which is to be retained.

criterion.args
A list of any arguments to be passed to the criterion function.

obj
an object.

Details

estimation.args and criterion.args should be NULL if no args are needed. If model is an object of class 'EstEval' or 'simulation' then the model and the seed are extracted so the evaluation will be based on the same generated sample. criterion can be 'coef', 'roots', 'TModel', or 'TSestModel'. With the default (coef) or with TModel the other criteria can be reconstructed (when the estimation method finds a known form for the model - which is not always the case, for example with estBlackBox methods). If criterion = 'roots' then criterion.args= list(verbose=FALSE) is advised.

Value

A list with element result of length replications, each element containing the results of criterion(estimation(simulate(model))). Other elements of the list contain information from the supplied arguments.

See Also

simulate MonteCarloSimulations distribution forecastCovWRTtrue

Examples

```r
data("eg1.DSE.data.diff", package="dse")
model <- estVARXls(TSdata(output=outputData(eg1.DSE.data.diff)))
z <- EstEval(model,
        estimation="estVARXls", estimation.args=list(max.lag=2))
tfplot(z)
zz <- EstEval(model,
        estimation="estVARXls", estimation.args=list(max.lag=2),
        simulation.args=list(sampleT=50, sd=1.5))
is.EstEval(z)
```

---

**generateSSmodel**

**Randomly generate a state space model**

**Description**

Randomly generate a state space model.
Usage

`generateSSmodel(m, n, p, stable=FALSE)`

Arguments

- `m`, `n`, `p`: Input, state and output dimensions.
- `stable`: TRUE or FALSE indicating if the model must be stable.

Details

Randomly generate a state space model. If stable is true then the largest root will have magnitude less than 1.0.

Value

An SS TSmodel.

Examples

```
z <- generateSSmodel(2, 3, 1)
```

Description

Generate data for Monte Carlo experiments

Usage

```r
genMineData(umodel, ymodel, uinput=NULL, sampleT=100,
              unoise=NULL, usd=1, ynoise=NULL, ysd=1, rng=NULL)
build.input.models(data, max.lag=NULL)
build.diagonal.model(multi.models)
```

Arguments

- `umodel`: Model for input data.
- `ymodel`: Model for output data.
- `sampleT`: Number of periods of data to generate.
- `unoise`: Input noise.
- `usd`: Standard deviation of input noise.
- `ynoise`: Output noise.
- `ysd`: Standard deviation of output noise.
- `rng`: RNG setting.
Monte Carlo Simulations

multi.models A list of TSestModels.
data data from which to build models.
max.lag number of lags in the estimated models.
uinput Input data to umodel.

Details

This function generates test data using specified models. umodel is used to generate data corresponding to input data and ymodel is used to generate data corresponding to output data. The result of umodel is used as input to ymodel so the input dimension of ymodel should be the output dimension of umodel. Typically the ymodel would be degenerate in some of the input variables so the effective inputs are a subset. If umodel requires input data it should be specified in uinput. If noise is NULL then an normal noise will be generated by simulate. This will be iid N(0,1). The RNG will be set first to rng if it is specified. If unoise or ynoise are specified they should be as expected by simulate for the specified umodel and ymodel.

genMineData uses build.input.models, which makes a list of univariate TSestModels, one for each series in inputData(data) estimated by estVARXls with max.lag lags. genMineData then uses build.diagonal.model which builds one diagonal model from a list of models returned by build.input.models. It uses the AR part only.

Value

A TSdata object.

See Also

simulate

Examples

data("eg1.DSE.data.diff", package="dse")
umodel <- build.diagonal.model(
    build.input.models(eg1.DSE.data.diff, max.lag=2))
z <- TSdata(output=outputData(eg1.DSE.data.diff),
            input = inputData(eg1.DSE.data.diff))
ymodel <- TSmodel(estVARXls(z, max.lag=3))
sim.data <- genMineData(umodel, ymodel)

MonteCarloSimulations Generate simulations

Description

Run multiple simulations
Usage

is.MonteCarloSimulations(obj)
MonteCarloSimulations(model, simulation.args=NULL,
    replications=100, rng=NULL, quiet = FALSE, ...)
## Default S3 method:
MonteCarloSimulations(model, simulation.args=NULL,
    replications = 100, rng = NULL, quiet = FALSE, ...)
## S3 method for class 'TSmodel'
MonteCarloSimulations(model, simulation.args=NULL,
    replications=100, rng=NULL, quiet=FALSE, ...)
## S3 method for class 'TSestModel'
MonteCarloSimulations(model, simulation.args=NULL,
    replications=100, rng=NULL, quiet=FALSE, ...)
## S3 method for class 'EstEval'
MonteCarloSimulations(model, simulation.args=NULL,
    replications=100, rng=getRNG(model), quiet=FALSE, ...)
## S3 method for class 'MonteCarloSimulations'
MonteCarloSimulations(model,
    simulation.args=NULL, replications=100, rng=getRNG(model), quiet=FALSE, ...)

Arguments

model     an object from which a model can be extracted. The model must have an associated simulation method (e.g. a TSmodel).
simulation.args,
    A list of arguments in addition to model which are passed to simulate.
replications The number of simulations.
rng       The RNG and starting seed.
quiet      logical indicating if printing and many warning messages should be suppressed.
obj        an object.
...        arguments passed to other methods.

Details

This function runs many simulations using simulate. Often it not be necessary to do this since the seed can be used to reproduce the sample and many functions for testing estimation methods, etc., will produce samples as they proceed. This function is useful for verification and for looking at the stochastic properties of the output of a model. If model is an object of class EstEval or simulation then the model and the seed!!! are extracted so the same sample will be generated. The default method expects the result of simulate(model) to be a matrix. There is a tfplot method (time series plots of the simulations) and a distribution method for the result. The latter plots kernel estimates of the distribution of the simulations at specified periods.

Value

A list of simulations.
nseries.MonteCarloSimulations

See Also

simulate EstEval distribution forecastCovWRTtrue

Examples

data("eg1.DSE.data.diff", package="dse")
model <- estVARXls(eg1.DSE.data.diff)
z <- MonteCarloSimulations(model, simulation.args=list(sampleT=100))
tfplot(z)
distribution(z)

nseries.MonteCarloSimulations

Number of Series

Description

Return the number of series.

Usage

## S3 method for class 'MonteCarloSimulations'
nseriesInput(x)
## S3 method for class 'MonteCarloSimulations'
nseriesOutput(x)

Arguments

x A featherForecasts object.

Details

See the generic method.

Value

An integer.
print.estimatedModels  

Print Specific Methods

Description
See the generic function description.

Usage

```r
## S3 method for class 'EstEval'
print(x, digits=options()$digits, ...)
## S3 method for class 'MonteCarloSimulations'
print(x, digits=options()$digits, ...)
```

Arguments

- `x` an object to be printed.
- `digits` a non-null value is used to indicate the number of significant digits. If `digits` is NULL then the value of `digits` specified by `options` is used.
- `...` (further arguments, currently disregarded).

See Also

- `print` summary

roots.coefEstEval  

Roots Specific Methods

Description
See the generic function description.

Usage

```r
## S3 method for class 'coefEstEval'
roots(obj, criterion.args=NULL, ...)
## S3 method for class 'rootsEstEval'
roots(obj, ...)
## S3 method for class 'TSestModelEstEval'
roots(obj, criterion.args=NULL, ...)
## S3 method for class 'TSmodelEstEval'
roots(obj, criterion.args=list(randomize = TRUE), ...)
```
Arguments

- **obj**: an object from which roots are to be extracted or calculated and printed.
- **criterion.args**: arguments to be passed to this method when it is called by `EstEval`.
- **...**: arguments to be passed to other methods.

Details

The methods `***.ee` are intended mainly to be called from `EstEval` as criterion for evaluating an estimation method.

See Also

- `roots` stability `EstEval`

---

### SeriesNamesInput.MonteCarloSimulations

*TS Input and Output Specific Methods*

Description

See the generic function description.

Usage

```r
## S3 method for class 'MonteCarloSimulations'
seriesNamesInput(x)
## S3 method for class 'MonteCarloSimulations'
seriesNamesOutput(x)
```

Arguments

- **x**: an object from which to extract the names of the input or output series.

---

### Summary.EstEval

*Summary Specific Methods*

Description

See the generic function description.
Usage

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

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

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

## S3 method for class 'MonteCarloSimulations'
summary(object, series=NULL, periods=1:3, ...)

## S3 method for class 'coefEstEval'
summary(object, verbose=TRUE, ...)

## S3 method for class 'rootsEstEval'
summary(object, verbose=TRUE, ...)

## S3 method for class 'summary.TTestModelEstEval'
print(x, digits=options()$digits, ...)

## S3 method for class 'summary.TModelEstEval'
print(x, digits=options()$digits, ...)

## S3 method for class 'summary.EstEval'
print(x, digits=options()$digits, ...)

## S3 method for class 'summary.MonteCarloSimulations'
print(x, digits=options()$digits, ...)

## S3 method for class 'summary.coefEstEval'
print(x, digits=options()$digits, ...)

## S3 method for class 'summary.rootsEstEval'
print(x, digits=options()$digits, ...)

Arguments

- **object**: an object for which a summary is to be printed.
- **x**: an object for which a summary is to be printed.
- **digits**: a non-null value is used to indicate the number of significant digits. If digits is NULL then the value of digits specified by options is used.
- **series**: The series which should be plotted. The default NULL gives all series.
- **periods**: optional integer vector indicating periods at which the summary should be calculated.
- **verbose**: logical indicating if a longer summary should be produced.
- **...**: arguments passed to other methods.

See Also

- `summary`, `print`
Description

See the generic function description.

Usage

```r
## S3 method for class 'EstEval'
testEqual(obj1, obj2, fuzz=0)
## S3 method for class 'MonteCarloSimulations'
testEqual(obj1, obj2, fuzz=1e-16)
```

Arguments

- `obj1`: an object which is to be compared with the second object.
- `obj2`: an object which is to be compared with the first object.
- `fuzz`: tolerance for numerical comparisons. Values within fuzz will be considered equal.

See Also

testEqual

---

Specific methods for `coefEstEval` objects

Description

See the generic function description.

Usage

```r
## S3 method for class 'coefEstEval'
tfplot(x, cumulate=TRUE, norm=FALSE, bounds=TRUE,
       invert=FALSE, Sort=FALSE, graphs.per.page = 5, ...)
```
tfplot.MonteCarloSimulations

Arguments

- **x**: an object for which a tfplot is to be produced.
- **cumulate**: logical indicating if the cumulative average of roots should be plotted.
- **invert**: logical indicating if the inverse of roots should be plotted.
- **Sort**: logical indicating if the roots should be sorted.
- **graphs.per.page**: integer indicating number of graphs to place on a page.
- **norm**: logical indicating if the euclidean norm of roots should be plotted (square root of the sum of squared roots).
- **bounds**: logical indicating if estimated one standard error bounds should be plotted around the lines for the true roots.
- **...**: arguments passed to other methods.

Details

If cumulate is true the cumulative average is plotted. If norm is true the norm is used, each parameter is plotted. If invert is true the reciprocal is used (before cumulating). If Sort is true then sort is applied (before ave). This is not usually recommended but of interest with estimation methods like black.box which may not return parameters of the same length or in the same order. Plotting the true lines only makes sense if truth is the same length as result (and sometimes not even then).

See Also

tfplot EstEval

tfplot.MonteCarloSimulations

Generate plots of Monte Carlo simulations

Description

Generate plots of Monte Carlo simulations.

Usage

```r
## S3 method for class 'MonteCarloSimulations'
tfplot(x,
    tf=tframe(x$simulations), start=tfstart(tf), end=tfend(tf),
    series=seq(dim(x$simulations)[2]),
    select.simulations=seq(dim(x$simulations)[3]),
    graphs.per.page=5, mar=par()$mar, ...)```
tfplot.rootsEstEval

Arguments

x The result of MonteCarloSimulations.
tf The time frame for plots. see tfplot.
start The starting period for plots, taken from tf by default.
end The ending period for plots, taken from tf by default.
series The series which should be plotted. The default NULL gives all series.
select.simulations Vector of integers indicating the simulations which should be plotted. The de-
fault plots all simulations.
graphs.per.page The number of graphs to put on a page.
mar Plot margins (see par).
... arguments passed to other methods.

Details

This function produces plots of the simulated series. Output graphics can be paused between pages
by setting par(ask=TRUE).

Value

None

See Also

distribution.MonteCarloSimulations

Examples

data("eg1.DSE.data.diff", package="dse")
model <- estVARXls(eg1.DSE.data.diff)
z <- MonteCarloSimulations(model)
tfplot(z)

---
tfplot.rootsEstEval Specific tfplot methods for rootsEstEval (EstEval) objects

Description

See the generic function description.
Usage

```r
## S3 method for class 'rootsEstEval'
tfplot(x, ...)
## S3 method for class 'rootsEstEval'
plot(x, complex.plane=TRUE, cumulate=TRUE, norm=FALSE,
     bounds=TRUE, transform=NULL, invert=FALSE, Sort=TRUE, ...)
```

Arguments

- `x` an object for which a tfplot is to be produced.
- `complex.plane` logical indicating if the plot should be on the complex plane.
- `cumulate` logical indicating if the cumulative average of roots should be plotted.
- `invert` logical indicating if the inverse of roots should be plotted.
- `Sort` logical indicating if the roots should be sorted.
- `...` arguments passed to other methods.
- `norm` logical indicating if the euclidean norm of roots should be plotted (square root of the sum of squared roots).
- `bounds` logical indicating if estimated one standard error bounds should be plotted around the lines for the true roots.
- `transform` an optional string indicating the name of a function which should be applied to the roots before plotting.

Details

If `complex.plane` is TRUE then all results are plotted on a complex plane and the arguments `cumulate` and `Sort` do not apply. If `complex.plane` is FALSE then a sequential plot of the real and imaginary parts is produced. If `cumulate` is true the cumulative average is plotted. If `mod` is true the modulus is used, otherwise real and imaginary are separated. If `invert` is true the reciprocal is used (before cumulating). If `Sort` is true then sort is applied (before cumulate but after `mod`) by the `Re` part of the root. Some grouping is usually necessary since roots are not in an obvious order but sorting by the real part of the roots could be improved upon.

See Also

- `tfplotEstEval`

Specific Methods for `tfplot`

Description

See the generic function description.
Usage

```r
## S3 method for class 'TSmodelEstEval'
tfplot(x, graph.args=NULL,
        criterion ="coef", criterion.args=NULL, ...)
## S3 method for class 'TSestModelEstEval'
tfplot(x, graph.args=NULL,
        criterion ="coef", criterion.args=NULL, ...)
## S3 method for class 'EstEval'
tfplot(x, tf=NULL, start=tfstart(tf), end=tfend(tf),
        truth= if(is.TSdata(x$truth)) outputData(x$truth) else x$truth,
        series = seq(length=nseries(truth)),
        Title="Estimated (and true) results",
        ylab = seriesNames(truth), remove.mean = FALSE,
        graphs.per.page=5, mar=par()$mar, reset.screen=TRUE, ...)
```

Arguments

- **x**: an object for which a tfplot is to be produced.
- **tf**: see `tfplot`.
- **start**: see `tfplot`.
- **end**: see `tfplot`.
- **truth**: true value which will be plotted along with estimates.
- **Title**: string of characters to use for title.
- **remove.mean**: logical indicating if means should be removed before plotting results.
- **ylab**: vector of strings for y axis labelling.
- **graphs.per.page**: integer indicating number of graphs to place on a page.
- **reset.screen**: logical indicating if the plot window should be cleared before starting.
- **series**: integer or string indicating the series which should be plotted.
- **mar**: plot margins. See `par`.
- **graph.args**: list of graphics arguments eventually passed to `plot`. See `par`.
- **criterion**: criterion which should be used to extract something from the object which will then be plotted. See `EstEval`.
- **criterion.args**: arguments to be passed to criterion.
- **...**: arguments passed to other methods.

See Also

- `tfplot EstEval`
Description
Return the number of Observations or the tframe.

Usage
```r
## S3 method for class 'MonteCarloSimulations'
Tobs(x)
## S3 method for class 'MonteCarloSimulations'
tframe(x)
```

Arguments

- `x` A MonteCarloSimulations object.

Details
See the generic method.

Value
An integer or a tframe object.

---

TobSimMonteCarloSimulations

Description

TS Extractor Specific Methods

Description
See the generic function description.

Usage
```r
## S3 method for class 'coefEstEval'
TSeestModel(obj)
## S3 method for class 'coefEstEval'
TSmodel(obj, ...)
```

Arguments

- `obj` an object from which to extract the TSmodel or TSeestModel.
- `...` arguments to be passed to other methods.
See Also

TSdata TSestModel TSmodel
Index

+Topic **programming**
  - nseries.MonteCarloSimulations, 10
  - TobsMonteCarloSimulations, 19
+Topic **ts**
  - coef.TSmodelEstEval, 2
  - distribution, 3
  - distribution.coefEstEval, 4
  - EstEval, 5
  - generateSSmodel, 6
  - genMineData, 7
  - MonteCarloSimulations, 8
  - nseries.MonteCarloSimulations, 10
  - print.estimatedModels, 11
  - roots.coefEstEval, 11
  - seriesNamesInput.MonteCarloSimulations, 12
  - summary.EstEval, 12
  - testEqual.EstEval, 14
  - tfplot.coefEstEval, 14
  - tfplot.MonteCarloSimulations, 15
  - tfplot.rootsEstEval, 16
  - tfplot.TSdata.ee, 17
  - TobsMonteCarloSimulations, 19
  - TSdata.coefEstEval, 19
+Topic **utilities**
  - nseries.MonteCarloSimulations, 10
  - TobsMonteCarloSimulations, 19

  - build.diagonal.model (genMineData), 7
  - build.input.models (genMineData), 7

  - coef, 2
  - coef.TSestModelEstEval
    (coef.TSmodelEstEval), 2
  - coef.TSmodelEstEval, 2

  - distribution, 3, 6, 10
  - distribution.coefEstEval, 4
  - distribution.MonteCarloSimulations, 16

  - distribution.rootsEstEval
    (distribution.coefEstEval), 4

  - EstEval, 2, 5, 10, 12, 15, 17, 18

  - forecastCovWRTtrue, 6, 10

  - generateSSmodel, 6
  - genMineData, 7

  - is.EstEval(EstEval), 5
  - is.MonteCarloSimulations
    (MonteCarloSimulations), 8

  - MonteCarloSimulations, 6, 8

  - nseries.MonteCarloSimulations, 10
  - nseriesInput.MonteCarloSimulations
    (nseries.MonteCarloSimulations), 10

  - nseriesOutput.MonteCarloSimulations
    (nseries.MonteCarloSimulations), 10

  - print.rootsEstEval
    (tfplot.rootsEstEval), 16

  - print, 11, 13

  - print.EstEval (print.estimatedModels), 11

  - print.estimatedModels, 11

  - print.MonteCarloSimulations
    (print.estimatedModels), 11

  - print.summary.coefEstEval
    (summary.EstEval), 12

  - print.summary.EstEval
    (summary.EstEval), 12

  - print.summary.MonteCarloSimulations
    (summary.EstEval), 12

  - print.summary.rootsEstEval
    (summary.EstEval), 12
print.summary.TSeEstModelEstEval
  (summary.EstEval), 12
print.summary.TSmodelEstEval
  (summary.EstEval), 12
roots, 12
roots.coefEstEval, 11
roots.rootsEstEval (roots.coefEstEval), 11
roots.TSeEstModelEstEval
  (roots.coefEstEval), 11
roots.TSmodelEstEval
  (roots.coefEstEval), 11
seriesNamesInput.MonteCarloSimulations,
  12
seriesNamesOutput.MonteCarloSimulations
  (seriesNamesInput.MonteCarloSimulations),
  12
simulate, 6, 8, 10
stability, 12
summary, 11, 13
summary.coefEstEval (summary.EstEval),
  12
summary.EstEval, 12
summary.MonteCarloSimulations
  (summary.EstEval), 12
summary.rootsEstEval (summary.EstEval),
  12
summary.TSeEstModelEstEval
  (summary.EstEval), 12
summary.TSmodelEstEval
  (summary.EstEval), 12
testEqual, 14
testEqual.EstEval, 14
testEqual.MonteCarloSimulations
  (testEqual.EstEval), 14
tfplot, 15, 17, 18
tfplot.coefEstEval, 14
tfplot.Estim (tfplot.TSdata.ee), 17
tfplot.MonteCarloSimulations, 4, 15
tfplot.rootsEstEval, 16
tfplot.TSdata.ee, 17
tfplot.TSeEstModelEstEval
  (tfplot.TSdata.ee), 17
tfplot.TSmodelEstEval
  (tfplot.TSdata.ee), 17