Package ‘ternarynet’

November 21, 2016

Version 1.18.0
Date 2014-11-17
Title Ternary Network Estimation
Description A computational Bayesian approach to ternary gene regulatory network estimation from gene perturbation experiments.
Author Matthew N. McCall <mccallm@gmail.com>, Anthony Almudevar <Anthony_Almudevar@urmc.rochester.edu>
Maintainer Matthew N. McCall <mccallm@gmail.com>
Depends R (>= 2.10.0), methods
Imports utils, igraph
License GPL (>= 2)
biocViews Software, CellBiology, GraphAndNetwork
NeedsCompilation yes

R topics documented:

attractorSummary .................................................. 2
graphPosterior ..................................................... 3
plotFit ............................................................... 3
plotPost ............................................................. 4
plotTraces ............................................................ 5
predictAttractor ..................................................... 6
simulateSteadyState ................................................ 7
ternaryFit-class ..................................................... 8
ternaryFitParameters-class ...................................... 9
ternaryPost-class .................................................. 10
tnetfit ............................................................... 11
tnetpost ............................................................. 12

Index 14
attractorSummary

Summarize Attractors

Description

This function summarizes the posterior probability of possible attractors.

Usage

attractorSummary(tpost, post.prob.limit = 0.01, wildtype = TRUE)

Arguments

tpost a ternaryPost object
post.prob.limit the minimum posterior probability for an attractor to be listed
wildtype if TRUE, the wildtype attractors are summarized; if FALSE, the perturbed attractors are summarized.

Value

The function returns a matrix of attractors and posterior probabilities for each perturbation.

Author(s)

Matthew N. McCall and Anthony Almudevar

See Also


Examples

ssObj <- matrix(c(1,1,1,0,1,0,1,0,1),nrow=3)
pObj <- matrix(c(1,0,0,0,1,0,0,0,1),nrow=3)
rownames(ssObj) <- rownames(pObj) <- colnames(ssObj) <- colnames(pObj) <- c("Gene1","Gene2","Gene3")
tnfitObj <- tnetfit(ssObj, pObj)
tnpostObj <- tnetpost(tnfitObj, mdelta=10, msample=10)
attractorSummary(tnpostObj)
**graphPosterior**

**Network Topology**

**Description**

This function summarizes the topology of the ternary network using marginal edge probabilities.

**Usage**

```r
graphPosterior(tpost)
```

**Arguments**

- `tpost`: a ternaryPost object

**Value**

The function returns a matrix of marginal posterior probabilities of each possible network edge – rows are children and columns are parents. The first column represents no parents.

**Author(s)**

Matthew N. McCall and Anthony Almudevar

**See Also**


**Examples**

```r
ssObj <- matrix(c(1,1,1,1,1,0,1,0,0,1),nrow=3)
pObj <- matrix(c(1,0,0,0,1,0,0,0,1),nrow=3)
rownames(ssObj) <- rownames(pObj) <- colnames(ssObj) <- colnames(pObj) <- c("Gene1","Gene2","Gene3")

tnfitObj <- tnetfit(ssObj, pObj)

tnpostObj <- tnetpost(tnfitObj, mdelta=10, msample=10)

graphPosterior(tnpostObj)
```

---

**plotFit**

**Network Fit Plot**

**Description**

This function plots the graph corresponding to the minimum scoring network.

**Usage**

```r
plotFit(ternaryFit, type="interactive", ...)
```
Arguments

ternaryFit  a ternaryFit object

type  the type of plot to produce. "interactive" produces a plot that can be altered in the plotting window using the tkplot function from the igraph package. "static" produces a standard plot in any R graphics device.

Value

A plot of the network corresponding to the minimum score (stored in the graphObjMin slot) is plotted.

Author(s)

Matthew N. McCall and Anthony Almudevar

See Also


Examples

```r
ssObj <- matrix(c(1,1,1,0,1,0,1,0,1),nrow=3)
pObj <- matrix(c(1,0,0,0,1,0,0,0,1),nrow=3)
tnfitObj <- tnetfit(ssObj, pObj)
plotFit(tnfitObj, type="static")
```

plotPost  Network Posterior Plot

Description

This function plots the graph consisting of all edges with a marginal posterior probability greater than the selected threshold.

Usage

`plotPost(ternaryPost, threshold=0.5, type="interactive", ...)`

Arguments

ternaryPost  a ternaryPost object

type  the type of plot to produce. "interactive" produces a plot that can be altered in the plotting window using the tkplot function from the igraph package. "static" produces a standard plot in any R graphics device.

threshold  the marginal posterior probability required for an edge to be included in the plot.

...  additional parameters passed to the plotting function
Value

A plot of the network consisting of all edges with a marginal posterior probability greater than the selected threshold.

Author(s)

Matthew N. McCall and Anthony Almudevar

See Also


Examples

```r
ssObj <- matrix(c(1,1,1,1,0,1,0,0,1),nrow=3)
pObj <- matrix(c(1,0,0,1,0,0,0,0,1),nrow=3)
lnetfitObj <- tnetfit(ssObj, pObj)
lnpostObj <- tnetpost(lnetfitObj, mdelta=10, msample=10)
plotPost(lnpostObj, type="static")
```

---

**plotTraces**

*Network Fit Traces*

**Description**

This function plots the trace of four model parameters.

**Usage**

```r
plotTraces(tfit)
```

**Arguments**

- `tfit` a ternaryFit object

**Value**

The function creates a 2x2 grid of the four trace plots.

**Author(s)**

Matthew N. McCall and Anthony Almudevar

**See Also**

predictAttractor

**Examples**

```r
ssObj <- matrix(c(1,1,1,0,1,0,0,1),nrow=3)
pObj <- matrix(c(1,0,0,1,0,0,1),nrow=3)
tnfitObj <- tnetfit(ssObj, pObj)
plotTraces(tnfitObj)
```

**Description**

This function computes the posterior probabilities of attractors reached for a given perturbation using the networks from a ternaryPost object.

**Usage**

```r
predictAttractor(tpost, perturbations, wildtype = TRUE, verbose = FALSE)
```

**Arguments**

- `tpost`: a ternaryPost object
- `perturbations`: a list with two elements: perturbed.genes and forced.states
- `wildtype`: if TRUE, the wildtype attractors are summarized; if FALSE, the perturbed attractors are summarized.
- `verbose`: if TRUE, periodic reports on progress are printed.

**Value**

The function returns a list with two elements:
- `post.prob`: the posterior probability of each attractor
- `attractor.summary`: a single vector of steady states based on the resulting attractor

**Author(s)**

Matthew N. McCall and Anthony Almudevar

**See Also**


**Examples**

```r
ssObj <- matrix(c(1,1,1,0,1,0,0,1),nrow=3)
pObj <- matrix(c(1,0,0,1,0,0,1),nrow=3)
rownames(ssObj) <- rownames(pObj) <- colnames(ssObj) <- colnames(pObj) <- c("Gene1","Gene2","Gene3")
tnfitObj <- tnetfit(ssObj, pObj)
tnpostObj <- tnetpost(tnfitObj, mdelta=10, msample=10)
predictAttractor(tnpostObj, list(perturbed.genes=c(1,2),forced.states=c(1,1)))
```
**simulateSteadyState**  
*Simulate Steady State Data*

**Description**
This function generates simulated steady state data from a given network.

**Usage**

```r
simulateSteadyState(perturbationObj, tableObj, graphObj, degreeObj, wildtype=FALSE)
```

**Arguments**
- **perturbationObj**: a matrix of perturbation experiments. Rows are genes and columns are experiments.
- **tableObj**: a matrix containing the transition function tables
- **graphObj**: a matrix containing the parents of each node
- **degreeObj**: a vector containing the in-degree of each node
- **wildtype**: if TRUE, the preturbations are assumed to be transient; if FALSE, the perturbations are assumed to be persistent.

**Value**
The function creates a steadyStateObj.

**Author(s)**
Matthew N. McCall and Anthony Almudevar

**See Also**

**Examples**
```r
pObj <- matrix(c(1,0,0,0,1,0,0,0,1),nrow=3)
degreeObj <- c(0,1,1)
graphObj <- matrix(nrow=1,ncol=3)
graphObj[1,1] <- 0
graphObj[1,2] <- 1
graphObj[1,3] <- 2
tableObj <- matrix(nrow=3,ncol=3)
tableObj[,1] <- rep(0,3)
tableObj[,2] <- c(-1,0,1)
tableObj[,3] <- c(-1,0,1)
ssObj <- simulateSteadyState(pObj, tableObj, graphObj, degreeObj)
```
ternaryFit-class

Ternary Network Fit

Description

This is a class representation of the output of the ternary network fitting algorithm implemented in the function tnetfit.

Creating Objects

While one can create their own objects using the function ternaryFit(), this is highly discouraged. Typically this class is created by running the tnetfit function.

Slots

perturbationObj: a matrix of perturbation experiments. Rows are genes and columns are experiments.
steadyStateObj: a matrix of steady gene expression observations from a perturbation experiment. Rows are genes and columns are experiments.
geneNames: a vector of gene names corresponding to the rows of the perturbationObj and steadyStateObj.
experimentNames: a vector of experiment names corresponding to the columns of the perturbationObj and steadyStateObj.
degreeObjMin: a vector containing the in-degree of each node in the fit achieving the minimum score
graphObjMin: a matrix containing the parents of each node in the fit achieving the minimum score
tableObjMin: a matrix containing the table in the fit achieving the minimum score
newScore: the most recent score
minScore: the minimum score
finalTemperature: the final value of the temperature parameter
traces: a dataframe contain the traces for 4 parameters
stageCount: the number of stages
xSeed: the random seed.
inputParams: the ternaryFitParameters object used.

Methods

All named elements can be accessed and set in the standard way (e.g. xSeed(object) and xSeed(object)<-).

Author(s)

Matthew N. McCall and Anthony Almudevar

See Also

ternaryFitParameters-class

Examples

```r
ssObj <- matrix(c(1,1,1,0,1,0,0,1),nrow=3)
pObj <- matrix(c(1,0,0,0,1,0,0,0,1),nrow=3)
tnfitObj <- tnetfit(ssObj, pObj)
class(tnfitObj)
```

ternaryFitParameters-class

Ternary Network Fitting Parameters

Description

This is a class representation of the input parameters for the ternary network fitting algorithm implemented in the function `tnetfit`.

Creating Objects

`ternaryFitParameters()`

This creates a `ternaryFitParameters` object with the default fitting parameters.

Slots

- `perturbationType`: this parameter currently can only be set to 1
- `scoreType`: the method to score networks. Can be set to either 1 or 2, corresponding the the score types in Almudevar et al. (2011).
- `backupStage`: current fit is output periodically according to this parameter
- `maxStage`: the maximum number of stages permitted. Ideally, the actual number of stages required until convergence should be much less than this value.
- `maxTransition`: This parameter provides an adaptive truncation of the stage sample size. The stage terminates before the specified fixed sample size if the number of transitions resulting in a strict increase of the score reaches this value. If the sampler is in steady state, then this count should be approximately half the number of transitions in which the score changes value.
- `epsilon`: Convergence tolerance.
- `beta0`: Algorithm terminates when this number of consecutive convergence events have occurred.
- `chi0`: The target initial acceptance rate. This should be close to 1, although setting it too close will increase computation time.
- `delta`: The increment change in steady state distribution between stages (as variational distance). Larger values tend to decrease computation time, but too large a value will result in spurious convergence.
- `ne`: The fixed sample size (number of MCMC transitions) per stage.
- `m0`: The sample size (number of transitions) used to determine the initial temperature.
- `maxDegree`: Maximum number of parents per node permitted in model topology.
- `pAddParent`: This is the probability of adding a parent to a randomly selected node in the proposal function.
- `pExchangeParent`: This parameter gives the probability of a parent exchange in the proposal function. The AddParent operation takes precedence, so this probability should be interpreted as being conditional on the rejection of the AddParent operation.
neighborDegree: Number of applications of the proposal function.

pNeighborhood: Vector of probabilities denoted, which generates the random number of proposal function iterations. The length is one less than neighborDegree. If neighborDegree equals 1 then no iteration is performed, and this vector is ignored.

rho: Weight parameter for the exponential smoothing of the variance estimate. For no smoothing set to 1.

degree: This parameter provides a complexity penalty. This number times the number of edges is added to the score. To apply no penalty set this parameter to 0.

Methods

All named elements can be accessed and set in the standard way (e.g. \texttt{scoreType(object)} and \texttt{scoreType(object)<-}).

Author(s)

Matthew N. McCall and Anthony Almudevar

See Also


Examples

# create an instance
ternaryFitParameters()

---

ternaryPost-class Ternary Network Posterior

Description

This is a class representation of the output of the ternary network posterior sampling algorithm implemented in the function tnetpost.

Creating Objects

While one can create their own objects using the function \texttt{ternaryPost()}, this is highly discouraged. Typically this class is created by running the \texttt{tnetpost} function.

Slots

perturbationObj: a matrix of perturbation experiments. Rows are genes and columns are experiments.

steadyStateObj: a matrix of steady gene expression observations from a perturbation experiment. Rows are genes and columns are experiments.

geneNames: a vector of gene names corresponding to the rows of the perturbationObj and steadyStateObj.
experimentNames: a vector of experiment names corresponding to the columns of the perturbationObj and steadyStateObj.
scores: the score of each sample
degreeObjs: the in-degree vector for each sample
graphObjs: the graph matrix for each sample
tableObjs: the table matrix for each sample
inputParams: the ternaryFitParameters object used

Methods

All named elements can be accessed and set in the standard way (e.g. scores(object) and scores(object)<-).

Author(s)

Matthew N. McCall and Anthony Almudevar

See Also


Examples

ssObj <- matrix(c(1,1,1,0,1,1,0,0,1),nrow=3)
pObj <- matrix(c(1,0,0,0,1,0,0,0,1),nrow=3)
tnfitObj <- tnetfit(ssObj, pObj)
tnpostObj <- tnetpost(tnfitObj, mdelta=10, msample=10)
class(tnpostObj)

---

Description

This function fits a ternary network based on perturbation experiments.

Usage

tnetfit(steadyStateObj, perturbationObj, params=ternaryFitParameters(), xSeed=NA)

Arguments

steadyStateObj a matrix of steady gene expression observations from a perturbation experiment. Rows are genes and columns are experiments.
perturbationObj a matrix of perturbation experiments. Rows are genes and columns are experiments.
params a ternaryFitParameters object
xSeed an integer random seed. If NA, a random seed is generated.
\textbf{Value}

The function returns a ternaryFit object.

\textbf{Author(s)}

Matthew N. McCall and Anthony Almudevar

\textbf{See Also}


\textbf{Examples}

\begin{verbatim}
ssObj <- matrix(c(1,1,0,1,0,1),nrow=3)
pObj <- matrix(c(1,0,0,1,0,0),nrow=3)
rownames(ssObj) <- rownames(pObj) <- colnames(ssObj) <- colnames(pObj) <- c("Gene1","Gene2","Gene3")

# tnetfit function

tnfitObj <- tnetfit(ssObj, pObj)
\end{verbatim}

---

\textbf{tnetpost \hspace{1cm} Ternary Network Posterior Sampling}

\textbf{Description}

This function samples from the posterior density of a ternary network based on perturbation experiments.

\textbf{Usage}

\begin{verbatim}
tnetpost(tfit, mdelta=as.integer(10000), msample=as.integer(2000), temperatureScale=1.0, xSeed=NA)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{tfit} \hspace{1cm} a ternaryFit object
  \item \texttt{mdelta} \hspace{1cm} number of transitions between samples
  \item \texttt{msample} \hspace{1cm} number of samples
  \item \texttt{temperatureScale} \hspace{1cm} the final temperature is multiplied by this value for sampling
  \item \texttt{xSeed} \hspace{1cm} an integer random seed. If NA, a random seed is generated.
\end{itemize}

\textbf{Value}

The function returns a ternaryPost object.

\textbf{Author(s)}

Matthew N. McCall and Anthony Almudevar
See Also


Examples

```
ssObj <- matrix(c(1,1,1,1,1,1,1,0,1),nrow=3)
pObj <- matrix(c(1,0,0,1,0,0,1,0,0),nrow=3)
rownames(ssObj) <- rownames(pObj) <- colnames(ssObj) <- colnames(pObj) <- c("Gene1","Gene2","Gene3")

tnfitObj <- tnetfit(ssObj, pObj)

tnpostObj <- tnetpost(tnfitObj, mdelta=10, msample=10)
```
Index

*Topic classes
  ternaryFit-class, 8
  ternaryFitParameters-class, 9
  ternaryPost-class, 10

*Topic manip
  attractorSummary, 2
  graphPosterior, 3
  plotFit, 3
  plotPost, 4
  plotTraces, 5
  predictAttractor, 6
  simulateSteadyState, 7
  tnetfit, 11
  tnetpost, 12

attractorSummary, 2
backupStage
  (ternaryFitParameters-class), 9
  backupStage, ternaryFitParameters-method
  (ternaryFitParameters-class), 9
  backupStage<-
  (ternaryFitParameters-class), 9
  backupStage<-, ternaryFitParameters-method
  (ternaryFitParameters-class), 9
  beta0 (ternaryFitParameters-class), 9
  beta0, ternaryFitParameters-method
  (ternaryFitParameters-class), 9
  beta0<- (ternaryFitParameters-class), 9
  beta0<-, ternaryFitParameters-method
  (ternaryFitParameters-class), 9
  chi0 (ternaryFitParameters-class), 9
  chi0, ternaryFitParameters-method
  (ternaryFitParameters-class), 9
  chi0<- (ternaryFitParameters-class), 9
  chi0<-, ternaryFitParameters-method
  (ternaryFitParameters-class), 9
  class: ternaryFit (ternaryFit-class), 8
  class: ternaryFitParameters
  (ternaryFitParameters-class), 9
  class: ternaryPost (ternaryPost-class), 10
  degreeObjMin (ternaryFit-class), 8

degreeObjMin, ternaryFit-method
  (ternaryFit-class), 8
  degreeObjMin<- (ternaryFit-class), 8
  degreeObjMin<-, ternaryFit-method
  (ternaryFit-class), 8
  degreeObjs (ternaryPost-class), 10
  degreeObjs, ternaryPost-method
  (ternaryPost-class), 10
  degreeObjs<-, ternaryPost-method
  (ternaryPost-class), 10
  delta (ternaryFitParameters-class), 9
  delta, ternaryFitParameters-method
  (ternaryFitParameters-class), 9
  delta<-, ternaryFitParameters-method
  (ternaryFitParameters-class), 9
  dim, ternaryFit-method
  (ternaryFit-class), 8
  dim, ternaryPost-method
  (ternaryPost-class), 10

degreeObjs (ternaryPost-class), 10
degreeObjs<- (ternaryPost-class), 10
degreeObjs<-, ternaryPost-method
(ternaryPost-class), 10
delta (ternaryFitParameters-class), 9
delta<-, ternaryFitParameters-method
(ternaryFitParameters-class), 9
dim, ternaryFit-method
(ternaryFit-class), 8
dim, ternaryPost-method
(ternaryPost-class), 10

degreeObjs<- (ternaryPost-class), 10
degreeObjs<-, ternaryPost-method
(ternaryPost-class), 10
delta (ternaryFitParameters-class), 9
delta<-, ternaryFitParameters-method
(ternaryFitParameters-class), 9
dim, ternaryFit-method
(ternaryFit-class), 8
dim, ternaryPost-method
(ternaryPost-class), 10
experimentNames<-,ternaryFit-method
  (ternaryFit-class), 8
experimentNames<-,ternaryPost-method
  (ternaryPost-class), 10
finalTemperature (ternaryFit-class), 8
finalTemperature,ternaryFit-method
  (ternaryFit-class), 8
finalTemperature<- (ternaryFit-class), 8
finalTemperature<-,ternaryFit-method
  (ternaryFit-class), 8
geneNames (ternaryFit-class), 8
geneNames,ternaryFit-method
  (ternaryFit-class), 8
geneNames<-,ternaryPost-method
  (ternaryPost-class), 10
geneNames<-, (ternaryFit-class), 8
geneNames<-,ternaryFit-method
  (ternaryFit-class), 8
geneNames<-,ternaryPost-method
  (ternaryPost-class), 10
graphObjMin (ternaryFit-class), 8
graphObjMin,ternaryFit-method
  (ternaryFit-class), 8
graphObjMin<- (ternaryFit-class), 8
graphObjMin<-,ternaryFit-method
  (ternaryFit-class), 8
graphObjs (ternaryPost-class), 10
graphObjs,ternaryPost-method
  (ternaryPost-class), 10
graphObjs<- (ternaryPost-class), 10
graphObjs<-,ternaryPost-method
  (ternaryPost-class), 10
graphPosterior, 3
initialize,ternaryFit-method
  (ternaryFit-class), 8
initialize,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
initialize,ternaryPost-method
  (ternaryPost-class), 10
inputParams (ternaryFit-class), 8
inputParams,ternaryFit-method
  (ternaryFit-class), 8
inputParams,ternaryPost-method
  (ternaryPost-class), 10
inputParams<- (ternaryFit-class), 8
inputParams<-,ternaryFit-method
  (ternaryFit-class), 8
inputParams<-,ternaryPost-method
  (ternaryPost-class), 10
m0,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
m0<-(ternaryFitParameters-class), 9
m0<-,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
maxDegree (ternaryFitParameters-class), 9
maxDegree,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
maxDegree<-
  (ternaryFitParameters-class), 9
maxDegree<-,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
maxStage (ternaryFitParameters-class), 9
maxStage,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
maxStage<-
  (ternaryFitParameters-class), 9
maxStage<-,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
maxTransition
  (ternaryFitParameters-class), 9
maxTransition,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
maxTransition<-
  (ternaryFitParameters-class), 9
maxTransition<-,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
minScore (ternaryFit-class), 8
minScore,ternaryFit-method
  (ternaryFit-class), 8
minScore<-
  (ternaryFit-class), 8
minScore<-,ternaryFit-method
  (ternaryFit-class), 8
ne (ternaryFitParameters-class), 9
ne,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
ne<-(ternaryFitParameters-class), 9
ne<-,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
neighborDegree
  (ternaryFitParameters-class), 9
neighborDegree,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
neighborDegree<-
  (ternaryFitParameters-class), 9
neighborDegree<-,ternaryFitParameters-method
  (ternaryFitParameters-class), 9
newScore (ternaryFit-class), 8
newScore,ternaryFit-method
  (ternaryFit-class), 8
newScore<-(ternaryFit-class), 8
newScore<-,ternaryFit-method
   (ternaryFit-class), 8
pAddParent
   (ternaryFitParameters-class), 9
pAddParent,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
pAddParent<-
   (ternaryFitParameters-class), 9
perturbationObj
   (ternaryFit-class), 8
perturbationObj,ternaryFit-method
   (ternaryFit-class), 8
perturbationObj
   (ternaryPost-class), 10
perturbationObj<-
   (ternaryPost-class), 10
perturbationObj<-,ternaryFit-method
   (ternaryFit-class), 8
perturbationObj<-,ternaryPost-method
   (ternaryPost-class), 10
perturbationType
   (ternaryFitParameters-class), 9
perturbationType,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
perturbationType<-\n   (ternaryFitParameters-class), 9
pExchangeParent
   (ternaryFitParameters-class), 9
pExchangeParent,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
pExchangeParent<-
   (ternaryFitParameters-class), 9
pExchangeParent<-,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
plotFit, 3
plotPost, 4
plotTraces, 5
pNeighborhood
   (ternaryFitParameters-class), 9
pNeighborhood,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
pNeighborhood<-\n   (ternaryFitParameters-class), 9
pNeighborhood<-,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
predictAttractor, 6
r<\h (ternaryFitParameters-class), 9
r<\h,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
rho<-
   (ternaryFitParameters-class), 9
rho<-,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
scores(ternaryPost-class), 10
scores,ternaryPost-method
   (ternaryPost-class), 10
scores<-
   (ternaryPost-class), 10
scores<-,ternaryPost-method
   (ternaryPost-class), 10
scoreType(ternaryFitParameters-class), 9
scoreType,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
scoreType<-
   (ternaryFitParameters-class), 9
scoreType<-,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
show,ternaryFit-method
   (ternaryFit-class), 8
show,ternaryFitParameters-method
   (ternaryFitParameters-class), 9
show,ternaryPost-method
   (ternaryPost-class), 10
simulateSteadyState, 7
stageCount(ternaryFit-class), 8
stageCount,ternaryFit-method
   (ternaryFit-class), 8
stageCount<-
   (ternaryFit-class), 8
stageCount<-,ternaryFit-method
   (ternaryFit-class), 8
steadyStateObj
   (ternaryFit-class), 8
steadyStateObj,ternaryFit-method
   (ternaryFit-class), 8
steadyStateObj,ternaryPost-method
   (ternaryPost-class), 10
steadyStateObj<-
   (ternaryFit-class), 8
steadyStateObj<-,ternaryFit-method
   (ternaryFit-class), 8
steadyStateObj<-,ternaryPost-method
   (ternaryPost-class), 10
tableObjMin
   (ternaryFit-class), 8
tableObjMin,ternaryFit-method
   (ternaryFit-class), 8
tableObjMin<-
   (ternaryFit-class), 8
tableObjMin<-,ternaryFit-method
   (ternaryFit-class), 8
tableObjs(ternaryPost-class), 10
tableObjs,ternaryPost-method
   (ternaryPost-class), 10
tableObjs<-
   (ternaryPost-class), 10
tableObjs<-,ternaryPost-method
  (ternaryPost-class), 10
ternaryFit (ternaryFit-class), 8
ternaryFit-class, 8
ternaryFit-methods (ternaryFit-class), 8
ternaryFitParameters
  (ternaryFitParameters-class), 9
ternaryFitParameters-class, 9
ternaryFitParameters-methods
  (ternaryFitParameters-class), 9
ternaryPost (ternaryPost-class), 10
ternaryPost-class, 10
ternaryPost-methods
  (ternaryPost-class), 10
tnetfit, 11
tnetpost, 12
traces (ternaryFit-class), 8
traces, ternaryFit-method
  (ternaryFit-class), 8
traces<-, (ternaryFit-class), 8
traces<-, ternaryFit-method
  (ternaryFit-class), 8
xSeed (ternaryFit-class), 8
xSeed, ternaryFit-method
  (ternaryFit-class), 8
xSeed<-, (ternaryFit-class), 8
xSeed<-, ternaryFit-method
  (ternaryFit-class), 8