

Package ‘GeneNet’

February 14, 2012

Version 1.2.5

Date 2012-01-22

Title Modeling and Inferring Gene Networks

Author Juliane Schaefer, Rainer Opgen-Rhein, and Korbinian Strimmer.

Maintainer Korbinian Strimmer <strimmer@uni-leipzig.de>

Depends R (>= 2.10.0), corpcor (>= 1.6.2), longitudinal (>= 1.1.7), fdrtool (>= 1.2.8)

Suggests graph (>= 1.4.0), Rgraphviz (>= 1.4.0)

Description GeneNet is a package for analyzing gene expression (time series) data with focus on the inference of gene networks. In particular, GeneNet implements the methods of Schaefer and Strimmer (2005a,b,c) and Opgen-Rhein and Strimmer (2006, 2007) for learning large-scale gene association networks (including assignment of putative directions).

License GPL (>= 3)

URL <http://strimmerlab.org/software/genenet/>

Repository CRAN

Date/Publication 2012-01-22 12:26:37

R topics documented:

GeneNet-package	2
arth800	2
cor0.test	3
ecoli	4
ggm.estimate.pcor	6
ggm.make.graph	7
ggm.simulate.data	9
ggm.simulate.pcor	11
ggm.test.edges	12
kappa2n	15
z.transform	16

Index**18**

GeneNet-package *The GeneNet package*

Description

GeneNet is a package for analyzing gene expression (time series) data with focus on the inference of gene networks. In particular, GeneNet implements the methods of Sch" afer and Strimmer (2005a,b,c) and Opgen-Rhein and Strimmer (2006, 2007) for learning large-scale gene association networks (including assignment of putative directions).

Author(s)

Juliane Sch" afer, Rainer Opgen-Rhein, and Korbinian Strimmer (<http://strimmerlab.org/>)

References

See website: <http://strimmerlab.org/software/GeneNet/>

See Also

[ggm.estimate.pcor](#), [network.test.edges](#), [extract.network](#), [network.make.dot](#).

arth800 *Time Series Expression Data for 800 Arabidopsis Thaliana Genes*

Description

This data set describes the temporal expression of 800 genes of *A. thaliana* during the diurnal cycle. The 800 genes are a subset of the data presented in Smith et al. (2004) and were selected for periodicity according to the method implemented in the R package GeneCycle (<http://cran.r-project.org/web/packages/GeneCycle/>).

Usage

```
data(arth800)
```

Format

`arth800.expr` is a [longitudinal](#) object with repetitions, and contains the log₂ transformed expression data.

`arth800.mexpr` is a [longitudinal](#) object, and contains the mean expression levels of `arth800.expr`.

`arth800.descr`, `arth800.name`, `arth800.probe`, `arth800.symbol` are vectors containing additional information about each gene.

Source

The microarray experiments were performed in the laboratory of S. Smith (Edinburgh). The data are available from the NASCArrays database (<http://affymetrix.arabidopsis.info/> under experiment reference number NASCARRAYS-60).

References

Smith et al. 2004. Diurnal changes in the transcriptome encoding enzymes of starch metabolism provide evidence for both transcriptional and posttranscriptional regulation of starch metabolism in Arabidopsis leaves. *Plant Physiol.* 136: 2687-2699

Examples

```
# load GeneNet library
library("GeneNet")

# load data set
data(arth800)

is.longitudinal(arth800.expr)
summary(arth800.expr)

# plot first nine time series
plot(arth800.expr, 1:9)
```

cor0.test

Test of Vanishing (Partial) Correlation

Description

cor0.test computes a p-value for the two-sided test with the null hypothesis $H_0: \rho == 0$ versus the alternative hypothesis $H_A: \rho != 0$.

If method="student" is selected then the statistic $t=r*\sqrt{(\text{kappa}-1)/(1-r*r)}$ is considered which under H_0 is student-t distributed with $df=\text{kappa}-1$. This method is exact.

If method="dcor0" is selected then the p-value is computed directly from the distribution function [pcor0](#). This method is also exact.

If method="ztransform" is selected then the p-value is computed using the z-transform (see [z.transform](#)), i.e. using a suitable chosen normal distribution. This method returns approximate p-values.

Usage

```
cor0.test(r, kappa, method=c("student", "dcor0", "ztransform"))
```

Arguments

r	observed correlation
kappa	degree of freedom of the null-distribution
method	method used to compute the p-value

Value

A p-value.

Author(s)

Juliane Schöfer and Korbinian Strimmer (<http://strimmerlab.org>).

See Also

[dcor0](#), [kappa2n](#), [z.transform](#).

Examples

```
# load GeneNet library
library("GeneNet")

# covariance matrix
m.cov <- rbind(
  c(3,1,1,0),
  c(1,3,0,1),
  c(1,0,2,0),
  c(0,1,0,2)
)

# compute partial correlations
m.pcor <- cor2pcor(m.cov)
m.pcor

# corresponding p-values
# assuming a sample size of 25, i.e. kappa=22
kappa2n(22, 4)
cor0.test(m.pcor, kappa=22)
cor0.test(m.pcor, kappa=22) < 0.05

# p-values become smaller with larger r
cor0.test(0.7, 12)
cor0.test(0.8, 12)
cor0.test(0.9, 12)

# comparison of various methods
cor0.test(0.2, 45, method="student")
cor0.test(0.2, 45, method="dcor0")
cor0.test(0.2, 45, method="ztransform")
```

Description

This data set describes the temporal expression of 102 genes of *E. Coli* after induction of the expression of SOD (recombinant human superoxide dismutase).

Usage

```
data(ecoli)
```

Format

caulobacter is a `longitudinal` object containing the data from the Schmidt-Heck et al. (2004) experiment. Essentially, this is a matrix with 102 columns (=genes) and 9 rows (=time points). All expression levels are given in log₂-ratios with respect to the first time point (i.e. the induction at time 0).

Source

The microarray experiment was performed at the Institute of Applied Microbiology, University of Agricultural Sciences of Vienne. The data and the experiment is described in Schmidt-Heck et al. (2004).

References

Schmidt-Heck, W., Guthke, R., Toepfer, S., Reischer, H., Duerrschmid, K., and Bayer, K. (2004) Reverse engineering of the stress response during expression of a recombinant protein. In: *Proceedings of the EUNITE 2004 European Symposium on Intelligent Technologies, Hybrid Systems and their Implementation on Smart Adaptive Systems, June 10-12, 2004, Aachen, Germany*, Verlag Mainz, Wissenschaftsverlag, Aachen, 2004, 407-441 (ISBN 3-86130-368-X).

Examples

```
# load GeneNet library
library("GeneNet")

# load data set
data(ecoli)
is.longitudinal(ecoli)

# how many samples and how many genes?
dim(ecoli)
summary(ecoli)
get.time.repeats(ecoli)

# plot first nine time series
plot(ecoli, 1:9)
```

ggm.estimate.pcor *Graphical Gaussian Models: Small Sample Estimation of Partial Correlation*

Description

ggm.estimate.pcor offers an interface to two related shrinkage estimators of partial correlation. Both are fast, statistically efficient, and can be used for analyzing small sample data.

The default method "statics" employs the function `pcor.shrink` whereas the "dynamic" method relies on `dyn.pcor`. The difference between the two estimators is that the latter takes the spacings between time points into account if the input are multiple time course data (these must be provided as `longitudinal` object).

Usage

```
ggm.estimate.pcor(x, method = c("static", "dynamic"), ...)
```

Arguments

x	data matrix (each rows corresponds to one multivariate observation)
method	method used to estimate the partial correlation matrix. Available options are "static" (the default) and "dynamic" - both are shrinkage methods.
...	options passed to <code>pcor.shrink</code> and to <code>dyn.pcor</code> .

Details

For details of the shrinkage estimators we refer to Opgen-Rhein and Strimmer (2006a,b) and Sch" afer and Strimmer (2005), as well as to the manual pages of `pcor.shrink` and `dyn.pcor`.

Previously, this function offered several furthers options. The old option called "shrinkage" corresponds to the present "static" option. The other old options "observed.pcor", "partial.bagged.cor", and "bagged.pcor" are now considered obsolete and have been removed.

Value

An estimated partial correlation matrix.

Author(s)

Rainer Opgen-Rhein, Juliane Sch" afer, and Korbinian Strimmer (<http://strimmerlab.org>).

References

Opgen-Rhein, R., and K. Strimmer. 2006a. Inferring gene dependency networks from genomic longitudinal data: a functional data approach. *REVSTAT* 4:53-65. (<http://http://www.ine.pt/revstat/>)

Opgen-Rhein, R., and K. Strimmer. 2006b. Using regularized dynamic correlation to infer gene dependency networks from time-series microarray data. The 4th International Workshop on Computational Systems Biology, WCSB 2006 (June 12-13, 2006, Tampere, Finland). (<http://www.cs.tut.fi/wcsb06/>)

Schöfer, J., and Strimmer, K. (2005). A shrinkage approach to large-scale covariance estimation and implications for functional genomics. *Statist. Appl. Genet. Mol. Biol.* **4**:32. (<http://www.bepress.com/sagmb/vol4/iss1/art32/>)

See Also

[ggm.simulate.data](#), [ggm.estimate.pcor](#), [pcor.shrink](#), and [dyn.pcor](#).

Examples

```
## Not run:

# load GeneNet library
library("GeneNet")

# generate random network with 40 nodes
# it contains 780=40*39/2 edges of which 5 percent (=39) are non-zero
true.pcor <- ggm.simulate.pcor(40)

# simulate data set with 40 observations
m.sim <- ggm.simulate.data(40, true.pcor)

# simple estimate of partial correlations
estimated.pcor <- cor2pcor( cor(m.sim) )

# comparison of estimated and true values
sum((true.pcor-estimated.pcor)^2)

# a slightly better estimate ...
estimated.pcor.2 <- ggm.estimate.pcor(m.sim)
sum((true.pcor-estimated.pcor.2)^2)

## End(Not run)
```

Description

`ggm.make.dot` converts an edge list as obtained by [ggm.test.edges](#) into a "dot" file that can directly be used for plotting the network with `graphviz`.

`network.make.dot` is just an alias to `ggm.make.dot`.

`ggm.make.graph` converts an edge list as obtained by [ggm.test.edges](#) into a graph object.

`show.edge.weights` summarizes a graph object by prints a vector of weights for all edges contained in a graph. This function is convenient to gain a first impression of the graph (in particular if the "Rgraphviz" library is not installed).

Usage

```
ggm.make.dot(filename, edge.list, node.labels, main=NULL, show.edge.labels=FALSE)
network.make.dot(filename, edge.list, node.labels, main=NULL, show.edge.labels=FALSE)
ggm.make.graph(edge.list, node.labels, drop.singles=FALSE)
show.edge.weights(gr)
```

Arguments

<code>filename</code>	name of file containing the "dot" commands for graphviz
<code>edge.list</code>	a data frame, as obtained by <code>ggm.test.edges</code> , listing all edges to be included in the graph
<code>node.labels</code>	a vector with labels for each node (will be converted to type character)
<code>main</code>	title included in plot
<code>show.edge.labels</code>	plot correlation values as edge labels (default: FALSE)
<code>drop.singles</code>	remove unconnected nodes
<code>gr</code>	a graph object
<code>...</code>	options passed to plot functions

Details

For network plotting the software "graphviz" is employed (<http://www.graphviz.org>).

For the functions `ggm.plot.graph` and `ggm.make.graph` the "graph" and "Rgraphviz" infrastructure from the Bioconductor project (<http://www.bioconductor.org>) is required.

Value

`ggm.make.dot` produces a "dot" network description file that can directly be fed into graphviz in order to produce a plot of a network.

`ggm.make.graph` returns a graph object, suitable for plotting with functions from the "Rgraphviz" library.

`show.edge.weights` returns a vector of weights for all edges contained in a graph.

Author(s)

Juliane Schläfer, Rainer Opgen-Rhein, and Korbinian Strimmer (<http://strimmerlab.org>).

See Also

`ggm.test.edges`, `plot.graph`.

Examples

```

# load GeneNet library
library("GeneNet")

# generate random network with 20 nodes and 10 percent edges (=19 edges)
true.pcor <- ggm.simulate.pcor(20, 0.1)

# convert to edge list
test.results <- ggm.list.edges(true.pcor)[1:19,]

##### use graphviz directly to produce a plot #####

# uncomment for actual use!

# nlab <- LETTERS[1:20]
# ggm.make.dot(filename="test.dot", test.results, nlab, main = "A graph")
# system("fdp -T svg -o test.svg test.dot") # SVG format

##### use Rgraphviz produce a plot #####

# uncomment for actual use!

# nlab <- LETTERS[1:20]
# gr <- ggm.make.graph( test.results, nlab)
# gr
# show.edge.weights(gr)
# gr2 <- ggm.make.graph( test.results, nlab, drop.singles=TRUE)
# gr2

# plot network
# NOTE: this requires the installation of the "Rgraphviz" library
# library("Rgraphviz")
# plot(gr)

```

ggm.simulate.data

Graphical Gaussian Models: Simulation of Data

Description

ggm.simulate.data takes a positive definite partial correlation matrix and generates an i.i.d. sample from the corresponding standard multinormal distribution (with mean 0 and variance 1). If the input matrix pcor is not positive definite an error is thrown.

Usage

```
ggm.simulate.data(sample.size, pcor)
```

Arguments

sample.size sample size of simulated data set
pcor partial correlation matrix

Value

A multinormal data matrix.

Author(s)

Juliane Sch" afer and Korbinian Strimmer (<http://strimmerlab.org>).

References

Sch" afer, J., and Strimmer, K. (2005). An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics* **21**:754-764.

See Also

[ggm.simulate.pcor](#), [ggm.estimate.pcor](#).

Examples

```
# load GeneNet library
library("GeneNet")

# generate random network with 40 nodes
# it contains 780=40*39/2 edges of which 5 percent (=39) are non-zero
true.pcor <- ggm.simulate.pcor(40)

# simulate data set with 40 observations
m.sim <- ggm.simulate.data(40, true.pcor)

# simple estimate of partial correlations
estimated.pcor <- cor2pcor( cor(m.sim) )

# comparison of estimated and true values
sum((true.pcor-estimated.pcor)^2)

# a slightly better estimate ...
estimated.pcor.2 <- ggm.estimate.pcor(m.sim)
sum((true.pcor-estimated.pcor.2)^2)
```

Description

`ggm.simulate.pcor` generates a random matrix of partial correlations that corresponds to a GGM network of a given size (`num.nodes`) with a specified fraction of non-zero edges.

Usage

```
ggm.simulate.pcor(num.nodes, etaA=0.05)
```

Arguments

<code>num.nodes</code>	number of nodes in the network
<code>etaA</code>	fraction of edges with non-zero partial correlation (default: 0.05)

Details

The output of `ggm.simulate.pcor` is always positive definite. This is ensured by using diagonally dominant matrices when generating the random GGM model. For the full algorithm see Schöfer and Strimmer (2005).

Value

A positive definite partial correlation matrix.

Author(s)

Juliane Schöfer and Korbinian Strimmer (<http://strimmerlab.org>).

References

Schöfer, J., and Strimmer, K. (2005). An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics* **21**:754-764.

See Also

[ggm.simulate.data.ggm.estimate.pcor](#).

Examples

```
## Not run:  
  
# load GeneNet library  
library("GeneNet")  
  
# generate random network with 40 nodes
```

```

# it contains 780=40*39/2 edges of which 5 percent (=39) are non-zero
true.pcor <- ggm.simulate.pcor(40)

# simulate data set with 40 observations
m.sim <- ggm.simulate.data(40, true.pcor)

# simple estimate of partial correlations
estimated.pcor <- cor2pcor( cor(m.sim) )

# comparison of estimated and true values
sum((true.pcor-estimated.pcor)^2)

# a slightly better estimate ...
estimated.pcor.2 <- ggm.estimate.pcor(m.sim)
sum((true.pcor-estimated.pcor.2)^2)

## End(Not run)

```

ggm.test.edges	<i>Graphical Gaussian Models: Assess Significance of Edges (and Directions)</i>
----------------	---

Description

ggm.test.edges returns a data frame containing all edges listed in order of the magnitude of the partial correlation associated with each edge. If fdr=TRUE then in addition the p-values, q-values and posterior probabilities (=1 - local fdr) for each potential edge are computed.

network.test.edges is the same function as ggm.test.edges.

extract.network returns a data frame with a subset of significant edges.

Usage

```

ggm.test.edges(r.mat, fdr=TRUE, direct=FALSE, plot=TRUE, ...)
network.test.edges(r.mat, fdr=TRUE, direct=FALSE, plot=TRUE, ...)
extract.network(network.all, method.ggm=c("prob", "qval", "number"),
  cutoff.ggm=0.8, method.dir=c("prob", "qval", "number", "all"),
  cutoff.dir=0.8, verbose=TRUE)

```

Arguments

r.mat	matrix of partial correlations
fdr	estimate q-values and local fdr
direct	compute additional statistics for obtaining a partially directed network
plot	plot density and distribution function and (local) fdr values
...	parameters passed on to fdrtool

network.all	list with partial correlations and fdr values for all potential edges (i.e. the output of network.test.edges)
method.ggm	determines which criterion is used to select significant partial correlations (default: prob)
cutoff.ggm	default cutoff for significant partial correlations
method.dir	determines which criterion is used to select significant directions (default: prob)
cutoff.dir	default cutoff for significant directions
verbose	print information on the number of significant edges etc.

Details

For assessing the significance of edges in the GGM a mixture model is fitted to the partial correlations using `fdrtool`. This results in (i) two-sided p-values for the test of non-zero correlation, (ii) corresponding posterior probabilities (= 1- local fdr), as well as (iii) tail area-based q-values. See Sch" afer and Strimmer (2005) for details.

For determining putatative directions on this GGM log-ratios of standardized partial variances re estimated, and subsequently the corresponding (local) fdr values are computed - see Opgen-Rhein and Strimmer (2007).

Value

`ggm.test.edges` and `network.test.edges` return sorted data frame with the following columns:

<code>pcor</code>	correlation (from <code>r.mat</code>)
<code>node1</code>	first node connected to edge
<code>node2</code>	second node connected to edge
<code>pval</code>	p-value
<code>qval</code>	q-value
<code>prob</code>	probability that edge is nonzero (= 1-local fdr)
<code>log.spvar</code>	log ratio of standardized partial variance (determines direction)
<code>pval.dir</code>	p-value (directions)
<code>qval.dir</code>	q-value (directions)
<code>prob.dir</code>	1-local fdr (directions)

Each row in the data frame corresponds to one edge, and the rows are sorted according the absolute strength of the correlation (from strongest to weakest)

`extract.network` processes the above data frame containing all potential edges, and returns a dataframe with a subset of edges. If applicable, an additional last column (11) contains additional information on the directionality of an edge.

Author(s)

Rainer Opgen-Rhein, Juliane Sch" afer, Korbinian Strimmer (<http://strimmerlab.org>).

References

Schäfer, J., and Strimmer, K. (2005). An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics* **21**:754-764.

Opgen-Rhein, R., and K. Strimmer. (2007). From correlation to causation networks: a simple approximate learning algorithm and its application to high-dimensional plant gene expression data. *BMC Syst. Biol.* **1**:37.

See Also

[cor0.test](#), [fdrtool](#), [ggm.estimate.pcor](#).

Examples

```
# load GeneNet library
library("GeneNet")

# ecoli data
data(ecoli)

# estimate partial correlation matrix
inferred.pcor <- ggm.estimate.pcor(ecoli)

# p-values, q-values and posterior probabilities for each potential edge
#
test.results <- ggm.test.edges(inferred.pcor)

# show best 20 edges (strongest correlation)
test.results[1:20,]

# extract network containing edges with prob > 0.9 (i.e. local fdr < 0.1)
net <- extract.network(test.results, cutoff.ggm=0.9)
net

# how many are significant based on FDR cutoff Q=0.05 ?
num.significant.1 <- sum(test.results$qval <= 0.05)
test.results[1:num.significant.1,]

# how many are significant based on "local fdr" cutoff (prob > 0.9) ?
num.significant.2 <- sum(test.results$prob > 0.9)
test.results[test.results$prob > 0.9,]

# parameters of the mixture distribution used to compute p-values etc.
c <- fdrtool(sm2vec(inferred.pcor), statistic="correlation")
c$param
```

kappa2n	<i>Relationship Between Sample Size and the Degree of Freedom of Correlation Distribution</i>
---------	---

Description

The function `kappa2n` returns the sample size that corresponds to a given degree of freedom `kappa`, whereas `n2kappa` converts sample size to the corresponding degree of freedom.

Usage

```
kappa2n(kappa, p=2)
n2kappa(n, p=2)
```

Arguments

<code>kappa</code>	degree of freedom
<code>p</code>	number of variables (<code>p=2</code> corresponds to simple correlation)
<code>n</code>	sample size

Details

The degree of freedom `kappa` of the sample distribution of the empirical correlation coefficient depends both on the sample size `n` and the number `p` of investigated variables, i.e. whether simple or partial correlation coefficients are being considered. For `p=2` (simple correlation coefficient) the degree of freedom equals $kappa = n - 1$, whereas for arbitrary `p` (with `p-2` variables eliminated in the partial correlation coefficient) $kappa = n - p + 1$ (see also `dcor0`).

Value

The sample size `n` corresponding to a given `kappa`, or the degree of freedom `kappa` corresponding to a given `p`.

Author(s)

Juliane Sch\"afer and Korbinian Strimmer (<http://strimmerlab.org>).

See Also

`dcor0`.

Examples

```
# load GeneNet library
library("GeneNet")

# sample sizes corresponding to kappa=7
```

```

kappa2n(7)      # simple correlation
kappa2n(7, 40) # partial correlation with p=40 variables

# degree of freedom corresponding to n=100
n2kappa(100)
n2kappa(100,40)

```

z.transform

Variance-Stabilizing Transformations of the Correlation Coefficient

Description

z.transform implements Fisher's (1921) first-order and Hotelling's (1953) second-order transformations to stabilize the distribution of the correlation coefficient. After the transformation the data follows approximately a normal distribution with constant variance (i.e. independent of the mean).

The Fisher transformation is simply $z.transform(r) = atanh(r)$.

Hotelling's transformation requires the specification of the degree of freedom kappa of the underlying distribution. This depends on the sample size n used to compute the sample correlation and whether simple or partial correlation coefficients are considered. If there are p variables, with p-2 variables eliminated, the degree of freedom is $kappa=n-p+1$. (cf. also [dcor0](#)).

Usage

```

z.transform(r)
hotelling.transform(r, kappa)

```

Arguments

r	vector of sample correlations
kappa	degrees of freedom of the distribution of the correlation coefficient

Value

The vector of transformed sample correlation coefficients.

Author(s)

Korbinian Strimmer (<http://strimmerlab.org>).

References

Fisher, R.A. (1921). On the 'probable error' of a coefficient of correlation deduced from a small sample. *Metron*, **1**, 1–32.

Hotelling, H. (1953). New light on the correlation coefficient and its transformation. *J. Roy. Statist. Soc. B*, **15**, 193–232.

See Also

[dcor0](#), [kappa2n](#).

Examples

```
# load GeneNet library
library("GeneNet")

# small example data set
r <- c(-0.26074194, 0.47251437, 0.23957283, -0.02187209, -0.07699437,
      -0.03809433, -0.06010493, 0.01334491, -0.42383367, -0.25513041)

# transformed data
z1 <- z.transform(r)
z2 <- hotelling.transform(r,7)
z1
z2
```

Index

- *Topic **datasets**
 - arth800, [2](#)
 - ecoli, [4](#)
- *Topic **hplot**
 - ggm.make.graph, [7](#)
- *Topic **htest**
 - cor0.test, [3](#)
 - ggm.estimate.pcor, [6](#)
 - ggm.test.edges, [12](#)
- *Topic **multivariate**
 - GeneNet-package, [2](#)
 - ggm.simulate.data, [9](#)
 - ggm.simulate.pcor, [11](#)
- *Topic **univar**
 - kappa2n, [15](#)
 - z.transform, [16](#)

arth800, [2](#)

cor0.test, [3](#), [14](#)

dcor0, [4](#), [15–17](#)

dyn.pcor, [6](#), [7](#)

ecoli, [4](#)

extract.network, [2](#)

extract.network (ggm.test.edges), [12](#)

fdrtool, [12–14](#)

GeneNet-package, [2](#)

ggm.estimate.pcor, [2](#), [6](#), [7](#), [10](#), [11](#), [14](#)

ggm.make.dot (ggm.make.graph), [7](#)

ggm.make.graph, [7](#)

ggm.simulate.data, [7](#), [9](#), [11](#)

ggm.simulate.pcor, [10](#), [11](#)

ggm.test.edges, [7](#), [8](#), [12](#)

hotelling.transform (z.transform), [16](#)

kappa2n, [4](#), [15](#), [17](#)

longitudinal, [2](#), [5](#), [6](#)

n2kappa (kappa2n), [15](#)

network.make.dot, [2](#)

network.make.dot (ggm.make.graph), [7](#)

network.test.edges, [2](#)

network.test.edges (ggm.test.edges), [12](#)

pcor.shrink, [6](#), [7](#)

pcor0, [3](#)

show.edge.weights (ggm.make.graph), [7](#)

z.transform, [3](#), [4](#), [16](#)