

# Package ‘ernm’

July 22, 2025

**Type** Package

**Title** Exponential-Family Random Network Models

**Version** 1.0.1

**Date** 2025-7-22

**Description** Estimation of fully and partially observed Exponential-Family Random Network Models (ERNM). Exponential-family Random Graph Models (ERGM) and Gibbs Fields are special cases of ERNMs and can also be estimated with the package. Please cite Fellows and Handcock (2012), ``Exponential-family Random Network Models" available at [doi:10.48550/arXiv.1208.0121](https://doi.org/10.48550/arXiv.1208.0121).

**License** LGPL-2.1

**Depends** R (>= 3.5.0), BH, methods, network, Rcpp

**Imports** dplyr, ggplot2, graphics, moments, rlang, stats, tidyr, trust

**Suggests** spelling, testthat

**LinkingTo** BH, Rcpp

**Copyright** 2014-2025 Ian Fellows

**Encoding** UTF-8

**Language** en-US

**LazyData** true

**LazyLoad** yes

**RcppModules** ernm

**RoxygenNote** 7.3.2

**NeedsCompilation** yes

**Author** Ian Fellows [aut],  
Duncan Clark [aut, cre]

**Maintainer** Duncan Clark <[dac6@williams.edu](mailto:dac6@williams.edu)>

**Repository** CRAN

**Date/Publication** 2025-07-22 21:22:27 UTC

## Contents

|                                         |    |
|-----------------------------------------|----|
| as.BinaryNet . . . . .                  | 3  |
| as.network.DirectedNet . . . . .        | 3  |
| as.network.Rcpp_DirectedNet . . . . .   | 4  |
| as.network.Rcpp_UndirectedNet . . . . . | 4  |
| as.network.UndirectedNet . . . . .      | 5  |
| BinaryNet . . . . .                     | 5  |
| calculateStatistics . . . . .           | 5  |
| call-symbols . . . . .                  | 6  |
| createCppModel . . . . .                | 6  |
| createCppSampler . . . . .              | 7  |
| DirectedNet-class . . . . .             | 8  |
| dutch_school . . . . .                  | 8  |
| ernm . . . . .                          | 9  |
| ernmFit . . . . .                       | 10 |
| ErnmModels . . . . .                    | 11 |
| ernmPackageSkeleton . . . . .           | 11 |
| ErnmSamplers . . . . .                  | 11 |
| ernm_gof . . . . .                      | 12 |
| extract-methods . . . . .               | 13 |
| fullErnmLikelihood . . . . .            | 13 |
| FullErnmModel . . . . .                 | 14 |
| marErnmLikelihood . . . . .             | 15 |
| mcmcEss . . . . .                       | 15 |
| mcmcse . . . . .                        | 16 |
| MissingErnmModel . . . . .              | 16 |
| plot.ernm . . . . .                     | 17 |
| plot.Rcpp_DirectedNet . . . . .         | 17 |
| plot.Rcpp_UndirectedNet . . . . .       | 18 |
| print.ernm . . . . .                    | 18 |
| registerDirectedStatistic . . . . .     | 19 |
| register_rcpp_net_methods . . . . .     | 19 |
| runErnmCppTests . . . . .               | 19 |
| samplike . . . . .                      | 20 |
| simulateStatistics . . . . .            | 21 |
| summary.ernm . . . . .                  | 22 |
| taperedErnmLikelihood . . . . .         | 22 |
| UndirectedNet-class . . . . .           | 23 |
| vcov.ernm . . . . .                     | 23 |

## Index

---

|              |                                                                             |
|--------------|-----------------------------------------------------------------------------|
| as.BinaryNet | <i>convert and network to either an UndirectedNet or DirectedNet object</i> |
|--------------|-----------------------------------------------------------------------------|

---

**Description**

convert and network to either an UndirectedNet or DirectedNet object

**Usage**

```
as.BinaryNet(x, ...)
```

**Arguments**

|     |            |
|-----|------------|
| x   | the object |
| ... | unused     |

**Value**

a BinaryNet object

---

|                        |                                     |
|------------------------|-------------------------------------|
| as.network.DirectedNet | <i>Conversion to network object</i> |
|------------------------|-------------------------------------|

---

**Description**

Conversion to network object

**Usage**

```
## S3 method for class 'DirectedNet'  
as.network(x, ...)
```

**Arguments**

|     |            |
|-----|------------|
| x   | the object |
| ... | unused     |

**Value**

a directed network object

---

```
as.network.Rcpp_DirectedNet
```

*convert and Rcpp\_DirectedNet to a network object*

---

**Description**

convert and Rcpp\_DirectedNet to a network object

**Usage**

```
## S3 method for class 'Rcpp_DirectedNet'  
as.network(x, ...)
```

**Arguments**

|     |            |
|-----|------------|
| x   | the object |
| ... | unused     |

**Value**

a directed network object

---

```
as.network.Rcpp_UndirectedNet
```

*convert and UndirectedNet to a network object*

---

**Description**

convert and UndirectedNet to a network object

**Usage**

```
## S3 method for class 'Rcpp_UndirectedNet'  
as.network(x, ...)
```

**Arguments**

|     |            |
|-----|------------|
| x   | the object |
| ... | unused     |

**Value**

a undirected network object

---

|                                     |
|-------------------------------------|
| as.network.UndirectedNet            |
| <i>Conversion to network object</i> |

---

**Description**

Conversion to network object

**Usage**

```
## S3 method for class 'UndirectedNet'  
as.network(x, ...)
```

**Arguments**

|     |            |
|-----|------------|
| x   | the object |
| ... | unused     |

**Value**

a undirected network object

---

|           |                  |
|-----------|------------------|
| BinaryNet | <i>BinaryNet</i> |
|-----------|------------------|

---

**Description**

BinaryNet

---

|                     |                                                  |
|---------------------|--------------------------------------------------|
| calculateStatistics | <i>calculate model statistics from a formula</i> |
|---------------------|--------------------------------------------------|

---

**Description**

calculate model statistics from a formula

**Usage**

```
calculateStatistics(formula)
```

**Arguments**

|         |                 |
|---------|-----------------|
| formula | An ernm formula |
|---------|-----------------|

**Value**

a list of statistics

---

`call-symbols`*Internal Symbols*

---

**Description**

Internal symbols used to access compiles code.

---

`createCppModel`*creates a model*

---

**Description**

creates a model

**Usage**

```
createCppModel(  
  formula,  
  ignoreMnar = TRUE,  
  cloneNet = TRUE,  
  theta = NULL,  
  modelArgs = list(modelClass = "Model")  
)
```

**Arguments**

|                         |                                                               |
|-------------------------|---------------------------------------------------------------|
| <code>formula</code>    | the model formula                                             |
| <code>ignoreMnar</code> | ignore missing not at random offsets                          |
| <code>cloneNet</code>   | should the network be cloned                                  |
| <code>theta</code>      | the model parameters.                                         |
| <code>modelArgs</code>  | additiional arguments for the model, e.g. tapering parameters |

**Value**

a Model object

---

|                  |                         |
|------------------|-------------------------|
| createCppSampler | <i>create a sampler</i> |
|------------------|-------------------------|

---

**Description**

create a sampler

**Usage**

```
createCppSampler(  
  formula,  
  modelArgs = list(modelClass = "Model"),  
  dyadToggle = NULL,  
  dyadArgs = list(),  
  vertexToggle = NULL,  
  vertexArgs = list(),  
  nodeSamplingPercentage = 0.2,  
  ignoreMnar = TRUE,  
  theta = NULL,  
  ...  
)
```

**Arguments**

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| formula                | the model formula                                                                                        |
| modelArgs              | additiional arguments for the model, e.g. tapering parameters                                            |
| dyadToggle             | the method of sampling to use. Defaults to alternating between nodal-tie-dyad and neighborhood toggling. |
| dyadArgs               | list of args for dyad                                                                                    |
| vertexToggle           | the method of vertex attribuate sampling to use.                                                         |
| vertexArgs             | list of args for vertex                                                                                  |
| nodeSamplingPercentage | how often the nodes should be toggled                                                                    |
| ignoreMnar             | ignore missing not at random offsets                                                                     |
| theta                  | parameter values                                                                                         |
| ...                    | additional parameters to be passed to createCppModel                                                     |

**Value**

a MetropolisHastings object

---

|                   |                          |
|-------------------|--------------------------|
| DirectedNet-class | <i>DirectedNet Class</i> |
|-------------------|--------------------------|

---

### Description

An S4 (old-style) class representing a directed network.

---

|              |                          |
|--------------|--------------------------|
| dutch_school | <i>Dutch School Data</i> |
|--------------|--------------------------|

---

### Description

This dataset contains network and actor attributes collected in early adolescence. It is provided by Andrea Knecht and stored in the package.

### Usage

```
data(dutch_school)

data("dutch_school")
```

### Format

An object of class `list` of length 4.

### Data taken from https

[//www.stats.ox.ac.uk/~snijders/siena/tutorial2010\\_data.htm](http://www.stats.ox.ac.uk/~snijders/siena/tutorial2010_data.htm). Processed as undirected networks.

### Source

Knecht, A. (2004). \*Network and actor attributes in early adolescence\*. DANS Data Station Social Sciences and Humanities. DOI: [doi:10.17026/dansz9bh2bp](https://doi.org/10.17026/dansz9bh2bp).

### References

Snijders, T.A.B., Steglich, C.E.G., and van de Bunt, G.G. (2010), Introduction to actor-based models for network dynamics, *Social Networks* 32, 44-60, <http://dx.doi.org/10.1016/j.socnet.2009.02.004>.



---

|      |                           |
|------|---------------------------|
| ernm | <i>fits an ERNM model</i> |
|------|---------------------------|

---

## Description

fits an ERNM model

## Usage

```
ernm(
  formula,
  tapered = TRUE,
  tapering_r = 3,
  modelArgs = list(),
  nodeSamplingPercentage = 0.2,
  modelType = NULL,
  likelihoodArgs = list(),
  fullToggles = c("Compound_NodeTieDyad_Neighborhood", "DefaultVertex"),
  missingToggles = c("Compound_NodeTieDyadMissing_NeighborhoodMissing", "VertexMissing"),
  ...
)
```

## Arguments

|                        |                                                                                                              |
|------------------------|--------------------------------------------------------------------------------------------------------------|
| formula                | model formula                                                                                                |
| tapered                | should the model be tapered                                                                                  |
| tapering_r             | the tapering parameter ( $\tau = 1/(\text{tapering\_r}^2 + 5)$ )                                             |
| modelArgs              | additiional arguments for the model, e.g. tapering parameters that override the defaults                     |
| nodeSamplingPercentage | how often are nodal variates toggled                                                                         |
| modelType              | either FullErnmModel or MissingErnmModel if NULL will check for missingness                                  |
| likelihoodArgs         | additiional arguments for the <code>ernmLikelihood</code>                                                    |
| fullToggles            | a character vector of length 2 indicating the dyad and vertex toggle types for the unconditional simulations |
| missingToggles         | a character vector of length 2 indicating the dyad and vertex toggle types for the conditional simulations   |
| ...                    | additional parameters for <code>ernmFit</code>                                                               |

## Value

a fitted model

---

|         |                          |
|---------|--------------------------|
| ernmFit | <i>fit an ernm model</i> |
|---------|--------------------------|

---

**Description**

fit an ernm model

**Usage**

```
ernmFit(
  sampler,
  theta0,
  mcmcBurnIn = 10000,
  mcmcInterval = 100,
  mcmcSampleSize = 10000,
  minIter = 3,
  maxIter = 40,
  objectiveTolerance = 0.5,
  gradTolerance = 0.25,
  meanStats,
  verbose = 1,
  method = c("bounded", "newton")
)
```

**Arguments**

|                    |                                                        |
|--------------------|--------------------------------------------------------|
| sampler            | the ErnmModel                                          |
| theta0             | initial starting values                                |
| mcmcBurnIn         | burn in                                                |
| mcmcInterval       | interval                                               |
| mcmcSampleSize     | sample size                                            |
| minIter            | minimum number of iterations                           |
| maxIter            | maximum number of iterations                           |
| objectiveTolerance | convergence criteria on change in log likelihood ratio |
| gradTolerance      | convergence criteria on scaled gradient                |
| meanStats          | if non-missing, these are the target statistics        |
| verbose            | level of verbosity 0, 1, or 2                          |
| method             | the optimization method to use                         |

**Value**

ernm object

---

ErnmModels

*Models*


---

## Description

Models

---

ernmPackageSkeleton

*Create an ERNM Package Skeleton*


---

## Description

Creates a skeleton for a package extending the ernm package by copying an example package.

## Usage

```
ernmPackageSkeleton(path = ".")
```

## Arguments

|      |                                                                                         |
|------|-----------------------------------------------------------------------------------------|
| path | A character string specifying the directory where the package skeleton will be created. |
|------|-----------------------------------------------------------------------------------------|

## Value

A logical value indicating whether the copy was successful.

---

ErnmSamplers

*Metropolis Samplers*


---

## Description

Metropolis Samplers

---

|          |              |
|----------|--------------|
| ernm_gof | <i>print</i> |
|----------|--------------|

---

## Description

Goodness of fit plot for ERNM models, particularly suited for comparing models

## Usage

```
ernm_gof(  
  models,  
  observed_network = NULL,  
  stats_formula,  
  style = "histogram",  
  scales = "fixed",  
  print = TRUE,  
  n_sim = 10000,  
  burnin = 10000,  
  interval = 100  
)
```

## Arguments

|                  |                                                            |
|------------------|------------------------------------------------------------|
| models           | named list of ernm models to be compared (can be length 1) |
| observed_network | the observed network                                       |
| stats_formula    | the formula for the statistics                             |
| style            | the style of the plot, either 'histogram' or 'boxplot'     |
| scales           | the scales of the plot, either 'fixed' or 'free'           |
| print            | whether to print the plot                                  |
| n_sim            | the number of simulations to run                           |
| burnin           | the burnin for the MCMC simulation                         |
| interval         | the sampling interval for MCMC simulation                  |

## Value

A list containing goodness-of-fit plots and simulated statistics

---

|                 |                                                  |
|-----------------|--------------------------------------------------|
| extract-methods | <i>Subsetting and assignment for Net objects</i> |
|-----------------|--------------------------------------------------|

---

**Description**

These methods allow standard subsetting ('[') and assignment ('[<-') for 'DirectedNet' and 'UndirectedNet' objects.

**Usage**

```
## S4 method for signature 'Rcpp_DirectedNet'
x[i, j, ..., maskMissing = TRUE, drop = TRUE]

## S4 method for signature 'Rcpp_UndirectedNet'
x[i, j, ..., maskMissing = TRUE, drop = TRUE]

## S4 replacement method for signature 'Rcpp_DirectedNet'
x[i, j, ...] <- value

## S4 replacement method for signature 'Rcpp_UndirectedNet'
x[i, j, ...] <- value
```

**Arguments**

|             |                                                 |
|-------------|-------------------------------------------------|
| x           | A 'DirectedNet' or 'UndirectedNet' object.      |
| i, j        | Index vectors.                                  |
| ...         | Currently unused.                               |
| maskMissing | Logical. Should missing values be masked by NA? |
| drop        | Ignored (present for compatibility).            |
| value       | Values to assign (for '[' only).                |

**Value**

A modified object or extracted submatrix depending on the method.

---

|                    |                                             |
|--------------------|---------------------------------------------|
| fullErnmLikelihood | <i>likelihood for a fully observed ernm</i> |
|--------------------|---------------------------------------------|

---

**Description**

likelihood for a fully observed ernm

**Usage**

```
fullErnmLikelihood(
  theta,
  sample,
  theta0,
  stats,
  minEss = 5,
  damping = 0.05,
  method = c("cumulant", "sample"),
  order = 3
)
```

**Arguments**

|         |                                            |
|---------|--------------------------------------------|
| theta   | parameters                                 |
| sample  | mcmc sample                                |
| theta0  | parameter values which generated sample    |
| stats   | observed statistics                        |
| minEss  | minimum effective sample size              |
| damping | a damping parameter                        |
| method  | cumulant generating function approximation |
| order   | the ordering                               |

**Value**

a list with value, gradient, and hessian

---

|               |                                         |
|---------------|-----------------------------------------|
| FullErnmModel | <i>creates an ERNM likelihood model</i> |
|---------------|-----------------------------------------|

---

**Description**

creates an ERNM likelihood model

**Usage**

```
FullErnmModel(sampler, logLik, ...)
```

**Arguments**

|         |                                              |
|---------|----------------------------------------------|
| sampler | a sampler                                    |
| logLik  | a log likelihood function (optional)         |
| ...     | additional parameters for the log likelihood |

**Value**

a FullyObservedModel object

---

|                   |                                                 |
|-------------------|-------------------------------------------------|
| marErnmLikelihood | <i>likelihood for an ernm with missing data</i> |
|-------------------|-------------------------------------------------|

---

**Description**

likelihood for an ernm with missing data

**Usage**

```
marErnmLikelihood(theta, sample, theta0, stats, minEss = 5, damping = 0.1)
```

**Arguments**

|         |                                         |
|---------|-----------------------------------------|
| theta   | parameters                              |
| sample  | mcmc sample                             |
| theta0  | parameter values which generated sample |
| stats   | observed statistics                     |
| minEss  | minimum effective sample size           |
| damping | a damping parameter                     |

**Value**

a list with value, gradient, and hessian

---

|         |                                   |
|---------|-----------------------------------|
| mcmcEss | <i>MCMC Effective Sample Size</i> |
|---------|-----------------------------------|

---

**Description**

Computes the effective sample size from a statistic vector.

**Usage**

```
mcmcEss(x)
```

**Arguments**

|   |                   |
|---|-------------------|
| x | A numeric vector. |
|---|-------------------|

**Value**

A numeric value representing the effective sample size.

## References

Kass, R. E., Carlin, B. P., Gelman, A., & Neal, R. M. (1998). "Markov Chain Monte Carlo in Practice: A Roundtable Discussion." *The American Statistician*, 52(2), 93-100. DOI: [doi:10.2307/2685466](https://doi.org/10.2307/2685466)

---

mcmcse

*MCMC Standard Error by Batch*


---

## Description

Computes the MCMC standard error from a statistic vector using a batching method.

## Usage

```
mcmcse(x, expon = 0.5)
```

## Arguments

**x** A numeric vector of statistics.  
**expon** A numeric value controlling the batch size; default is 0.5.

## Value

A numeric value representing the estimated standard error.

---

MissingErnmModel

*creates an ERNM likelihood model*


---

## Description

creates an ERNM likelihood model

## Usage

```
MissingErnmModel(observedSampler, unobservedSampler, ...)
```

## Arguments

**observedSampler** a sampler  
**unobservedSampler** a sampler conditional upon the observed values  
**...** additional parameters for the log likelihood

## Value

a MarModel object



---

|           |                            |
|-----------|----------------------------|
| plot.ernm | <i>plot an ernm object</i> |
|-----------|----------------------------|

---

**Description**

plot an ernm object

**Usage**

```
## S3 method for class 'ernm'  
plot(x, ...)
```

**Arguments**

|     |            |
|-----|------------|
| x   | the object |
| ... | unused     |

**Value**

No return value, plots the likelihood history

---

|                       |                                         |
|-----------------------|-----------------------------------------|
| plot.Rcpp_DirectedNet | <i>plot an Rcpp_UnirectedNet object</i> |
|-----------------------|-----------------------------------------|

---

**Description**

plot an Rcpp\_UnirectedNet object

**Usage**

```
## S3 method for class 'Rcpp_DirectedNet'  
plot(x, ...)
```

**Arguments**

|     |                                        |
|-----|----------------------------------------|
| x   | the object                             |
| ... | additional parameters for plot.network |

**Value**

No return value, invisibly NULL

---

```
plot.Rcpp_UndirectedNet
    plot an UndirectedNet object
```

---

**Description**

plot an UndirectedNet object

**Usage**

```
## S3 method for class 'Rcpp_UndirectedNet'
plot(x, ...)
```

**Arguments**

|     |                                        |
|-----|----------------------------------------|
| x   | the object                             |
| ... | additional parameters for plot.network |

**Value**

No return value, invisibly NULL

---

```
print.ernm    print
```

---

**Description**

print

**Usage**

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

**Arguments**

|     |        |
|-----|--------|
| x   | x      |
| ... | unused |

**Value**

No return value, prints summary

---

```
registerDirectedStatistic
```

*Register Statistics*

---

**Description**

Register Statistics

**Usage**

```
registerDirectedStatistic
```

**Value**

no return value

---

```
register_rcpp_net_methods
```

*Register setter methods for Rcpp net objeccts*

---

**Description**

Register setter methods for Rcpp net objeccts

**Usage**

```
register_rcpp_net_methods()
```

**Value**

no return value

---

```
runErnmCppTests
```

*runErnmCppTests*

---

**Description**

Runs the internal C++ tests for the ernm package.

**Value**

A logical value indicating whether all tests passed.

**Examples**

```
runErnmCppTests()
```

---

samplike

*Sampson's Monks Data*

---

### Description

This dataset represents the social network of relationships among monks in a monastery, as studied by Samuel F. Sampson. The data were collected during a period of instability and document both positive and negative interactions among the monks.

### Usage

```
data(samplike)
```

```
data("samplike")
```

### Format

An object of class network of length 5.

### Details

The study recorded friendships, antagonisms, and other social relationships among the monks before and after a significant schism occurred in the monastery.

### NOTE COPIED FROM ERGM PACKAGE

Mislabeling in Versions Prior to 3.6.1: In *ergm* version 3.6.0 and earlier, the adjacency matrices of the datasets reflected an older ordering of the names.

### Source

Sampson, S. F. (1969). *\*Crisis in a cloister\**. Unpublished Ph.D. dissertation, Cornell University.

### References

White, H.C., Boorman, S.A. and Breiger, R.L. (1976). *Social structure from multiple networks. I. Blockmodels of roles and positions*. American Journal of Sociology, 81(4), 730-780.

### See Also

florentine, network, plot.network, ergm

---

|                    |                            |
|--------------------|----------------------------|
| simulateStatistics | <i>simulate statistics</i> |
|--------------------|----------------------------|

---

**Description**

simulate statistics

**Usage**

```
simulateStatistics(  
  formula,  
  theta,  
  nodeSamplingPercentage = 0.2,  
  mcmcBurnIn = 10000,  
  mcmcInterval = 100,  
  mcmcSampleSize = 100,  
  ignoreMnar = TRUE,  
  modelArgs = list(modelClass = "Model"),  
  ...  
)
```

**Arguments**

|                        |                                                               |
|------------------------|---------------------------------------------------------------|
| formula                | the model formula                                             |
| theta                  | model parameters                                              |
| nodeSamplingPercentage | how often the nodes should be toggled                         |
| mcmcBurnIn             | burn in                                                       |
| mcmcInterval           | interval                                                      |
| mcmcSampleSize         | sample size                                                   |
| ignoreMnar             | ignore missing not at random offsets                          |
| modelArgs              | additiional arguments for the model, e.g. tapering parameters |
| ...                    | additional arguments to createCppSampler                      |

**Value**

a list of statistics

---

|              |                |
|--------------|----------------|
| summary.ernm | <i>summary</i> |
|--------------|----------------|

---

**Description**

summary

**Usage**

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

**Arguments**

|        |        |
|--------|--------|
| object | object |
| ...    | unused |

**Value**

a data frame summary of the model

---

|                       |                                                |
|-----------------------|------------------------------------------------|
| taperedErnmLikelihood | $(E(g(X)) - g(x_o))^2$ for <i>TaperedModel</i> |
|-----------------------|------------------------------------------------|

---

**Description**

$(E(g(X)) - g(x_o))^2$  for *TaperedModel*

**Usage**

```
taperedErnmLikelihood(  
  theta,  
  centers,  
  tau,  
  sample,  
  theta0,  
  stats,  
  minEss = 5,  
  damping = 0.05  
)
```

Arguments

|         |                                         |
|---------|-----------------------------------------|
| theta   | parameters                              |
| centers | center of statistics                    |
| tau     | tapering parameter                      |
| sample  | mcmc sample                             |
| theta0  | parameter values which generated sample |
| stats   | observed statistics                     |
| minEss  | minimum effective sample size           |
| damping | a damping parameter                     |

Value

a list with value, gradient, and hessian

---

|                     |                            |
|---------------------|----------------------------|
| UndirectedNet-class | <i>UndirectedNet Class</i> |
|---------------------|----------------------------|

---

Description

An S4 (old-style) class representing an undirected network.

---

|           |                                    |
|-----------|------------------------------------|
| vcov.ernm | <i>parameter covariance matrix</i> |
|-----------|------------------------------------|

---

Description

parameter covariance matrix

Usage

```
## S3 method for class 'ernm'
vcov(object, ...)
```

Arguments

|        |        |
|--------|--------|
| object | object |
| ...    | unused |

Value

covariance matrix

# Index

- \* **datasets**
  - dutch\_school, 8
  - samplike, 20
- [,DirectedNet-method (extract-methods), 13
- [,Rcpp\_DirectedNet-method (extract-methods), 13
- [,Rcpp\_UndirectedNet-method (extract-methods), 13
- [,UndirectedNet-method (extract-methods), 13
- [<-,DirectedNet-method (extract-methods), 13
- [<-,Rcpp\_DirectedNet-method (extract-methods), 13
- [<-,Rcpp\_UndirectedNet-method (extract-methods), 13
- [<-,UndirectedNet-method (extract-methods), 13
- \_ernm\_initStats (call-symbols), 6
- \_ernm\_initToggles (call-symbols), 6
- \_rcpp\_module\_boot\_ernm (call-symbols), 6
- as.BinaryNet, 3
- as.network.DirectedNet, 3
- as.network.Rcpp\_DirectedNet, 4
- as.network.Rcpp\_UndirectedNet, 4
- as.network.UndirectedNet, 5
- BinaryNet, 5
- calculateStatistics, 5
- call-symbols, 6
- createCppModel, 6
- createCppSampler, 7
- DirectedCdSampler (ErnmSamplers), 11
- DirectedGibbsCdSampler (ErnmSamplers), 11
- DirectedGibbsCdSampler2 (ErnmSamplers), 11
- DirectedMetropolisHastings (ErnmSamplers), 11
- DirectedModel (ErnmModels), 11
- DirectedNet (BinaryNet), 5
- DirectedNet-class, 8
- DirectedTaperedModel (ErnmModels), 11
- dutch\_school, 8
- ernm, 9
- ernm\_gof, 12
- ernmFit, 10
- ErnmModels, 11
- ernmPackageSkeleton, 11
- ErnmSamplers, 11
- extract-methods, 13
- fullErnmLikelihood, 13
- FullErnmModel, 14
- initLatent (call-symbols), 6
- marErnmLikelihood, 15
- mcmcEss, 15
- mcmcse, 16
- MissingErnmModel, 16
- plot.ernm, 17
- plot.Rcpp\_DirectedNet, 17
- plot.Rcpp\_UndirectedNet, 18
- print.ernm, 18
- Rcpp\_DirectedCdSampler-class (ErnmSamplers), 11
- Rcpp\_DirectedGibbsCdSampler-class (ErnmSamplers), 11
- Rcpp\_DirectedGibbsCdSampler2-class (ErnmSamplers), 11
- Rcpp\_DirectedMetropolisHastings-class (ErnmSamplers), 11
- Rcpp\_DirectedModel-class (ErnmModels), 11



Rcpp\_DirectedNet-class (BinaryNet), 5  
Rcpp\_DirectedTaperedModel-class  
    (ErnmModels), 11  
Rcpp\_UndirectedCdSampler-class  
    (ErnmSamplers), 11  
Rcpp\_UndirectedGibbsCdSampler-class  
    (ErnmSamplers), 11  
Rcpp\_UndirectedGibbsCdSampler2-class  
    (ErnmSamplers), 11  
Rcpp\_UndirectedMetropolisHastings-class  
    (ErnmSamplers), 11  
Rcpp\_UndirectedModel-class  
    (ErnmModels), 11  
Rcpp\_UndirectedNet-class (BinaryNet), 5  
Rcpp\_UndirectedTaperedModel-class  
    (ErnmModels), 11  
register\_rcpp\_net\_methods, 19  
registerDirectedOffset  
    (registerDirectedStatistic), 19  
registerDirectedStatistic, 19  
registerUndirectedOffset  
    (registerDirectedStatistic), 19  
registerUndirectedStatistic  
    (registerDirectedStatistic), 19  
runErnmCppTests, 19  
  
samplike, 20  
sampsom (samplike), 20  
simulateStatistics, 21  
summary.ernm, 22  
  
taperedErnmLikelihood, 22  
  
UndirectedCdSampler (ErnmSamplers), 11  
UndirectedGibbsCdSampler  
    (ErnmSamplers), 11  
UndirectedGibbsCdSampler2  
    (ErnmSamplers), 11  
UndirectedMetropolisHastings  
    (ErnmSamplers), 11  
UndirectedModel (ErnmModels), 11  
UndirectedNet (BinaryNet), 5  
UndirectedNet-class, 23  
UndirectedTaperedModel (ErnmModels), 11  
  
vcov.ernm, 23