Package ‘CCMnet’

May 25, 2026

Type Package

Title Congruence Class Models for Networks

Version 1.1.0

Description Provides an implementation of Congruence Class Models (CCMs) for generating networks. For additional details on CCMs see Goyal, Blitzstein, and De Gruttola (2014) <doi:10.1017/nws.2014.2> and Goyal, De Gruttola, Martin, Rennert, and Onnela <doi:10.48550/arXiv.2603.02467>. 'ccmnet' facilitates sampling networks based on specific topological properties and attribute mixing patterns using a Markov Chain Monte Carlo framework. The implementation builds upon code from the 'ergm' package; see Handcock, Hunter, Butts, Goodreau, and Morris (2008) <doi:10.18637/jss.v024.i01>.

License GPL-3

Encoding UTF-8

Imports dplyr, ergm, ggplot2, gtools, igraph, intergraph, kableExtra, mvtnorm, network, RBesT, rlang, stats, tibble, tidyr, utils

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation yes

Author Ravi Goyal [aut, cre], Statnet Development Team [ctb, cph]

Maintainer Ravi Goyal <ravi.j.goyal@gmail.com>

Repository CRAN

Date/Publication 2026-05-25 15:20:13 UTC

Contents

ccm_distributions 2
ccm_properties 3
plot.ccm_sample 4
sample_ccm 5
sample_target_distr 7

Index 9

`ccm_distributions` *Probability Distributions in CCMnet*

Description

Details on the probability distributions implemented in the **CCMnet** C engine. These distributions define the "soft constraints" placed on network properties during the MCMC sampling process.

Details

By decoupling the probability distribution from the network property, **CCMnet** allows researchers to represent structural uncertainty. For example, one might target a specific degree distribution via `"dirmult"` while allowing the total edge count to follow a wide `"gamma"` distribution.

Supported Distributions

`"poisson"` Requires `list(lambda)`. Typically used for count-based statistics like `"edges"` or `"triangles"`.

`"gamma"` Requires `list(shape, rate)`. Useful for continuous or skewed properties. The implementation uses the kernel $x^{\alpha-1}e^{-\beta x}$.

`"dirmult"` Requires `list(alphas)`. A Dirichlet-Multinomial implementation optimized for proportions. In **CCMnet**, the global normalizing constant is omitted to facilitate sampling in systems where the total count (e.g., total edges) is variable.

`"normal"` Requires `list(mean, sd)`. Standard Gaussian constraint.

`"beta"` Requires `list(shape1, shape2)`. Restricted to properties bounded in the interval [0,1], such as `"density"`.

`"uniform"` A flat distribution where the MCMC explores the congruence class without preference for specific statistic values.

See Also

`ccm_properties`, `sample_ccm`

Other ccm_core: `ccm_properties`, `sample_ccm()`

`ccm_properties` *Network Properties in CCMnet*

Description

CCMnet implements a systematic hierarchy of network properties motivated by the dk-series framework (Mahadevan et al., 2006; Orsini et al., 2015). These properties can be targeted using soft constraints governed by user-specified probability distributions.

Topological-based Properties (dk-series)

The following properties follow the dk-series framework, which defines increasingly constrained random graph ensembles:

`"edges"` **(0k)** The total number of edges within the graph. This is the simplest topological constraint.

`"degreedist"` **(1k)** A vector where the j-th entry represents the number of nodes having degree j.

`"degmixing"` **(2k)** The joint degree distribution, represented as a degree mixing matrix. The $(i,j) entry represents the number of edges between nodes with degree i and degree j, capturing degree-degree correlations (assortativity).

`"degmixing"` + `"triangles"` **(2.1k)** Extends the 2k distribution by including the total number of closed triads (triangles) in the network, allowing for clustering constraints.

Covariate-based Properties

CCMnet incorporates node-level attributes by defining congruence classes based on categorical or discretized covariates (e.g., homophily or social stratification).

`"mixing"` The attribute mixing matrix. The (i,j) entry represents the number of edges between nodes belonging to covariate groups i and j. Requires `cov_pattern`.

`"degreedist+degreedist+mixing"` A combination property that includes the attribute mixing matrix alongside separate degree distributions calculated for each distinct covariate group. Currently on a binary covariate is implemented. Therefore, input is characteristics for two degree distributions and one scalar mixing value.

References

Mahadevan, P., Krioukov, D., Fomenkov, K., Huffaker, B., & Vahdat, A. (2006). Systematic topology analysis and generation using degree correlations. *ACM SIGCOMM Computer Communication Review*.

Orsini, C., et al. (2015). Quantifying randomness in real networks. *Nature Communications*.

See Also

`ccm_distributions`, `sample_ccm`

Other ccm_core: `ccm_distributions`, `sample_ccm()`

plot.ccm_sample *Methods for ccm_sample Objects*

Description

Printing, summarizing, and plotting methods for results generated by `sample_ccm`.

Usage

```
## S3 method for class 'ccm_sample'
plot(
  x,
  stats = NULL,
  type = c("density", "hist", "trace"),
  target_distr = FALSE,
  ...
)

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

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

Arguments

`x`, `object`	An object of class `ccm_sample`.
`stats`	Character vector of statistic names to plot. If `NULL`, all targeted statistics are plotted.
`type`	Character string specifying the plot type: `"density"`, `"hist"`, or `"trace"`.
`target_distr`	Logical. If `TRUE`, overlays the target distribution (requires running `sample_target_distr` first).
`...`	Additional arguments passed to methods.

Details

For `type = "trace"`, setting `target_distr = TRUE` adds a red dashed line for the target mean and red dotted lines for the 2.5% and 97.5% quantiles.

sample_ccm *Sample from a Congruence Class Model (CCM)*

Description

sample_ccm generates networks from a Congruence Class Model using a Metropolis-Hastings MCMC framework. CCM samples networks where topological properties follow specified target probability distributions.

Usage

```
sample_ccm(
  network_stats,
  prob_distr,
  prob_distr_params,
  population,
  sample_size = 1000L,
  burnin = 200000L,
  interval = 1000L,
  cov_pattern = NULL,
  initial_g = NULL,
  use_initial_g = FALSE,
  partial_network = as.integer(0),
  obs_nodes = NULL,
  Obs_stats = NULL,
  stats_only = TRUE,
  verbose = 0
)
```

Arguments

network_stats
: Character vector of statistic names to be targeted. See ccm_properties for the full list of available network properties.

prob_distr
: Character vector of probability distribution names corresponding to each statistic. See ccm_distributions for details on implemented probability distributions and parameter requirements.

prob_distr_params
: List of parameter sets for each specified distribution. Each element must be a list containing the parameters required by the chosen distribution (e.g., list(shape, rate) for "gamma").

population
: Integer. The number of nodes in the network.

sample_size
: Integer. Number of MCMC samples to return. Default is 1000.

burnin
: Integer. Number of MCMC iterations to discard before sampling begins. Default is 200,000.

interval
: Integer. Thinning interval (number of iterations between samples). Default is 1000.

cov_pattern	Integer vector. Optional nodal attributes (group IDs) required for mixing or degree-mixing targets.
initial_g	An igraph object. The starting graph for the MCMC chain.
use_initial_g	Logical. If TRUE, the MCMC chain starts from initial_g.
partial_network	Integer. Reserved for future use.
obs_nodes	Integer vector. Reserved for future use in specifying observed nodes.
Obs_stats	Character vector of additional network statistics to monitor (but not target) during sampling. Reserved for future use.
stats_only	Logical. If TRUE, only sufficient statistics are returned; if FALSE, the list of sampled igraph objects is included.
verbose	Integer. Level of output logging (0 = silent, 1 = basic, 2 = detailed).

Details

The CCM framework allows generation of networks under flexible specifications for network structures. The framework decouples network properties from their probability distributions, enabling users can model a range of probability distributions for network properties (e.g., a Gamma or Multivariate-Normal distributions for degree mixing).

For specific mathematical details on how distributions like "dirmult" and "gamma" are implemented in the underlying C engine, refer to ccm_distributions.

The returned ccm_sample object has associated plot and sample_target_distr methods for diagnostic and comparative analysis.

Value

An object of class ccm_sample containing:

- mcmc_stats: A data frame of sampled network statistics.
- population: The number of nodes in the network.
- prob_distr: The names of the target distributions used.
- prob_distr_params: The parameter values used for the target distributions.
- network_stats: The names of the network statistics targeted.
- cov_pattern: The nodal covariate pattern used (if any).
- target_distr: A list containing target distribution samples, populated by calling sample_target_distr().
- g: A list of sampled igraph objects (or the last network if stats_only = TRUE).

See Also

ccm_properties, ccm_distributions, sample_target_distr, plot.ccm_sample

Other ccm_core: ccm_distributions, ccm_properties

Examples

```
# 1. Define target distributions and sample from the CCM
ccm_sample <- sample_ccm(
  network_stats = "edges",
  prob_distr = "poisson",
  prob_distr_params = list(list(350)),
  population = 50
)

# 2. Generate theoretical samples for the same target
ccm_sample <- sample_target_distr(ccm_sample)

# 3. Visualize MCMC samples against theoretical target
plot(ccm_sample, type = "hist", target_distr = TRUE)
```

---

sample_target_distr *Generate Samples from Target Distributions*

Description

This function draws samples directly from the target probability distributions specified in a `ccm_sample` object. These samples serve as a "ground truth" to evaluate whether the MCMC chain has converged to the intended target.

Usage

```
sample_target_distr(object, n_sim = nrow(object$mcmc_stats))
```

Arguments

`object`	An object of class `ccm_sample` generated by `sample_ccm`.
`n_sim`	Integer. The number of independent samples to draw from the theoretical target distributions. Default is equal to the number of CCM samples.

Details

This function performs direct i.i.d. sampling (e.g., using `rpois`, `rnorm`, etc.) based on the parameters stored in the `ccm_sample` object. It does not use MCMC. The resulting samples are used by `plot.ccm_sample` when `include_theoretical = TRUE` is specified.

Value

The input `ccm_sample` object with the `target_distr` slot populated. This slot contains a data frame of statistics sampled directly from the target distributions.

Examples

```
# 1. Generate MCMC samples
ccm_sample <- sample_ccm(
  network_stats = "edges",
  prob_distr = "poisson",
  prob_distr_params = list(list(350)),
  population = 50
)

# 2. Generate theoretical samples for comparison
ccm_sample <- sample_target_distr(ccm_sample, n_sim = 1000)

# 3. Compare MCMC to theoretical target
plot(ccm_sample, stats = "edges", type = "hist", target_distr = TRUE)
```

Index

∗ **ccm_core**
    ccm_distributions, 2
    ccm_properties, 3
    sample_ccm, 5

ccm_distributions, 2, *3*, *5*, *6*
ccm_properties, *2*, 3, *5*, *6*

plot.ccm_sample, 4, *6*, *7*
print.ccm_sample *(*plot.ccm_sample*)*, 4

sample_ccm, *2*, *3*, 5, *7*
sample_target_distr, *6*, 7
summary.ccm_sample *(*plot.ccm_sample*)*, 4