The R package BDgraph for Bayesian structure learning in graphical models

Mohammadi, A.; Dobra, A.

Publication date
2017
Document Version
Final published version
Published in
ISBA Bulletin

SOFTWARE HIGHLIGHT

THE R PACKAGE BDGRAPH FOR BAYESIAN STRUCTURE LEARNING IN GRAPHICAL MODELS

Abdolreza Mohammadi, Adrian Dobra
University of Amsterdam, University of Washington
a.mohammadi@uva.nl, adobra@uw.edu

Graphical models [12] provide a probabilistic framework to characterize the multivariate dependency structure among random variables. These models have received considerable attention in the literature, and have a vast domain of applicability that encompasses all the scientific fields in which the analysis of multivariate datasets is key, e.g. biology, neurosience, social sciences, and economics. Graphical models make use of graphs to represent relations (e.g., independence or conditional independence) among random variables. A crucial step in data analysis with graphical models is estimating the underlying graph. This is a very difficult computational problem when many random variables are involved. Bayesian methods provide a flexible framework for incorporating uncertainty about the graph structure: inference and estimation are based on averages of the posterior distribution of quantities of interest, weighted by the corresponding posterior probabilities of graphs [9].

The package `BDgraph` [17, 18] for `R` [22] provides easy-to-use functions for Bayesian structure learning in undirected graphical models for continuous, discrete, and mixed variables. The package implements recent results in the Bayesian literature, including [3, 4, 5, 14, 15, 16]. The package provides several distinctive features: (1) the computationally intensive tasks have efficient parallel implementations in `OpenMP` [19]; (2) all the code is written in `C++` and interfaced with `R`; (3) in addition to functions for Gaussian graphical models (GGMs), `BDgraph` provides functions for fitting Gaussian copula graphical models (GCGMs); (4) `BDgraph` has functions for graph determination for graphical models continuous, discrete and mixed variables based on the marginal pseudo-likelihood (MPL) approach [5, 20]. The MPL gives a practical way to balance computational complexity and accuracy to scale up to large-scale problems.

Over the last couple of years, the `BDgraph` package has been constantly improved in terms of the speed of computations, and its functionality has been extended. Now at version `2.43`, the package offers the end-user a practical option for carrying out Bayesian structure learning in undirected graphical models with several hundreds of variables.

1 Software design and implementation

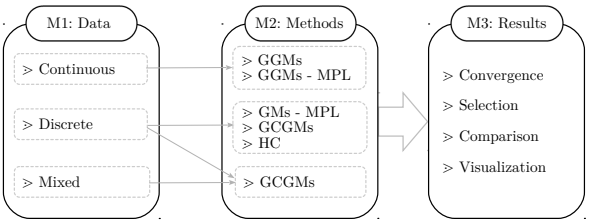

Figure 1: Functional modules of the `BDgraph` package: (M1) data simulation; (M2) methods and search algorithms; and (M3) various functions for convergence check of the search algorithms, graph selection, comparison and goodness-of-fit, and graph visualization.

Module 1 (M1) – Data simulation. The function `bdgraph.sim` generates multivariate Gaussian, discrete and mixed data given different types of undirected graphs, including "random", "cluster", "scale-free", "hub", "circle", "AR(1)", "AR(2)" and "fixed". Users can employ this function to simulate multivariate data based on these types graphs, or just to simulate graphs of those types.

Module 2 (M2) – Methods and algorithms. The main functions of this module are called `bdgraph` and `bdgraph.mpl`. They implement several estimation methods with different sampling algorithms, as follows.

- Graph estimation in GGMs using the birth-death MCMC (BDMCMC) sampling algorithm described in [15, 16]. This approach is implemented in the function `bdgraph`.
- Graph estimation in GCGMs for non-Gaussian, discrete, and mixed data, using

the BDMCMC sampling algorithm described in [14]. This approach is implemented in the function `bdgraph`.
- Graph estimation based on the reversible jump MCMC (RJMCMC) structural learning algorithms [3, 4]. This approach is implemented in the function `bdgraph`.
- Graph estimation based on MPL for discrete, Gaussian, and non-Gaussian data, using the BDMCMC sampling algorithm described in [5]. This approach is implemented in the function `bdgraph.mpl`.
- Graph estimation based on the hill-climbing (HC) algorithm for discrete data described in [20]. This approach is implemented in the function `bdgraph.mpl`.

Module 3 (M3) – Results. This module includes four types of functions:

- *Graph selection*: The functions `select`, `plinks`, and `pgraph` provide the selected graph, the posterior edge inclusion probabilities, and the posterior probability of each graph, respectively.
- *Convergence check*: The functions `plotcoda` and `traceplot` provide several visualization plots to monitor the convergence of the sampling algorithms.
- *Comparison and goodness-of-fit*: The functions `compare` and `plotroc` provide several comparison measures and a receiver operating characteristic (ROC) plot for model performance comparison.
- *Visualization*: The functions `plot.bdgraph` and `plot.sim` produce graph visualizations of the simulated data and estimated graphs. They are implemented based on `igraph` package [2] for `R` [22].

While the `BDgraph` package has functions for graph estimation using several different methods including RJMCMC and HC, those methods based on the BDMCMC algorithm have key advantages in terms of their computational performance. A brief introduction to BDMCMC for graphical models is presented in the next section.

2 Bayesian structure learning in graphical models

A graphical model for a random vector $\mathrm{X} = (X_1, X_2, \ldots, X_p)$ is specified by an undirected graph $G = (V, E)$ where $V = \{1, ..., p\}$ are vertices or nodes, and $E \subset V \times V$ are edges or links [12]. A vertex $i \in V$ of G corresponds with variable X_i. The absence of an edge between vertices i and j in G means that X_i and X_j are conditional independent given the remaining variables $X_{V\setminus\{i,j\}}$. The graph G also has a predictive interpretation. Denote by $\mathsf{nbd}_G(i) = \{j \in V : (i, j) \in E\}$ the neighbors of vertex i in G. Then X_i is conditionally independent of $X_{V\setminus(\mathsf{nbd}_G(i)\cup\{i\})}$ given $X_{\mathsf{nbd}_G(i)}$ which implies that, given G, a mean squared optimal prediction of X_i can be made from the neighboring variables $X_{\mathsf{nbd}_G(i)}$.

We focus on the structural learning problem [6, 11] which aims to estimate the structure of G (i.e., which edges are present or absent in E) from the available data $\mathrm{x} = (x^{(1)}, ..., x^{(n)})$. In a Bayesian framework, we explore the posterior distribution of G conditional on the data x, i.e.

$$\mathsf{P}(G \mid \mathrm{x}) = \frac{\mathsf{P}(G)\mathsf{P}(\mathrm{x} \mid G)}{\sum_{G\in\mathcal{G}_p} \mathsf{P}(G)\mathsf{P}(\mathrm{x} \mid G)}, \qquad (1)$$

where $\mathsf{P}(G)$ is a prior distribution on the graph space $\mathcal{G}_p$ and $\mathsf{P}(\mathrm{x} \mid G)$ is the marginal likelihood of the data conditional on G [11]. Determining the graphs with the highest posterior probabilities (1) is a complex problem since the number of possible undirected graphs $2^{\binom{p}{2}}$ becomes large very fast as p increases. For example, for $p = 50$, the number of possible undirected graphs exceeds the largest possible value in `R` ($1.8e + 308$). This motivated the development of computationally efficient search algorithms for exploring large spaces of graphs that have the ability to move quickly towards high posterior probability regions by taking advantage of local computation.

Among them, the BDMCMC algorithm [5, 14, 15, 16] is a trans-dimensional MCMC algorithm, and represents an alternative to the well known RJMCMC algorithm [8]. The BDMCMC algorithm is based on a continuous time birth-death Markov process [21]. Its underlying sampling scheme traverses $\mathcal{G}_p$ by adding and removing edges corresponding to the birth and death events. Given that the process is at state $G = (V, E)$, we define the birth and death events as independent Poisson processes as follows:

Birth event – each edge $e \in \overline{E}$ where $\overline{E} = \{e \in V \times V : e \notin E\}$, is born independently of other edges as a Poisson process with rate $B_e(G)$. If the birth of edge e occurs, the process jumps to $G^{+e} = (V, E \cup \{e\})$ which is a graph with one edge more than G.

Death event – each edge $e \in E$ dies independently of other edges as a Poisson process with

rate $D_e(G)$. If the death of edge e occurs, the process jumps to $G^{-e} = (V, E \setminus \{e\})$ which is a graph with one edge less than G.

This birth-death Markov process is a jump process with intensity $\alpha(G) = \sum_{e \in \overline{E}} B_e(G) + \sum_{e \in E} D_e(G)$. Its waiting time to the next jump has an exponential distribution with mean $1/\alpha(G)$. Thus, the birth and death probabilities are proportional to the birth and death rates.

To optimize the convergence speed, following [5, 16], we define the birth and death rates as follows:

$$R_e(G) = \min\left\{\frac{\mathsf{P}(G^* \mid \mathrm{x})}{\mathsf{P}(G \mid \mathrm{x})}, 1\right\}, \quad \text{for each } e \in \{E \cup \overline{E}\}, \tag{2}$$

where for the birth of edge e we take $G^* = (V, E \cup \{e\})$, and for the death of edge e we take $G^* = (V, E \setminus \{e\})$. The rates are calculated based on the MPL approach as described in [5].

The BDMCMC algorithm is presented in pseudo-code in Algorithm 1. It samples from the target posterior distribution (1) on $\mathcal{G}_p$ based on the above birth-death mechanism.

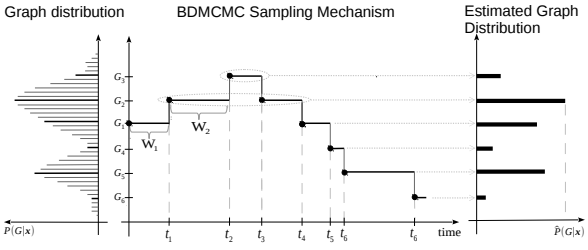

Figure 2: The left and right panels display the true and estimated posterior distribution (1) on the graph space. The middle panel reports sample scheme of Algorithm 1 where $\{W_1, W_2, \ldots\}$ denote waiting times, and $\{t_1, t_2, \ldots\}$ denote jumping times.

Figure 1 shows how the output from Algorithm 1 can be used to estimate posterior quantities of interest. The output consists of a set of sampled graphs and a set of waiting times $\{W_1, W_2, \ldots\}$. Based on the Rao-Blackwellized estimator [1], the estimated posterior probability of each sampled graph is proportional to the expectation of the length of the holding time in that graph which is estimated as the sum of the waiting times in that graph. The posterior inclusion probability of an edge $e \in V \times V$ is estimated by

$$\widehat{\mathsf{P}}(\text{edge } e \mid \mathrm{x}) = \frac{\sum_{t=1}^{N} \mathsf{I}(e \in G^{(t)}) W(G^{(t)})}{\sum_{t=1}^{N} W(G^{(t)})},$$

where N denotes the number of iterations, and $\mathsf{I}(\cdot)$ is an indicator function: $\mathsf{I}(e \in G^{(t)}) = 1$ if $e \in G^{(t)}$, and 0 otherwise. This estimation is implemented in the function `plinks` of `BDgraph`.

3 The user interface of `BDgraph`

We exemplify the user interface of the `BDgraph` package by analyzing the "reinis" dataset which is available in the package. The dataset consists of 6 binary variables that are potential risk factors of coronary heart disease: smoking (`smoke`), strenuous mental work (`mental`), strenuous physical work (`phys`), systolic blood pressure (`systol`), ratio of beta and alpha lipoproteins (`protein`), and family anamnesis of coronary heart disease (`family`). These data were collected from 1841 men employed in a car factory in Czechoslovakia [7].

```
R> library( BDgraph )
R> data( reinis )          # Load the data
```

Since "reinis" is a binary contingency table, we apply the Bayesian structure learning framework based on the MPL of [5] by calling the function `bdgraph.mpl` with the option `method = "dgm-binary"`. The default prior of `bdgraph.mpl` on the graph space is uniform. This is the prior we use here. We run BDMCMC for $10,000$ iterations with $6,000$ discarded as burn-in with the call:

```
R> sample <- bdgraph.mpl( data = reinis,
+ method = "dgm-binary",
+ algorithm = "bdmcmc",
+ iter = 10000,
+ burnin = 6000,
+ save.all = TRUE )
```

We specify the option "`save.all = TRUE`" to save all the sampled graphs in order to check the convergence of the algorithm. Running this function takes less than 1 second on a laptop computer, as the computational intensive tasks are performed in `C++` in parallel. Users can obtain the adjacency matrix of the selected graph (`selected_g`) and the estimated posterior probabilities of all possible edges (`p_links`) as follows:

```
R> summary( sample )
$selected_g
        smoke mental phys systol protein family
smoke       .      .    1      1       1      .
mental      .      .    1      .       .      .
phys        .      .    .      .       .      .
systol      .      .    .      .       1      .
protein     .      .    .      .       .      .
family      .      .    .      .       .      .
```

Algorithm 1 BDMCMC algorithm

Input: A graph $G = (V, E)$ and data x
  for N iterations **do**
    for all the possible edges in parallel **do**
      Calculate the birth and death rates in (2),
    end for
    Calculate the waiting time:

$$W(G) = \frac{1}{\left(\sum_{e \in \overline{E}} B_e(G) + \sum_{e \in E} D_e(G)\right)}.$$

    Update G based on birth/death probabilities.
  end for
Output: Samples from the posterior distribution (1).

```
$p_links
     smoke mental phys systol protein family
[1,]     .       .    1   0.75    1.00      .
[2,]     .       .    1      .    0.11   0.06
[3,]     .       .    .      .    0.01      .
[4,]     .       .    .      .    0.99      .
[5,]     .       .    .      .       .      .
[6,]     .       .    .      .       .      .
```

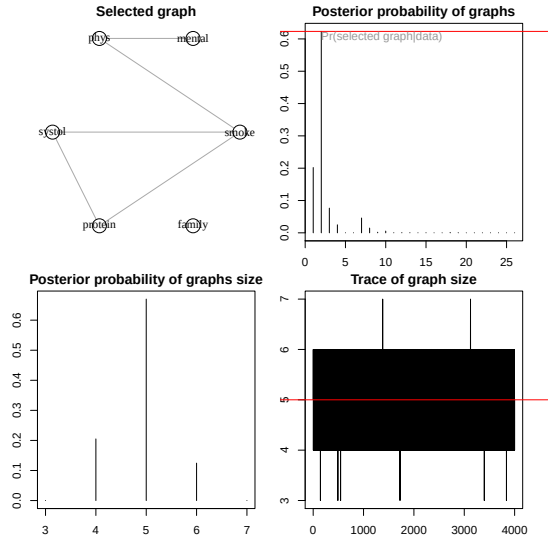

Figure 3: Visualization summary of the BDMCMC algorithm for the "reinis" data.

The function `summary` also generates a visualization summary – see Figure 3. The top-left panel gives the graph with the estimated highest posterior probability. The top-right panel gives the estimated posterior probabilities of all the graphs which are visited by the BDMCMC algorithm; it indicates that the algorithm visited 27 different graphs, and that the estimated highest posterior probability is around 0.63. The bottom-left panel shows the estimated posterior probabilities of the size of the graphs (number of edges); it indicates that the algorithm visited mainly graphs with sizes 4, 5 and 6. The bottom-right panel shows a trace plot based on the size of the sampled graphs.

We remark that the edges that belong to the highest posterior probability graph from Figure 3 are among the edges of the graphs determined for the "reinis" data using the `gRim` package [10], the MCMC algorithm of [13] or the loglinear model determination method of [7].

The convergence of the sampling algorithm from the posterior distribution of the graphs can be examined with the command:

```
R> plotcoda( sample )
```

This produces Figure 4. This plot shows that the BDMCMC algorithm converges after around 1000 iterations.

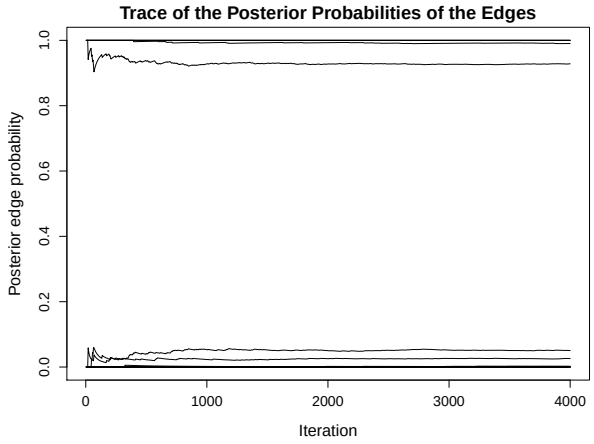

Figure 4: Trace plot showing the convergence of the the BDMCMC algorithm based on the trace of estimated posterior probability of all possible edges for the "reinis" data.

4 Analyzing a hyper-sparse high-dimensional contingency table with `BDgraph`

We perform structural learning in a $p = 214$ dimensional binary contingency table constructed by mapping geolocated tweets into the district municipalities of South Africa. The complete analysis of this table with `BDgraph` is presented in [5]. This table is hyper-sparse: only 55015 cells contain positive counts (the logarithm of the percentage of non-zero counts is -132.813). Among the 55015 non-zero counts, there are 46175 (83.93%) counts of 1, 3439 (6.25%) counts of 2, 1411 (2.56%) counts of 3, 747 (1.36%) counts of 4, and 476 (0.87%) counts of 5. The top five largest counts are 58929, 42781, 28731, 28197 and 22313.

The function `bdgraph.mpl` implements the following prior on the space of graphs [11]:

$$\mathsf{P}(G) \propto \left(\frac{\beta}{1-\beta}\right)^{|E|}, \qquad (3)$$

where $\beta \in (0,1)$. In this application, we employ the prior (3) with $\beta = 1/\binom{214}{2} = 4.388 \times 10^{-5}$ by setting the `g.prior` argument of `bdgraph.mpl`. With this choice, the expected number of edges is 1, thus sparser graphs receive larger prior probabilities compared to denser graphs.

We run the function `bdgraph.mpl` for $10,000$ iterations with $6,000$ iterations as burn-in as follows:

```
R> sample <- bdgraph.mpl( data = TwitterData,
+ method = "dgm-binary",
+ multi.update = 1, g.prior= 2/(p*(p-1)),
+ iter = 10000, burnin = 6000 )
```

This function call completed after about 12 hours on a computer with an Intel Xeon 2.6 GHz processor with 48 cores, and a Linux operating system. The resulting median graph that contains the edges with estimated posterior probabilities greater than 0.5 has 1534 edges, and it is determined and visualized with the following command:

```
R> select( sample, cut = 0.5, vis = TRUE )
```

By using option `vis = TRUE`, the function `select` plots the selected graph. The function `plinks` returns the matrix with estimated posterior probabilities of all possible edges in the graph. A visualization of this 214×214 matrix is presented in Figure 5.

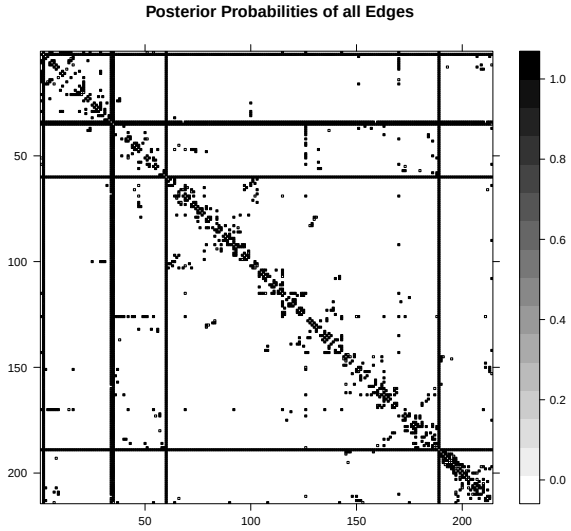

Figure 5: Heatmap of the 214×214 matrix of posterior inclusion probability of edges for the contingency table from Section 4.

5 More information

The `BDgraph` package is available on the CRAN at:
https://cran.r-project.org/web/packages/BDgraph

References

[1] O. Cappé, C.P. Robert, and T. Rydén. Reversible jump, birth-and-death and more general continuous time markov chain monte carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(3):679–700, 2003.

[2] G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[3] A. Dobra and A. Lenkoski. Copula gaussian graphical models and their application to modeling functional disability data. *The Annals of Applied Statistics*, 5(2A):969–993, 2011.

[4] A. Dobra, A. Lenkoski, and A. Rodriguez. Bayesian inference for general gaussian graphical models with application to multivariate lattice data. *Journal of the American Statistical Association*, 106(496):1418–1433, 2011.

[5] A. Dobra and A. Mohammadi. Loglinear model selection and human mobility. *arXiv preprint arXiv:1711.02623*, 2017.

[6] M. Drton and M. H. Maathuis. Structure learning in graphical modeling. *The Annual Review of Statistics and Its Application*, 4:365–393, 2017.

[7] D. Edwards and T. Havranek. A fast procedure for model search in multidimensional contingency tables. *Biometrika*, 72(2):339–351, 1985.

[8] P. J. Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[9] Jennifer A Hoeting, David Madigan, Adrian E Raftery, and Chris T Volinsky. Bayesian model averaging: a tutorial. *Statistical science*, pages 382–401, 1999.

[10] S. Hojsgaard, D. Edwards, and S. Lauritezen. *Graphical Models with R*. Springer-Verlag, New York, 2012.

[11] B. Jones, C. Carvalho, A. Dobra, C. Hans, C. Carter, and M. West. Experiments in stochastic computation for high-dimensional graphical models. *Statistical Science*, 20(4):388–400, 2005.

[12] S. L. Lauritzen. *Graphical Models*, volume 17. Oxford University Press, USA, 1996.

[13] H. Massam, J. Liu, and A. Dobra. A conjugate prior for discrete hierarchical log-linear models. *Annals of Statistics*, 37:3431–3467, 2009.

[14] A. Mohammadi, F. Abegaz Yazew, E. van den Heuvel, and E. C. Wit. Bayesian modelling of dupuytren disease using gaussian copula graphical models. *Journal of Royal Statistical Society-Series C*, 66(3):629–645, 2017.

[15] A. Mohammadi, H. Massam, and G. Letac. The ratio of normalizing constants for bayesian graphical gaussian model selection. *arXiv preprint arXiv:1706.04416*, 2017.

[16] A. Mohammadi and E. C. Wit. Bayesian structure learning in sparse gaussian graphical models. *Bayesian Analysis*, 10(1):109–138, 2015.

[17] A. Mohammadi and E. C Wit. BDgraph: Bayesian structure learning of graphs in R. *arXiv preprint arXiv:1501.05108v4*, 2017.

[18] M. Mohammadi and E. C. Wit. *BDgraph: Bayesian Structure Learning in Graphical Models using Birth-Death MCMC*, 2017. R package version 2.43.

[19] OpenMP Architecture Review Board. OpenMP application program interface version 3.0, 2008.

[20] J. Pensar, H. Nyman, J. Niiranen, and J. Corander. Marginal pseudo-likelihood learning of discrete markov network structures. *Bayesian Analysis*, 2017.

[21] C. J. Preston. Spatial birth-and-death processes. *Bulletin of the International Statistical Institute*, 46:371–391, 1977.

[22] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.