Bayesian Model Averaging methods and R package for gene network construction

Ka Yee Yeung  
Institute of Technology  
University of Washington, Tacoma, WA 98402.  
kayee@uw.edu

Chris Fraley  
Department of Statistics  
University of Washington, Seattle, WA 98195  
fraley@uw.edu

William Chad Young  
Department of Statistics  
University of Washington, Seattle, WA 98195  
wmc@uw.edu

Roger Bumgarner  
Department of Microbiology  
University of Washington, Seattle, WA 98195.  
rogerb@uw.edu

Adrian E. Raftery  
Department of Statistics  
University of Washington, Seattle, WA 98195  
raftery@uw.edu

ABSTRACT
Advances in biotechnology allow the generation of big data quantifying activities or signals across the entire genome. These genome-wide data can be used to infer functional relationships between biological entities. In this paper, we will review Bayesian Model Averaging (BMA) methods developed to infer these gene networks. In particular, our work uses a regression framework that allows systematic integration of multiple types of biological data. We will also present our latest contributions to the software implementation of our methods.

Many methods have been developed to infer gene networks, and different methods generally yield different networks. Hence, the assessment and evaluation of inferred networks is of great importance. We present an R package “networkBMA” for network inference and assessment implementing: (i) Bayesian Model Averaging methods for network inference using time series gene expression data; (ii) assessment functions that compute contingency tables and associated evaluation statistics for comparing inferred networks to complete or incomplete external knowledge; (iii) plotting functions that compute receiver operating characteristic (ROC) and precision-recall (PR) curves. Our package also includes in silico and real test data that demonstrate the usability of the functions.

Categories and Subject Descriptors
G.3 [Probability and Statistics]: Statistical computing.
G.4 [Mathematical software]: Algorithm design and analysis.

General Terms
Algorithms, Performance, Experimentation.

Keywords
Bayesian statistics, software, gene network inference, data integration, computational biology, data sources, network inference.

1. INTRODUCTION
The ultimate goal of genomics research is to enable the discovery of novel diagnostics and therapeutics. Advances in biotechnology allow the measurement of the signal or activity levels across the entire genome in the order of thousands or even tens of thousands of biological entities. Various types of these genome-wide data, such as gene expression and sequence data, have been generated and made available from public databases such as the Gene Expression Omnibus (GEO) [1, 2], ArrayExpress [3] and the European Nucleotide Archive (ENA) [4]. By mining these genome-wide data, we aim to deduce the complex interactions between genes and gene products through predictive modeling. The resulting models can subsequently be used to develop novel therapies [5, 6].

In this paper, we define a gene network as a directed graph, in which each node represents a messenger RNA (mRNA) level and each directed edge (r→g) represents the relationship between the expression levels (or activity levels) of a regulator r and a gene g. We aim to infer the directed edges that describe the relationships among the nodes. This is a challenging problem, especially on a genome-wide scale, since the goal is to unravel a small number of regulators (parent nodes) out of thousands of candidate nodes in the graph. Another challenge is the effective integration of external knowledge and data sources to guide the network search.

1.1 Related Work
A variety of methods have been developed to infer gene networks, for example, Bayesian networks [5, 7-10], ordinary differential equations [11] and regression-based methods [12-23]. In regression-based methods, network construction is formulated as a series of variable (feature) selection problems to infer regulators (parent nodes) for each gene. Examples of regression-based network inference methods include regression trees [24-27], forward selection, nonparametric regression embedded within a Bayesian network [28], and L1-norm regularization methods such as the elastic net [29, 30] and weighted LASSO [31].

1.2 Our Contributions
In this manuscript, we review our Bayesian Model Averaging (BMA) methods for gene network inference using time series...
gene expression data [32-34], and present our software implementation (“networkBMA” R package) that has not previously been documented in the literature. Specifically, we formulate network inference as a variable selection problem in which candidate regulators are selected for each gene. Integration of multiple genome-wide data sources is accomplished using prior probabilities in the Bayesian framework. These prior probabilities are used to guide the search of regulators (parent node) in the model space. The greatest challenge in formulating network construction as a series of variable selection problems is that there are usually far more candidate regulators than observations for each gene. Our methods build on BMA, which is a multivariate variable (feature) selection method that accounts for uncertainty in model selection [35, 36]. In the case of high-dimensional gene expression data, it is highly likely that there are multiple models (sets of regulators) that fit the data well. BMA is an ensemble method that averages over multiple possible models. This is in contrast to other variable selection methods that select a single “best” model from models of similar quality. The resulting network is a directed graph consisting of a set of directed edges from regulators to genes, with edges calibrated by posterior probabilities.

Our “networkBMA” package (with major revision in April 2014, publicly available at http://bioconductor.org/packages/release/bioc/html/networkBMA.html) extends our previous work in two main ways. First, we unify the iBMA and ScanBMA methods for network construction and provide a software implementation that allows the specification of prior probabilities to bias the search of candidate regulators for each gene. Second, we add functions for network assessment, including functions for computing contingency tables for inferred networks as compared to complete or incomplete reference regulatory relationships, functions for computing evaluation statistics from contingency tables, and functions for plotting receiver operating characteristic (ROC) curves and precision-recall (PR) curves.

2. BAYESIAN MODEL AVERAGING (BMA)

2.1 BMA as a variable selection method

BMA is a variable selection approach that takes model uncertainty into account by averaging over the posterior distribution of a quantity of interest based on multiple models, weighted by their posterior model probabilities [35, 36]. In the context of network construction, each model consists of a set of candidate regulators. To efficiently identify a small set of promising models out of all possible models, our approach applies the leaps and bounds algorithm [37] to identify the best nbest models for each number of variables (i.e., regulators), and Occam’s window to discard models with much lower posterior model probabilities than the best one [38]. The Bayesian Information Criterion (BIC) [39] is used to approximate each model’s integrated likelihood, from which its posterior model probability can be determined. BIC corrects for over-fitting with a penalty term for the number of parameters in the model.

2.2 BMA as a network inference algorithm

We formulate network construction from time series data as a regression problem in which the expression of each gene is predicted by a linear combination of the expression of candidate regulators at the previous time point. Specifically, $X_{g,t} = \beta_{0,g} + \sum_{h \in H} \beta_{h,g} X_{h,t-1} + \epsilon_{g,t}$, where $X_{g,t}$ is the expression of gene $g$ at time $t$, $H$ is the set of regulators for gene $g$, $\beta_i$’s are the regression coefficients, and $\epsilon_{g,t} \sim N(0, \sigma^2)$ is the error term for gene $g=1, \ldots, n$ and time $t=1, \ldots, T$.

2.2.1 Iterative BMA algorithm (iBMA)

Since BMA in its original form is not applicable to high-dimensional data that contain more variables than samples, we developed the iterative BMA (iBMA) algorithm [32, 40]. In iBMA, we use a pre-processing step to rank all variables. We then iteratively apply the original BMA to the top $w$ variables ($w=30$ by default), and discard predictor variables with low posterior inclusion probabilities. In the iterative step, new variables from the ranked list are added to replace the discarded variables. This procedure of repeatedly applying BMA and variable swaps is continued until the $n_{var}$ top ranked variables have been processed.

2.2.2 Scan BMA algorithm (ScanBMA)

ScanBMA is an alternative algorithm for exploring the space of models for BMA when there are more variables than samples [34]. It does so by taking a set of the best models found thus far and looking at models in which a single variable is added or removed, adding those models to the set of best models to explore around if their score is within a factor of the score of the best model. Additionally, ScanBMA allows the use of Zellner’s g-prior [41], which replaces BIC for scoring the models. Zellner’s g-prior is more flexible than BIC and can be either specified or estimated using an Expectation-Maximization (EM) algorithm.

There are two distinguishing features of ScanBMA. First, a pre-processing step ranking all variables via a univariate measure is used in the model exploration step of iBMA, but not in ScanBMA. Second, there is no upper limit on the maximum size of the models inferred in ScanBMA. This is in contrast to iBMA, in which the number of maximum variables is limited by the BMA window size $w$.

2.2.3 Applications to static (non-time series) data

Our iBMA and ScanBMA algorithms can be modified to be applied to non-time series data [32]. Specifically, let $Y_{g,e}$ be the expression of gene $g$ in experiment $e$, $H$ is the set of regulators for gene $g$ in a candidate model, $\beta_i$’s are the regression coefficients, and $\epsilon_{g,e} \sim N(0, \sigma^2)$ is the error term for $g=1, \ldots, n$ and $e=1, \ldots, E$. Our model is defined as $Y_{g,e} = \beta_{0,g} + \sum_{h \in H} \beta_{h,g} X_{h,e} + \epsilon_{g,e}$. An issue with this formulation is that the resulting networks may contain directed cycles. In order to yield a well-defined probability distribution, we need a Gaussian graphical model that does not contain any cycles [42-44]. Drawing from our previous work [32, 33], we identify strongly connected components using the directed graph resulting from applying iBMA to each gene, and then greedily remove the edges with the lowest posterior probabilities until no strongly connected component exists in the graph.

2.3 Data integration in BMA

A major challenge of data mining in computational biology is data integration. In the context of gene networks, we aim to integrate multiple genome-wide data sources to infer robust, accurate and compact gene-to-gene interactions. We have previously developed a supervised framework to integrate multiple data sources and illustrated our framework using yeast data as an example [32, 33]. Specifically, we compute prior
probabilities of regulatory relationships using multiple data sources. Known regulatory relationships in the form of transcription factor and gene (TF,G) pairs from the literature curated in various yeast databases are used as positive training examples. Randomly generated (TF,G) relationships that are not documented in any databases are used as negative training examples. We further adjust for the sampling bias between positive and negative training examples using the expected fraction of regulatory relationships in the literature [33]. These prior probabilities are then used to guide the selection of candidate regulators (parent nodes) for each gene in the regression step using gene expression data. While our work was done primarily in yeast, the supervised framework is applicable to other organisms for which similar data sets exist.

3. IMPLEMENTATION

3.1 networkBMA R package

The “networkBMA” package is written in R and C++, and is publicly available from the Bioconductor project [45]. The core function networkBMA returns a set of directed edges consisting of the inferred regulator (parent), target gene (child), and the corresponding posterior probability of the inferred relationship. The user can subsequently specify a posterior probability threshold to filter out edges with low posterior probabilities.

In our latest revision (April 2014), we add the implementation of ScanBMA, written in R and C++, which provides a significant increase in computational efficiency over our previous implementation of iBMA in R. The most computationally intensive routines searching the model space using the ScanBMA algorithm are written in C++. Our latest revision allows the inference of large gene networks (i.e., running the core function networkBMA across thousands of genes) to be accomplished in under minutes for small nvar such as 100, while also being viable for larger nvar in the thousands. Please refer to Young et al. [34] for detailed running time analyses. Key input parameters to the core function networkBMA include:

- The argument nvar specifies the number of top univariate ranked variables to be retained before the regression step in both iBMA and ScanBMA. During the regression step, this univariate ranking will only be used in iBMA, but not in ScanBMA.
- The argument ordering specifies the univariate measure used to pre-process and rank variables prior to the regression step. We extend our previous work [32, 33] by allowing users to choose from the univariate BIC score (ordering=”bic1”), user-specified prior probabilities (ordering=”prior”) or univariate BIC score corrected by prior probabilities (ordering=”bic1+prior”). The univariate BIC score (“bic1”) measures the fit of each predictor variable (candidate regulator) in the linear regression. The Bayesian Information Criterion (BIC), given by $-2 \log \text{likelihood} + p \log(n)$, where $p$ is the number of parameters (regression coefficients), and $n$ is the number of observations, is an approximation to the integrated likelihood for a model [46]. Ordering variables by the “bic1” score is equivalent to ordering them by likelihood, because in the univariate case $p \log(n) = 2 \log(n)$ is constant across variables (the parameters are the intercept and a coefficient for the variable in question). The option “bic1+prior” corrects the BIC score by subtracting twice the log odds of the prior probabilities, hence combining the contribution of the fit and the prior.
- The specification of prior probabilities to bias the search of candidate regulators for each gene is accomplished using the argument prior.prob in function networkBMA. This argument can be used to specify a matrix P such that entry (i,j) corresponds to the prior probability that regulator i regulates gene j computed using external data sources. If external data sources are unavailable, the user is encouraged to specify a constant prior probability, network density $\tau$, which is equal to the expected number of regulators per gene divided by the total number of genes in the genome. Guelzim et al. [47] estimated that each yeast gene is regulated by approximately 2.76 transcription factors on average, hence, we set this network density parameter to $2.76/6000 = 0.00046$ when inferring yeast networks. We have shown that this size prior yields compact and accurate networks [33].

3.2 Contingency tables for network assessment

Our “networkBMA” package allows the specification of either a complete or incomplete reference network to assess an inferred network. In the case of simulated data, a complete reference network (i.e., the underlying true network spanning all regulators and genes) is often available. However, for real biological data, knowledge of reference networks would at best, be incomplete. As an example, we compared networks inferred from yeast time-series data [32-34] to the literature curated regulatory relationships in the YEASTRACT database [48].

The function contabs.netxBMA in the networkBMA package computes a 2 x 2 contingency table to quantify the associations between an inferred and an incomplete reference network for each distinct value in a vector of posterior probability thresholds. This function computes the numbers of true positive (TP), false positive (FP), true negative (TN) and false negative (FN) using the set of regulators and genes covered by the reference network at a given posterior probability threshold to define the inferred edges. Table 1 shows the definitions of TP, FN, TN and FP in a contingency table.

Table 1. Definition of contingency tables.

<table>
<thead>
<tr>
<th>Inferred network</th>
<th>Gold standard reference network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>TP</td>
</tr>
<tr>
<td>No</td>
<td>FP</td>
</tr>
<tr>
<td></td>
<td>FN</td>
</tr>
<tr>
<td></td>
<td>TN</td>
</tr>
</tbody>
</table>

The function scores summarizes the concordance in the computed contingency tables using the following statistics:

- True positive rate (TPR) or sensitivity or hit rate or recall, $TPR = TP/(TP + FN)$
- False positive rate (FPR) = $FP/(FP + TN)$
- True negative rate (TNR) or specificity = $TN/(FP + TN) = 1 - FPR$
- False discovery rate (FDR) = $FP/(TP + FP)$
- Positive predictive value (PPV) or precision = $TP/(TP + FP)$
- Negative predictive value (NPV) = $TN/(TN + FN)$
- F1 score is defined as the harmonic mean of precision and recall, i.e., $F1 = 2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$
- Accuracy (ACC) = $(TP + TN) / (TP + FN + FP + TN)$
- $O/E$ ratio = $TP^*FN^*/(TP^*FP + FP^*FN)$


3.3 ROC and PR curves for network assessment

Functions roc and prc in the networkBMA package take the evaluation statistics returned by contabs.netwBMA over the range of posterior probability thresholds for the inferred edges, and plot the TPR against FPR in the ROC curve, and the precision against recall in the PR curve, respectively. TPR measures the fraction of correct edges among all edges in the reference network, precision measures the fraction of correct edges among all predicted edges, and FPR quantifies the fraction of incorrect edge predictions among edges not in the reference network. A perfect prediction would yield TPR=1, precision=1, and FPR=0. In general, good predictions would yield a point above the diagonal line in the ROC curve.

The contingency tables from “networkBMA” cover a range of values for the FPR (ROC) or recall (PR) that may not span the entire interval from 0 to 1. For ROC curves, we use linear interpolation, while for PR curves, we use the method described by Davis and Goadrich [49] to interpolate values outside the range. Davis and Goadrich [49] address practical issues of interpolation between points in the ROC and PR space. While linear interpolation between points makes sense for ROC curves, interpolation is more complicated in the PR space. As the level of Recall varies, the Precision does not necessarily change linearly due to the fact that FP replaces FN in the denominator of the Precision metric. As a result, linear interpolation gives an overly optimistic estimate of performance. The achievable PR curve can be found using the analogous ROC convex hull, and interpolation can be based on this curve. We use this method when plotting PR curves in the networkBMA package.

4. RESULTS

4.1 Datasets

The “networkBMA” package includes the following set of test data and reference networks:

1. A subset of the yeast time-series expression data consisting of 100 genes over 6 time points [32] and literature-curated regulatory relationships from the YEASTRACT database [48]. This time-series expression data measure the response to a drug perturbation over 6 time points at 10-minute intervals in 95 yeast segregants and 2 parental strains. The full dataset is available from http://www.ebi.ac.uk/arrayexpress/experiments/E-MTAB-412/. We also include the prior probabilities computed using external data sources for these 100 genes (as described in Section 2.3).

2. In-silico 10-gene and 100-gene time-series data over 21 time points and the corresponding reference networks from DREAM4 [50-54].

3. A subset of static (non-time series) yeast gene expression data [55].

4.2 Using the networkBMA package

We illustrate our package using the 100-gene yeast time series expression data (called timeSeries in our example). First, we use the core function networkBMA and apply ScanBMA. The prior probability matrix of this 100-gene subset is denoted by reg.prob. We can estimate the true posterior probabilities for a predictor with an estimated 100% (0%) posterior probability by comparing the posterior probability of the best model with the predictor removed (added) with that of the best model itself to assess the relative importance of each predictor. We call this process differentiation, and use arguments diff100 and diff0 as indicator variables denoting whether differentiation is performed. Subsequently, we threshold the resulting directed edges at 50% and obtain a network consisting of 439 edges.

4.3 Contingency tables

Next, we compare this inferred network to the reference network using contingency tables at posterior probability thresholds 50%, 75% and 90%.

4.4 Results

Figure 1 shows the ROC curve plotting the TPR against the FPR by varying posterior probability thresholds of the inferred networks. The area under the ROC curve is 0.65. See the vignette accompanying the “networkBMA” package for details of the R code used to produce this example.

![ROC curve produced by networkBMA](image)
5. CONCLUSIONS
We present the networkBMA bioconductor package for network inference and assessment. Our R package implements both the ScanBMA and iBMA methods, and allows specification of prior probabilities to bias the search of regulators towards candidates supported by additional data sources or expert knowledge. In addition, our package provides functions to evaluate inferred networks against complete or incomplete regulatory relationships using contingency tables, to compute various assessment statistics and to plot receiver operating characteristic (ROC) curves and precision-recall (PR) curves. Most importantly, our method and software implementation is highly computationally efficient compared to other network inference methods, with projected running time ~15 minutes for 20,000 genes, see [34]). Thus, networkBMA is a promising candidate to infer gene networks in mammalian systems.

6. ACKNOWLEDGMENTS
We would like to thank Drs. Kenneth Dombek, Kenneth Lo and John Mittler for their participation in the project.

AER, CF, KYY, REB and WCY are supported by NIH grant 5R01GM084163. KYY is also supported by the Institute of Translational Health Sciences (ITHS) eScience Seed Grant at University of Washington and the Microsoft Azure for Research Award. AER is also supported by NIH grants R01 HD054511 and R01 HD070936, and by Science Foundation Ireland ETS Walton visitor award 11/W.1/I207.

7. REFERENCES


