

# Uncovering gene regulatory relationships from time-series expression data using networkBMA

Ka Yee Yeung, Chris Fraley, Adrian E. Raftery, Wm. Chad Young  
Departments of Microbiology (KYY) and Statistics (CF, AER, and WCY)  
University of Washington

This document illustrates the use of the **networkBMA** R package (Fraley et al. 2012) to uncover regulatory relationships in yeast (*Saccharomyces cerevisiae*) from microarray data measuring time-dependent gene-expression levels in 95 genotyped yeast segregants subjected to a drug (rapamycin) perturbation.

## 1 Data

The expression data for this vignette is provided in the **networkBMA** package in the **vignette** database, which consists of three R objects:

**timeSeries**: A 582 by 102 data frame in which the first two columns are factors identifying the replicate and time (in minutes) after drug perturbation, and the remaining 100 columns are the expression measurements for a subset of 100 genes from the yeast-rapamycin experiment described in Yeung et al. (2011). There are  $582/6 = 97$  replicates (the 95 segregants plus two parental strains of the segregants), each with measurements at 6 time points. The complete time series data is available from Array Express (Parkinson et al. 2007) with accession number E-MTAB-412 (<http://www.ebi.ac.uk/arrayexpress/experiments/E-MTAB-412>).

**reg.known**: A 18 by 3 data frame giving known regulatory relationships among this subset of 100 genes. The first two columns give the regulator and target gene, respectively, while the third encodes the source of the regulatory information ('YPD' for Yeast Proteome Database (Hodges et al. 1999) and 'SCPD' for The Promoter Database of *Saccharomyces cerevisiae* (Zhu and Zhang 1999). In this example, we constrained **reg.known** to high-confidence experimental results obtained from biochemical (non-high-throughput) experiments.

**reg.prob**: A 100 by 100 matrix, giving probability estimates for regulatory relationships in which the  $(i, j)$  entry gives the estimated probability that gene  $i$  regulates gene  $j$ . These were computed using the supervised framework integrating multiple data sources of Lo et al. (2012).

**referencePairs**: A 2-column data frame giving 287 regulator-gene pairs among the selected set of 100 genes reported from the literature. In this yeast example, the reference network was extracted from the documented evidence from the YEASTRACT database (Teixeira et al. 2006), which includes curated regulatory relationships from the literature inferred from high-throughput experiments.

`brem.data`: An 85 by 111 subset of the data used for network inference in yeast (Brem et al. 2002, Brem and Kruglyak 2005). The rows correspond to genes and the columns to experiments. Provided courtesy of Rachel Brem.

```
> library(networkBMA)
> data(vignette)
> dim(timeSeries)
```

```
[1] 582 102
```

```
> dim(reg.prob)
```

```
[1] 100 100
```

```
> dim(brem.data)
```

```
[1] 85 111
```

```
> reg.known
```

|    | Regulator | TargetGene | source |
|----|-----------|------------|--------|
| 1  | YDR216W   | YKR009C    | YPD    |
| 2  | YER040W   | YPL111W    | YPD    |
| 3  | YER040W   | YKL015W    | YPD    |
| 4  | YER040W   | YOR348C    | YPD    |
| 5  | YJR094C   | YDR523C    | YPD    |
| 6  | YKL062W   | YMR169C    | YPD    |
| 7  | YKL062W   | YPL061W    | YPD    |
| 8  | YKL062W   | YAL062W    | YPD    |
| 9  | YKL062W   | YIL155C    | YPD    |
| 10 | YKL062W   | YFL014W    | YPD    |
| 11 | YKL062W   | YCR021C    | YPD    |
| 12 | YKL062W   | YDR258C    | YPD    |
| 13 | YKL062W   | YJR094C    | YPD    |
| 14 | YKL062W   | YER150W    | YPD    |
| 15 | YKL062W   | YNL194C    | YPD    |
| 16 | YBL103C   | YNL037C    | YPD    |
| 17 | YKL112W   | YCL064C    | SCPD   |
| 18 | YKL112W   | YHR051W    | SCPD   |

## 2 Network Modeling

Given the yeast expression data from the Rapamycin experiments, the `networkBMA` function can be invoked to estimate the probabilities of regulatory relationships using either ScanBMA or iterative Bayesian Model Averaging (Yeung et al. 2005, 2011). The ScanBMA algorithm, as shown in Algorithm 1, is preferred when available. The parameter "prior.prob" indicates

the prior probabilities of regulatory relationships. If the parameter "prior.prob" is set to a single positive fraction, it represents the probability of a regulator-gene pair in the network (*i.e.* the expected network density as defined in (Lo et al. 2012)). The parameter "prior.prob" can also be set to a matrix in which the (i,j) entry is the estimated prior probability that gene i regulates gene j. The default value of "prior.prob" is NULL, which implies that no prior information will be used in modeling the network.

```
> edges.ScanBMA <- networkBMA(data = timeSeries[,-(1:2)],
+                             nTimePoints = length(unique(timeSeries$time)),
+                             prior.prob = reg.prob,
+                             nvar = 50,
+                             ordering = "bic1+prior", diff100 = TRUE, diff0 = TRUE)
> edges.ScanBMA[1:9,]
```

|     | Regulator | TargetGene | PostProb |
|-----|-----------|------------|----------|
| 1   | YJL217W   | YJL217W    | 1        |
| 2   | YIL037C   | YIL037C    | 1        |
| 3   | YGL009C   | YGL009C    | 1        |
| 101 | YHR216W   | YHR216W    | 1        |
| 102 | YOL014W   | YOL014W    | 1        |
| 201 | YAL062W   | YAL062W    | 1        |
| 202 | YHR136C   | YHR136C    | 1        |
| 203 | YJR094C   | YJR094C    | 1        |
| 301 | YOR032C   | YOR032C    | 1        |

For each gene  $g$ , the observed gene expression of genes at time  $t - 1$  serve as linear predictors for modeling the observed expression of gene  $g$  at time  $t$ . BMA modeling results in a weighted average of models consisting of relatively small numbers of predictors. The probability of gene  $h$  being a linear predictor in the model for gene  $g$  is taken as the probability that gene  $h$  regulates gene  $g$  in the network.

There are options for ordering the variables (parameter "ordering") and specifying the number of ordered variables (parameter "nvar") to be included in the modeling. In both algorithms ScanBMA and iBMA, all the candidate variables (genes) are initially ranked using the method specified in "ordering", and the top "nvar" such variables will be used as input in the BMA regression step. Note that if ScanBMA is used, the parameter "ordering" will have no effect in the BMA regression step.

Differentiation can also be performed on edges returned with 0% or 100% posterior probability. To include known regulatory relationships, the iBMA algorithm must be used. This can be done as shown below, but the results shown subsequently will use the call with ScanBMA as the algorithm, as above.

```
> edges.iBMA <- networkBMA(data = timeSeries[,-(1:2)],
+                             nTimePoints = length(unique(timeSeries$time)),
+                             prior.prob = reg.prob, known = reg.known,
+                             nvar = 50, control = iBMAcontrolLM(),
+                             ordering = "bic1+prior", diff100 = FALSE,
```

```

+                                     diff0 = FALSE)
> edges.iBMA[1:9,]

```

|    | Regulator | TargetGene | PostProb |
|----|-----------|------------|----------|
| 1  | YBL103C   | YBL103C    | 1        |
| 2  | YNR053C   | YBL103C    | 1        |
| 3  | YOR206W   | YBL103C    | 1        |
| 4  | YKL112W   | YKL112W    | 1        |
| 5  | YMR229C   | YKL112W    | 1        |
| 8  | YDR216W   | YDR216W    | 1        |
| 9  | YDL170W   | YDR216W    | 1        |
| 10 | YOR302W   | YDR216W    | 1        |
| 11 | YPL265W   | YDR216W    | 1        |

---

**Algorithm 1:** ScanBMA

---

```

Initialize  $\mathcal{M}_{keep}, \mathcal{M}_{next} = \{\}$ 
Initialize  $\mathcal{M}_{active} = \{null\ model\}$ ,  $bestScore = 0$ 
while  $\mathcal{M}_{active}$  not empty do
    for model  $m_{new}$  in NeighborsOf( $\mathcal{M}_{active}$ ) do
         $mScore = \text{EvaluateModelScore}(m_{new})$ 
        if  $mScore$  in OccamsWindow( $bestScore$ ) then
            add  $m_{new}$  to  $\mathcal{M}_{next}$ 
             $bestScore = \text{BestModelScore}(bestScore, mScore)$ 
        end
    end
    Trim models from  $\mathcal{M}_{keep}$  according to  $bestScore$ 
    Add good models from  $\mathcal{M}_{active}$  to  $\mathcal{M}_{keep}$ 
     $\mathcal{M}_{active} = \text{good models from } \mathcal{M}_{next}$ 
end
return  $\mathcal{M}_{keep}$ 

```

---

### 3 Assessment of Network Models

Although, except for synthetic data, the true underlying network is unknown, the results can be assessed using a set of regulator-target gene network edges reported in the literature. The comparison is done as follows:

- Let  $E$  be the set of regulator-target gene edges resulting from **networkBMA**, possibly reduced using a probability threshold. In the context of the example in Section 2,  $E$  corresponds to the set of edges represented in the object **edges.ScanBMA**.
- Let  $K$  be the set of known regulator-target gene edges hardcoded in the modeling. In the example in Section 2,  $K$  corresponds to **reg.known**.

- Let  $L$  be the set regulator-target gene edges reported in the literature. In the example in Section 2,  $L$  corresponds to `referencePairs`.
- Let  $E \setminus K$  and  $L \setminus K$  be the set of pairs in  $E$  and  $L$ , respectively, with any hardcoded edges removed. In the example of Section 2,  $E$  represented by `edges.ScanBMA` contains 483 pairs, and  $L$  represented by `referencePairs` contains 287 pairs. Both  $E$  and  $L$  include all 18 of the known hardcoded edges  $K$  represented by `reg.known`. Hence  $E \setminus K$  contains 465 pairs, and  $L \setminus K$  contains 269 pairs.
- Let  $U$  be the set of all directed pairs  $r$ - $g$  such that  $r$  is a regulator in  $L \setminus K$  and  $g$  is a target gene in  $L \setminus K$ . For the example of Section 2,  $L \setminus K$  has 11 unique regulator genes and 99 unique target genes. So there are  $11 \times 99$  or 1089 pairs in  $U$ . Assume further that the linked pairs in  $U$  are precisely those pairs in  $L \setminus K$ , and that all other pairs are unlinked.
- Let  $U \setminus K$  be the set of pairs in  $U$  with any hardcoded edges removed (hardcoded edges may resurface in the unlinked pairs). For the example of Section 2, 17 of the 18 pairs in  $K$  occur in  $U$ , so there are  $1089 - 17 = 1072$  edges in  $U \setminus K$ .

The assessment is done using the contingency table of  $(E \setminus K) \cap (U \setminus K)$  relative to  $U \setminus K$ . For the example of Section 2, the assessment would be done with the 57 of the 465 pairs in  $E \setminus K$  that also belong to  $U \setminus K$ .

A function called `contabs.netBMA` is provided to produce contingency tables from a reference network according the procedure described above. Here we compare the edges produced in Section 2 by `networkBMA` modeling on the yeast data with the reference network `referencePairs` made up of results reported in the literature:

```
> ctables <- contabs.netwBMA( edges.ScanBMA, referencePairs, reg.known,
+                               thresh=c(.5,.75,.9))
> ctables
```

|      | TP | FN  | FP | TN  |
|------|----|-----|----|-----|
| 0.5  | 18 | 251 | 21 | 782 |
| 0.75 | 18 | 251 | 16 | 787 |
| 0.9  | 18 | 251 | 13 | 790 |

Another function called ‘`contabs`’ is provided for computing contingency tables when the true underlying network is known. The `scores` function can be used to obtain common assessment statistics from the contingency tables, including sensitivity, specificity, precision, recall, and false discovery rate among other measures.

```
> scores( ctables, what = c("FDR", "precision", "recall"))
```

|      | FDR       | precision | recall    |
|------|-----------|-----------|-----------|
| 0.5  | 0.5384615 | 0.4615385 | 0.0669145 |
| 0.75 | 0.4705882 | 0.5294118 | 0.0669145 |
| 0.9  | 0.4193548 | 0.5806452 | 0.0669145 |

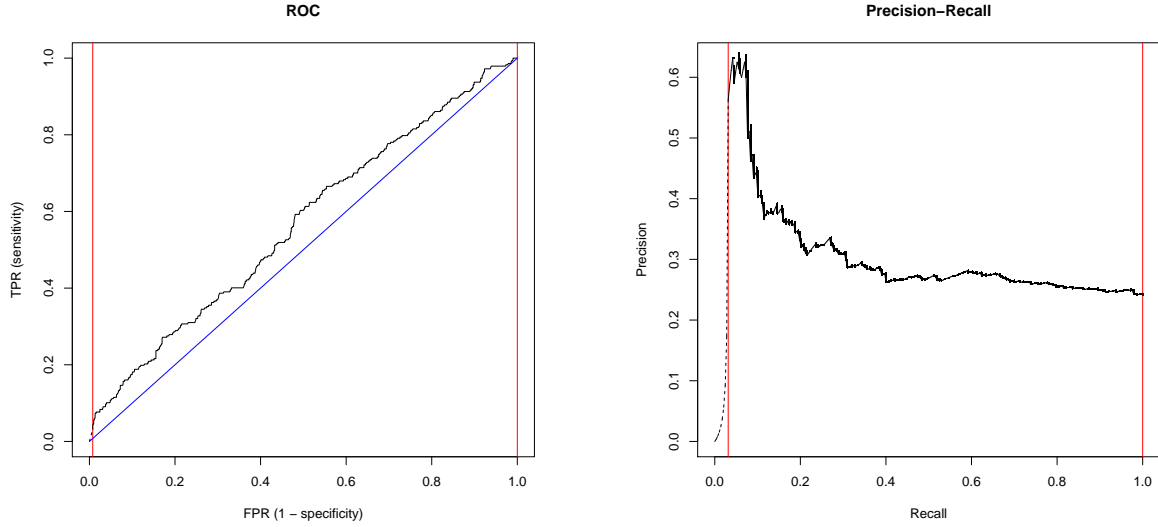


Figure 1: ROC and Precision-Recall curve sectors for a **networkBMA** model of the yeast-rapamycin test data. The black lines delineate the estimated curves. The vertical red lines delineate the range of horizontal values covered by the contingency tables. The dotted black lines are linear interpolants outside this range. The diagonal blue line on the ROC plot indicates the line between (0,0) and (1,1).

Areas under the ROC and Precision-Recall curves covered by contingency tables can also be estimated using functions `roc` and `prc`, with the option to plot the associated curves. The following gives the ROC and Precision-Recall curves associated with the default contingency tables, in which the thresholds are all values for posterior probabilities that appear in `edges.ScanBMA`.

```
> roc( contabs.netwBMA( edges.ScanBMA, referencePairs), plotit = TRUE)
      area    sector    width
0.5649801 0.5637484 0.9911210
> title("ROC")
> prc( contabs.netwBMA( edges.ScanBMA, referencePairs), plotit = TRUE)
      area    sector    width
0.2925174 0.2909923 0.9686411
> title("Precision-Recall")
```

The resulting plots are shown in Figure 1. The output components are as follows:

- **area:** The estimated area under the curve for the horizontal sector ranging from 0 to 1. This should be used with caution when the sector in which the data falls is small.
- **sector:** The estimated area under the horizontal sector covered by the contingency tables.
- **width:** The width of the horizontal sector covered by the contingency tables.

## 4 Linear Modeling for Static Gene Expression Data

`networkBMA` relies on sparse linear modeling via iterative Bayesian model averaging (BMA). BMA addresses uncertainty in model selection, and builds a weighted-average model from plausible models. The resulting model has better overall predictive ability than constituent models, and tends to use few variables from among a larger set. BMA has been iteratively extended to data with more variables than observations (Yeung et al. 2005, 2009, 2011). The `networkBMA` package functions, `ScanBMA` and `iterateBMAIm`, for linear modeling via iterative BMA. We illustrate their use on a static gene expression dataset (without any time points), `brem.data`, to infer the regulators of a particular gene by regressing it on the expression levels of the other genes. Functions `ScanBMA` and `iterateBMAIm` can be applied to each gene so as to infer all edges in the network. For one gene, the procedure is as follows:

```
> gene <- "YNL037C"
> variables <- which(rownames(brem.data) != gene)
> control <- ScanBMAcontrol(OR = 20, useg = TRUE,
+                           gCtrl = gControl(optimize = FALSE, g0 = 20))
> ScanBMAmodel.YNL037C <- ScanBMA(x = t(brem.data[variables,]),
+                                 y = unlist(brem.data[gene,]), control = control)
```

Function `ScanBMAcontrol` facilitates input of BMA control parameters, including `useg` for indicating whether to use Zellner's  $g$ -prior or BIC for model likelihood approximation and `OR` for defining the width of 'Occam's window' for model exclusion. `gCtrl` allows specification of parameters related to the use of Zellner's  $g$ -prior, including whether to use a static  $g$  or optimize  $g$  using an EM algorithm. See the R help documentation for `ScanBMAcontrol` and `gControl` for detailed description of these parameters, and Hoeting et al. (1999) for a tutorial on the underlying BMA paradigm. The estimated posterior probabilities (in percent) for genes that regulate YBL103C can be seen as follows:

```
> ScanBMAmodel.YNL037C$probne0[ScanBMAmodel.YNL037C$probne0 > 0]
```

|           |           |          |           |           |           |           |           |
|-----------|-----------|----------|-----------|-----------|-----------|-----------|-----------|
| YBL103C   | YDL170W   | YDR043C  | YDR216W   | YER040W   | YGL254W   | YJR094C   | YKL062W   |
| 15.81895  | 6.33311   | 17.25494 | 99.30193  | 5.31088   | 2.83664   | 1.58314   | 3.95383   |
| YMR280C   | YOR032C   | YLR276C  | YMR193W   | YHR051W   | YPL111W   | YGL009C   | YPR002W   |
| 2.83050   | 5.24749   | 2.94435  | 1.39853   | 2.58522   | 2.79859   | 3.43629   | 100.00000 |
| YML123C   | YHR136C   | YER158C  | YJL217W   | YKR075C   | YLR258W   | YDR171W   | YIL136W   |
| 3.63262   | 1.38361   | 1.72177  | 1.60902   | 1.21411   | 100.00000 | 100.00000 | 99.72728  |
| YKR056W   | YAL062W   | YCR021C  | YDL110C   | YGR043C   | YGR142W   | YHR018C   | YJL088W   |
| 10.12298  | 100.00000 | 1.09232  | 1.36683   | 1.92756   | 1.74354   | 2.40810   | 1.29696   |
| YJR148W   | YLR009W   | YLR130C  | YLR267W   | YLR327C   | YNL036W   | YNR053C   | YOR302W   |
| 100.00000 | 3.22791   | 2.11767  | 1.29808   | 8.36596   | 1.45602   | 1.32398   | 1.24863   |
| YPL012W   | YIL037C   | YKR009C  | YFR022W   | YPL265W   | YOR100C   | YOR348C   | YHL028W   |
| 84.09918  | 1.23302   | 1.32936  | 100.00000 | 100.00000 | 99.41495  | 98.79002  | 1.40035   |
| YGR067C   | YLL011W   | YPL036W  | YAL054C   | YLR438W   | YIL155C   | YBR054W   | YCL064C   |
| 1.10718   | 1.88925   | 1.05122  | 5.08782   | 3.38030   | 1.36713   | 2.34692   | 100.00000 |
| YDL160C   | YDR384C   | YER067W  | YER150W   | YFL014W   | YHR029C   | YHR087W   | YLR178C   |

|          |           |           |         |          |          |         |          |
|----------|-----------|-----------|---------|----------|----------|---------|----------|
| 1.26625  | 4.70142   | 2.49145   | 2.05581 | 5.76384  | 5.43642  | 1.91634 | 64.98200 |
| YMR169C  | YMR229C   | YNR064C   | YOR206W | YOR388C  | YDR380W  | YPL061W | YJL172W  |
| 9.73588  | 100.00000 | 4.78895   | 2.77098 | 17.46033 | 1.91489  | 1.23917 | 1.45981  |
| YNL194C  | YDR361C   | YHR216W   | YNR046W | YGR183C  | YLL034C  | YDR523C | YOL014W  |
| 11.01615 | 4.27226   | 1.36634   | 1.15586 | 99.73287 | 36.44980 | 7.59316 | 3.15712  |
| YLR174W  | YJL153C   | YGL229C   | YJL063C |          |          |         |          |
| 1.52603  | 100.00000 | 100.00000 | 1.21337 |          |          |         |          |

## 5 Acknowledgements

We would like to thank Dr. Roger Bumgarner for helpful discussion.

*Funding:* AER, KYY and WCY are supported by NIH grant 5R01GM084163. KYY is also supported by 3R01GM084163-05S1. 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.

## References

- [1] R. B. Brem and L. Kruglyak. The landscape of genetic complexity across 5,700 gene expression traits in yeast. *Proceedings of the National Academy of Sciences*, 102(5):1572–1577, 2005.
- [2] R. B. Brem, G. Yvert, R. Clinton, and L. Kruglyak. Genetic dissection of transcriptional regulation in budding yeast. *Science*, 296(5568):752–755, 2002.
- [3] C. Fraley, K. Y. Yeung, and A. E. Raftery. networkBMA: Regression-based network inference using BMA, 2012. R package distributed through Bioconductor.
- [4] P. E. Hodges, A. H. Z. McKee, B. P. Davis, W. E. Payne, and J. I. Garrels. The Yeast Proteome database (YPD): a model for the organization and presentation of genome-wide functional data. *Nucleic Acids Research*, 27(1):69–73, 1999.
- [5] J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky. Bayesian model averging: a tutorial. *Statistical Science*, 14(4):382–401, 1999.
- [6] K. Lo, A. E. Raftery, K. M. Dombek, J. Zhu, E. E. Schadt, R. E. Bumgarner, and K. Y. Yeung. Integrating external biological knowledge in the construction of regulatory networks from time-series expression data. *BMC Systems Biology*, 6:101, 2012.
- [7] H. Parkinson, M. Kapushesky, M. Shojatalab, N. Abeygunawardena, R. Coulson, A. Farne, E. Holloway, N. Kolesnykov, P. Lilja, M. Lukk, R. Mani, T. Rayner, A. Sharma, E. William, U. Sarkans, and A. Brazma. ArrayExpress — a public database of microarray experiments and gene expression profiles. *Nucleic Acids Research*, 35(suppl 1):D747–D750, 2007.
- [8] A. Raftery, J. Hoeting, C. Volinsky, I. Painter, and K. Y. Yeung. BMA: Bayesian model averaging, 2005. R package distributed through CRAN, revised in 2012.



- [9] M. C. Teixeira, P. Monteiro, P. Jain, S. Tenreiro, A. R. Fernandes, N. P. Mira, N. Alenquer, A. T. Freitas, A. L. Oliveira, and I. Sá-Correia. The YEASTRACT database: A tool for the analysis of transcription regulatory associations in *Saccharomyces cerevisiae*. *Nucleic Acids Research*, 34(Issue supplement 1):D446–D451, 2006.
- [10] K. Y. Yeung. iterativeBMA: The iterative bayesian model averaging (BMA) algorithm, 2009. R package distributed through Bioconductor, includes contributions from A. Raftery and I. Painter.
- [11] K. Y. Yeung, R. E. Bumgarner, and A. E. Raftery. Bayesian model averaging: development of an improved multi-class, gene selection and classification tool for microarray data. *Bioinformatics*, 21(10):2394–2402, 2005.
- [12] K. Y. Yeung, K. M. Dombek, K. Lo, J. E. Mittler, J. Zhu, E. E. Schadt, R. E. Bumgarner, and A. E. Raftery. Construction of regulatory networks using expression time-series data of a genotyped population. *Proceedings of the National Academy of Sciences*, 108(48):19436–19441, November 2011.
- [13] J. Zhu and M. Q. Zhang. SCPD: A promoter database of the yeast *Saccharomyces cerevisiae*. *Bioinformatics*, 15(7):607–611, 1999.