

Identifying cycling genes by combining sequence homology and expression data

Yong Lu¹, Roni Rosenfeld¹ and Ziv Bar-Joseph^{1,2,*}

¹School of Computer Science and ²Department of Biology, Carnegie Mellon University
5000 Forbes Ave. Pittsburgh, PA, 15213

ABSTRACT

Motivation: The expression of genes during the cell division process has now been studied in many different species. An important goal of these studies is to identify the set of cycling genes. To date, this was done independently for each of the species studied. Due to noise and other data analysis problems, accurately deriving a set of cycling genes from expression data is a hard problem. This is especially true for some of the multicellular organisms, including humans.

Results: Here we present the first algorithm that combines microarray expression data from multiple species for identifying cycling genes. Our algorithm represents genes from multiple species as nodes in a graph. Edges between genes represent sequence similarity. Starting with the measured expression values for each species we use Belief Propagation to determine a posterior score for genes. This posterior is used to determine a new set of cycling genes for each species.

We applied our algorithm to improve the identification of the set of cell cycle genes in budding yeast and humans. As we show, by incorporating sequence similarity information we were able to obtain a more accurate set of genes compared to methods that rely on expression data alone. Our method was especially successful for the human dataset indicating that it can use a high quality dataset from one species to overcome noise problems in another.

Availability: C implementation is available from the supporting website: <http://www.cs.cmu.edu/~lyongu/pub/cellcycle/>

Contact: zivbj@cs.cmu.edu

1 INTRODUCTION

The cell cycle system has been studied using microarray expression data in several species. These include humans (Whitfield *et al.*, 2002), budding and fission yeast (Spellman *et al.*, 1998; Rustici *et al.*, 2004), plants (Menges *et al.*, 2002) and bacteria (Laub *et al.*, 2000). One of the first questions researchers face when analyzing such experiments is how to identify the set of cycling genes. Many methods have been developed for identifying such genes in a *single* species. These include methods that rely on Fourier transform (Spellman *et al.*, 1998; Wichert *et al.*, 2004), sinusoids (Schliep *et al.*, 2003; Zhao *et al.*, 2001), deconvolution (Bar-Joseph, 2004; Lu *et al.*, 2004) and methods that combine expression amplitude with Fourier analysis (de Lichtenberg *et al.*, 2005). All of the above methods rely on thresholds and other parameters which are not

always easy to determine. Indeed, while these methods have been successful for some species, their success varied depending on the quality of the microarray data and the noise level (Shedden and Cooper, 2002).

The recent expression profiling of the cell cycle system in fission yeast provided a good opportunity for researchers to compare the set of cycling genes in two closely related species (budding and fission yeast). Surprisingly, the results indicated that cell cycle expression is not well conserved among these two species. As Rustici *et al.* (2004) write: “*Our comparisons with budding yeast data revealed a surprisingly small core set of genes that are periodically expressed in both yeasts.*” There could be many reasons for the disagreement between the list of cycling genes in different species. One possibility is that cell cycle expression is not well conserved (though cell cycle function may still be conserved on the post-transcriptional, or protein, level). However, there may be other reasons for this discrepancy. The different computational methods used to determine the set of cycling genes, noise in the data and differences in the quality of the data may result in one list being more accurate than the other. In such cases it may be possible to rely on one species to improve our detection of cycling genes in the other. This process may yield higher quality lists for both species.

In this paper we present a method for combining experiments from multiple species. Our algorithm combines sequence and expression data to identify the set of cycling genes. By considering sequence information we can use homologs to overcome noise and cutoff problems in individual species. By using expression data we can detect *functional* conservation, that is, sets of genes that are not only similar in sequence but also similar in function.

We use probabilistic graphical models, and in particular Markov random fields, to combine these data sources. We represent genes as nodes in the graph, with edges corresponding to sequence similarity as determined by a BLAST score. Each node (gene) is assigned an initial score which is determined by the expression experiment. Starting with this score we propagate information along the edges of the graph until convergence. Thus, if a node with a medium score is connected to a set of nodes with high scores, the information from the neighboring nodes can be used to elevate our belief in the assignment of this node, and vice versa.

Because the algorithm assumes expression conservation it leads to better agreement between cycling genes in different species. In order to test this algorithm it is thus important to show that

*To whom correspondence should be addressed.

this agreement does not come at the expense of a high quality set in either species. To show that our algorithm actually improves the quality of the identified set of cycling genes we tested it using two species for which additional information is available: Budding yeast and humans. As we show, by combining sequence and expression data our algorithm was able to detect a more accurate set of cycling genes in both species when compared to methods that rely on expression data only. While the improvement was mild for the high quality budding yeast expression data, it was much more substantial for the more noisy human cell cycle expression data.

1.1 Related work

Many methods have been suggested to identify the set of cycling genes from one or more expression datasets in a single species. For example, Spellman *et al.* (1998) used Fourier transform to identify cycling genes in budding yeast. Wichert *et al.* (2004) presented statistical methods for identifying periodically expressed genes and applied them (separately) to human and yeast. Lu *et al.* (2004) and Bar-Joseph *et al.* (2004) presented methods for deconvolving yeast expression data in order to improve the identification of cycling genes. de Lichtenberg *et al.* (2005) used scores that look at both, the amplitude of the expression value peak as well as the peak in the Fourier spectrum around the cell cycle period. Unlike the above methods, our method combines information from multiple species using sequence similarity. This allows us to overcome noise and improve the identification of cycling genes.

A number of previous papers combined sequence and expression data to study similarities in expression between different species. For example, Bergmann *et al.* (2004) clustered data from six different species to identify modules of genes that are co-expressed. Stuart *et al.* (2003) identified ‘metagenes’, a group of homolog genes from four different species (one gene from each species), and then used correlation coefficients to link metagenes forming a co-expression network. Our work differs from these papers in several important aspects. First, unlike prior work that relied on clustering to identify groups of co-expressed genes under a wide range of conditions, our approach uses a *classification* framework to achieve a different goal: identifying a set of conserved cycling genes. Second, prior work only looked at pairwise expression similarities, whereas our algorithm utilizes the complete graph topology to propagate information. Finally, previous papers used sequence similarity as a binary value (similar or not). In contrast, our framework uses the extent of this similarity to determine edge weights. The higher the similarity the greater the importance of neighboring genes for determining the cyclic score.

Recently, a number of papers compared the regulatory networks of various species (Sharan *et al.*, 2005). These papers used graph theoretic methods to compare networks across species and identify similar pathways in these species. The focus of these papers and their goals are very different from ours. While we are focused on identifying the set of cycling genes using expression data the above papers relied on the regulatory information in each species. Such information may not be accurately available for all genes and transcription factors in various species. Specifically, the networks they relied on were not systems specific but rather general, and the goal was to extract global and local similarities as opposed to the cell cycle oriented goal in our paper.

A number of papers used belief propagation to combine different biological data sources. These include the physical networks model

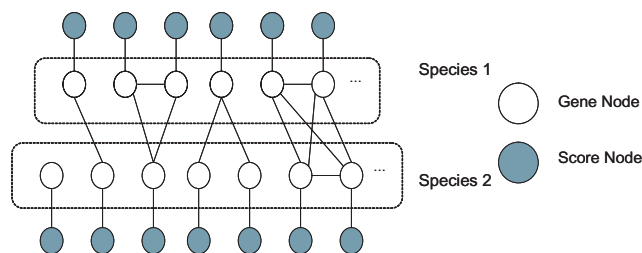


Fig. 1. A graphical model for two species. Dark nodes are score nodes, representing the score derived from such experiments. The lighter nodes are gene nodes. Gene nodes are connected by edges if their sequence is similar.

by Yeang *et al.* (2004) and methods for determining protein functions (Letovsky and Kasif, 2003). These are very different in their goal from our work, and use different types of data. In addition, these papers did not try to combine information from different species, as we do here.

2 MODELING EVOLUTIONARY CONSERVATION USING GRAPHICAL MODELS

We formulate the problem of assigning cyclic status to genes using probabilistic graphical models. In such models, random variables are represented by nodes in a graph and conditional dependencies are represented by edges. The structure of the graph and the conditional dependencies it implies specify a joint probability distribution on the random variables. By taking advantage of the structural relations in the graph, efficient algorithms have been proposed to either learn the parameters of graphical models or do inferences on learned models.

Here we use Markov random fields (MRF) to represent dependencies between genes in different species. Unlike Bayesian networks, MRFs are undirected graphical models, in which dependency among nodes is represented using potential functions. There are two types of nodes in the graph we use for this problem (see Figure 1). The first represents genes and the second represents expression scores from the related cell cycle experiments. Edges between gene nodes correspond to sequence similarity, and carry a weight which depends on that similarity. These edges are used to capture the conditional dependencies of phylogenetically related genes. All edges between a gene node and its corresponding score node have the same weight and correspond to the gene nodes’ potentials.

To generate the edges between potential homologous genes, we run BLAST between all pairs of genes in the two species. We insert an edge between two gene nodes (either belonging to the same species or to two different species) if their BLAST score is higher than a fixed threshold. We use a conservative cutoff such that we are fairly confident that when an edge is added to the graph, the two genes it connects are very likely to be homologous. While we use a cutoff to determine whether we place an edge or not, edges that are present in the graph are weighted based on their BLAST score. The resulting graph comprises of a set of connected components, as demonstrated in the diagram in Figure 1.

To represent the latent status of a gene (whether or not it is a cell cycle gene) we associate a hidden variable C_i with each gene node. $C_i = 1$ means that this gene is cell cycle regulated, otherwise $C_i = 0$.

Based on the definitions above, the joint probability distribution over the random variables C_i of this model is defined as follows (Pearl, 1988)

$$L = \frac{1}{Z} \prod_i \psi_i(C_i) \prod_{i,j} \psi_{ij}(C_i, C_j) \quad (1)$$

where $\psi_i(C_i)$ is the node potential function (derived from the score node), $\psi_{ij}(C_i, C_j)$ is the edge potential function, and Z is the partition function, i.e. the normalization term. Potential functions capture constraints on a single variable or between a pair of dependent variables. For example, if two gene nodes i and j are connected by an edge with a large weight, it is likely that they are functionally related. Thus, the potential function will penalize assignments that are different in the different nodes (e.g., setting C_i to 0 and C_j to 1). Below we discuss the potential function in detail.

2.1 Score distribution

A key to our algorithm is the derivation of an expression score which is consistent across all species used. Once such an expression score has been derived, each score node is assigned the corresponding gene's score, S_i . We assume that S_i is drawn from a mixture distribution. Specifically, we assume two different distributions (for each species): a cell cycle specific distribution, which applies to all genes that are cell cycle regulated, and a null, or background distribution which applies to all other genes.

An important practical issue is to choose the form of the two component distributions of the S_i scores. While the Gaussian distribution has been successfully applied to model expression values, here we are modeling scores that are derived from such values, and not the values themselves. In many cases, such scores are derived by taking the max value of some transformation. Cell cycle score calculation involves taking the maximum peak of the expression time series or the Fourier transform and the resulting distribution often has a heavy tail and is more appropriately modeled as an Extreme Value Distribution (EVD). This heavy tail property is clearly noticeable in the scores assigned to known cycling genes as can be seen in Figure 3.

The EVD is defined using two parameters: location (a) and scale (b). Its PDF is given by:

$$p(x) = \frac{1}{b} e^{-\exp\{\frac{a-x}{b}\}} e^{\frac{a-x}{b}}$$

The location and scale parameters of EVD are similar to the mean and variance parameters of the Gaussian distribution. As in a Gaussian, they control the mode and the spread of the distribution, though they do not necessarily correspond to the mean and variance. Using the EVD mixture model we need to fit four parameters for each species a_0, b_0, a_1, b_1 where

$$\begin{aligned} S_i | C_i = 0 &\sim EVD(a_0, b_0) \\ S_i | C_i = 1 &\sim EVD(a_1, b_1) \end{aligned}$$

The values of these parameters are fitted to the score distributions using an EM-type algorithm. As with any EM algorithm, the initial guess plays an important role in reaching a good local maximum. To initialize the parameters for the null distribution we permute each of the original time series randomly to simulate the expression levels of non cell-cycle genes. Scores are calculated from these artificial expression data, and are subsequently used to estimate the parameters of the null score distribution. To initialize the score for

cell-cycle genes, we compile a list of such genes that appear in the corresponding papers and use the scores of these genes to derive a maximum-likelihood estimate of the parameters.

2.2 Node potential function

The node potential function is defined using Bayes rule as

$$\begin{aligned} \psi_i(C_i) &= Pr(C_i | S_i) \\ &= \frac{Pr(S_i | C_i) Pr(C_i)}{Pr(S_i | C_i = 0) Pr(C_i = 0) + Pr(S_i | C_i = 1) Pr(C_i = 1)} \end{aligned}$$

Using the EVD mixture assumption, the potential function becomes

$$\begin{aligned} \psi_i(0) &= Pr(C_i = 0 | S_i) = \frac{t_{i0}}{t_{i0} + t_{i1}}, \\ \psi_i(1) &= Pr(C_i = 1 | S_i) = \frac{t_{i1}}{t_{i0} + t_{i1}} \end{aligned}$$

where

$$\begin{aligned} t_{i0} &= (1 - P_c) \cdot \frac{1}{b_0} e^{-\exp\{\frac{a_0 - S_i}{b_0}\}} e^{\frac{a_0 - S_i}{b_0}} \\ t_{i1} &= P_c \cdot \frac{1}{b_1} e^{-\exp\{\frac{a_1 - S_i}{b_1}\}} e^{\frac{a_1 - S_i}{b_1}} \end{aligned}$$

and P_c is a prior probability for cycling genes in the species to which i belongs.

In practice, we require $b_0 = b_1$ so that the two score distributions have a similar spread. This guarantees that the posterior score will have the same ranking as the expression scores when there are no edges in the graph.

2.3 Edge potential functions

Our edge potential functions capture the a-priori functional similarity between gene pairs. This is based on our assumption regarding evolutionary conservation of gene functions, namely, that genes that are highly similar in sequence are likely to be similar in function. We use BLAST (Altschul *et al.*, 1997) to determine sequence similarity. As mentioned earlier, we do not transform these BLAST scores into binary features. Rather, we use the similarity score to determine the edge potential which penalizes contradictory assignments. The penalty is proportional to how close the two genes' sequences are.

For each query sequence, the BLASTALL program returns an E-value and a bit score S . The relation between them is $E = mn2^{-S}$ where m is the length of the query sequence and n is the length of the genome of the second species. Note that bit scores are not "symmetric" as they depend on the total genome length. To overcome this, and generate a single similarity score for pairs of genes we set the weight on edge (i, j) to

$$w_{ij} = \frac{1}{2} (b_{ij} + b_{ji})$$

where b_{ij} is the BLAST bit score of gene i against gene j . Using $w_{i,j}$ we define the edge potential as

$$\psi_{ij}(C_i, C_j) = 2^{-\lambda w_{ij} (C_i - C_j)^2}.$$

This potential function penalizes assignments that do not agree between connected nodes. λ is an externally specified parameter that controls the impact of edge potentials relative to the node potentials.

3 LEARNING THE PARAMETERS OF OUR MODEL

The model parameters we need to learn are the score distribution parameters for each species. We learn the score distribution parameters (a_0, b_0, a_1, b_1) in an iterative manner using an EM-style algorithm. We start with an informative guess for the score parameters, as mentioned above. Based on the score distributions we determine a posterior assignment to nodes using belief propagation, as we discuss below. Following convergence of the belief propagation algorithm we use the (soft) label assignments to update the score distribution parameters. We then repeat these steps by performing belief propagation again based on the updated score distributions and so forth until both the label assignment and score distribution parameters do not change anymore.

3.1 Iterative step 1: inference by belief propagation

To infer the node status variables C_i , we need to compute the marginal posterior label distribution on each gene node. This posterior is hard to compute directly because of the intractable normalization term Z in Formula (1). Fortunately, for these types of graphical models, we can use a standard belief propagation algorithm for inference avoiding the direct calculation of the Z term (Pearl, 1988). Note that our graph is loopy and thus the belief propagation algorithm is not guaranteed to converge to a global maximum. Still, as was shown in Yedidia *et al.* (2003) in practice these algorithms achieve good results in loopy networks as well.

The belief propagation algorithm consists of two steps: ‘Message passing’, where each node sends its current belief to all its neighbors, and ‘belief update’, where nodes update their belief based on the messages received. In our case the messages depend on the node’s expression score and the belief of genes that are similar in sequence. The algorithm is summarized below.

- (1) ‘Message passing’. The messages sent by node i to node j about its belief in an assignment of 1 to j is:

$$m_{i,j}(1) \leftarrow \sum_{k=0,1} \left((\psi_i(k) \psi_{ij}(k, 1) \prod_{n \in N(i) \setminus j} m_{n,i}(k)) \right)$$

Where $N(i)$ is the set of neighbors of node i in the graph. Intuitively, this message informs j about i ’s agreement with an assignment of 1 to j . In order to determine this, i takes into account its own belief (from its score node), the strength of the edge between i and j and the belief of i ’s neighbors about the right assignment to i . For the belief in a 0 assignment we simply replace every 1 with 0 in the above equation. Note that the weighting parameter λ is already incorporated into the edge potential function and so it is incorporated into the message as well.

- (2) ‘Belief update’. The belief of i in an assignment of 1 is computed by setting:

$$b_i(1) = (1/\nu) \psi_i(1) \prod_{j \in N(i)} m_{j,i}(1)$$

where ν is a normalization constant to make beliefs sum to 1. As can be seen, i ’s belief depends on both its original score and the messages it received from its neighbors about what they ‘believe’ should be assigned to i .

Table 1. Algorithm for combining microarray expression data from multiple species

Input

1. For each gene, expression score S_i
2. Graph structure (edge weights)

Output:

- For each gene its posterior cycling status, C_i

Initialization:

- For each species compute estimates for a_0 , a_1 and b using permutation analysis and original lists

Iterate until convergence:

1. Carry out Belief Propagation to determine a posterior C_i for each gene
2. Use the computed posterior to recompute the EVD parameters for the score distribution in each species

3.2 Iterative step 2: updating the score distribution

Using the belief computed in the inference step, we update the score distribution parameters. Our goal is to maximize the auxiliary function $Q(\Theta, \Theta^{(g)})$, which is defined as the expected log likelihood of the complete data over the observed scores given the parameters $\Theta^{(g)} = (a_0^{(g)}, a_1^{(g)}, b^{(g)})$ at the g ’th iteration.

We were unable to find a reference for deriving update rules for the EVD mixture distribution. We have thus derived these ourselves. In general, to derive an update rule for this distribution we need to simplify the Q function and separate the parameters into two terms which can be maximized independently. If we require that $b_0 = b_1$, then for each species we have three parameters: two location parameters a_0 and a_1 and one scale parameter b . We can find the location parameters that maximize Q easily if we know b , but there is no close form solution for b . However, we can use numerical methods to solve for b . The final update rules for each species are as follows

$$a_l^{(g+1)} = \frac{1}{\beta} \log \frac{\sum_{i=1}^N P_{il}}{\sum_{i=1}^N e^{-\beta S_i} P_{il}}, \quad l = 0, 1$$

$$b^{(g+1)} = \frac{1}{\beta}$$

where N is the number of genes in that species, P_{il} represents $p(C_i = l | S_i, \Theta^g)$, $l = 0, 1$, and β is the root of the equation:

$$\frac{1}{\beta} = \frac{\sum_{l=\{0,1\}} \sum_{i=1}^N S_i P_{il}}{\sum_{l=\{0,1\}} \sum_{i=1}^N P_{il}} - \sum_{l=\{0,1\}} \left[\sum_{i=1}^N P_{il} \frac{\sum_{i=1}^N e^{-\beta S_i} S_i P_{il}}{\sum_{i=1}^N e^{-\beta S_i} P_{il}} \right] / \sum_{l=\{0,1\}} \sum_{i=1}^N P_{il} \quad (2)$$

Equation (2) can be solved using linear line search since the reasonable range of β is not large. Note that the Newton-Raphson method does not work here, because the solution is very close to the local extrema of the function. See Appendix for more details.

We can also extend our model to use the Generalized Extreme Value Distribution which in some cases gives better results. For details please refer to our supporting website (Lu *et al.*, 2006).

Our algorithm is summarized in Table 1 above.

4 RESULTS

We tested our algorithm on simulated and real biological data. For the biological species we selected budding yeast and humans. While budding and fission yeast are closer from the evolutionary standpoint, there is less complementary information for the set of cycling genes in fission yeast. In contrast, many of the human cell cycle genes have been extensively studied leading to good annotation databases for these genes. This makes it easier to evaluate a new list of cycling human genes when compared with a list for cycling fission yeast genes. Another advantage of looking at human instead of fission yeast is that it indicates that even if the two species are relatively far, they can still benefit from a joint analysis of their cell cycle expression experiments.

4.1 Simulated data

To test our model using simulated data we first generated the graph structure from the two species as discussed before. We then generated labels (i.e. cycling or not) for nodes in the graph using a Gibbs sampler method that took into account previously assigned neighboring nodes when assigning labels to individual nodes. See the supporting website (Lu *et al.*, 2006) for complete details on the label assignment.

After generating the labels we assigned scores to nodes. We used two (overlapping) score distributions, one for the nodes with $C_i = 1$ and the other for those with $C_i = 0$. In all experiments we used a fixed distribution for one species. However, each experiment used a different distribution for the second species. These distributions varied in their separability, ranging from highly separable to highly overlapping (see Figure 2). We have next hidden the node assignments, and used our algorithm to infer these assignments. We repeated this process 10 times for each set of score distributions.

Figure 2 presents the results of two of these experiments. As can be seen, by relying on the graph structure we were able to improve the recovery of the true label assignments when compared to label assignments that are based on a cutoff of the score alone. As the separation between the two distributions became smaller the difference between the two methods became more apparent. For the less separable distributions our algorithm performed much better than the score only method by relying more heavily on the distribution of the other species.

These results indicate that under the evolutionary assumptions we stated in the introduction, our algorithm can improve the assignment of cycling genes and correctly recover more such genes.

4.2 Cell cycle expression data

To date, cell cycle expression was measured in more than six species. As mentioned above, the two most studied species are budding yeast and humans. Both provide access to a number of different validation sets, and are thus useful for comparison of our algorithm and score based methods.

We downloaded expression data from the corresponding websites for the budding yeast (Spellman *et al.*, 1998) and human (Whitfield *et al.*, 2002) cell cycle papers. All protein sequences for genes in these species were downloaded from the NCBI ftp server (<http://ftp.ncbi.nlm.nih.gov>). We used Blastall (Altschul *et al.*, 1997) to score all pairs of genes in both species.

For this data we tested our algorithm using an Intel Pentium 4 PC with single 2.40GHz CPU. It typically took less than 6 minutes to converge.

Expression Scores: As mentioned earlier, it is important to use the same method to derive scores for genes in different species. We derived such scores based on the observed expression values. As was recently noted for yeast by de Lichtenberg *et al.* (2005), scores that look at both amplitude of the expression value peak as well as the peak in the spectrum around the cell cycle period seem to provide the best results for identifying genes using expression data only. We thus applied a similar method to extract such scores for all genes in both species (see the supporting website Lu *et al.*, (2006) for details). To validate this method we compared our results to the benchmark provided by de Lichtenberg *et al.*, (2005) and determined that our results for budding yeast were comparable to the best method presented in their paper. The results below use this scoring method. However, using the Whitfield *et al.* (2002) scoring method did not change the results. See the supporting website (Lu *et al.*, 2006) for more details.

Comparison sets: As far as we know, this is the first method to combine sequence and expression data for the task of identifying cycling genes. In order to compare our results to previous methods we use two different alternative lists. The first list is the list of cycling genes (in each species) based on the expression score alone. As mentioned in the introduction, this is the method used by previous approaches. We have also compared our results to a more naive method for combining expression and sequence. Unlike our probabilistic approach, this naive method first computes ranking independently for each species based on expression score alone. Next, we identify conserved genes in both species and compute a joint ranking based on the average ranking for the orthologs in each species. While we do not claim that this method is ideal, it can at least serve as a baseline for evaluating the more sophisticated algorithm we present in this paper.

Identifying cycling human genes: To test the success of our algorithm for the task of identifying cycling human genes we used the GO human annotations. Of the 7254 human genes in the dataset we used, 498 were annotated by GO as cycling. We first ranked human genes using expression scores and the naive method mentioned above. Next, we ranked them using the posterior score computed by our algorithm.

Figure 3 (left) presents the precision recall curve for GO annotated cycling genes for the top ranked 1000 human genes. Based on the analysis in the original paper (Whitfield *et al.*, 2002), roughly 1000 genes are determined to be cycling, which is why we focus on the top 1000. As can be seen, all three methods perform substantially better than a random ordering (dashed-dotted curve). Comparing our method with a score based method we see that while at the very high expression score (bottom left) we do slightly worse, overall, and in particular for lower scores our algorithm provides results that are superior to score based methods. Specifically, for the top 1000 genes our algorithm was able to recover 23% more genes (135 vs. 110) when compared to both, the score only method and the naive method for combining sequence and expression data.

Note that while we relied on the GO list for this analysis, it is not complete. It is possible that there are many cycling genes which are not on that list. Thus, the recall rate is probably much higher

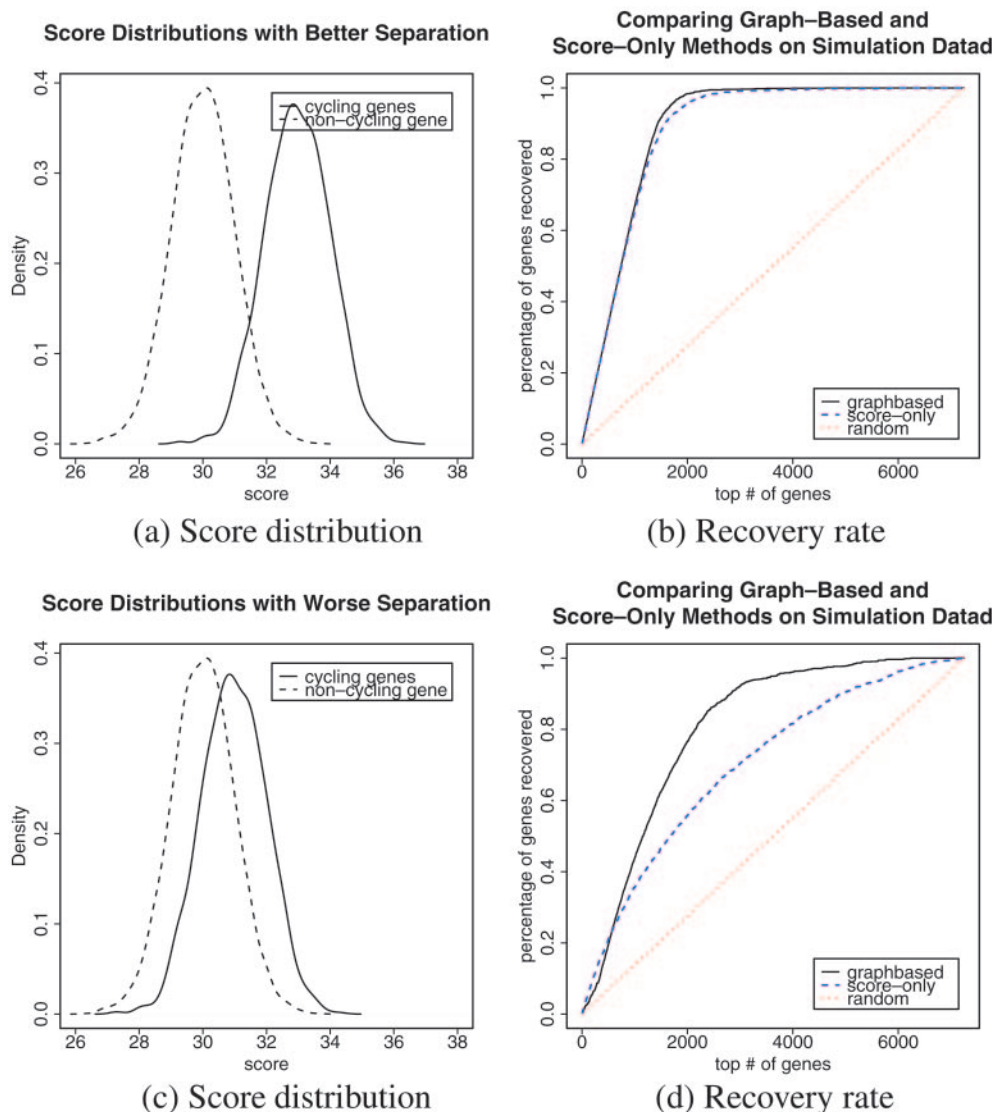


Fig. 2. Simulation results. 20% of the nodes were labeled with 1 and the rest were labeled with 0. (a) Score distribution and (b) Recovery rate for a well separated distribution. Both score based (dashed line) and graph based (solid line) methods were able to correctly recover the node assignments. (c) Score distribution and (d) Recovery rate for an overlapping score distribution. Note that while our graph based method can still achieve good precision and recall the score based method does significantly worse, especially for the higher recall rates (above 40%).

than the one we report here. Figure 3 (right) presents the expression score distribution of genes annotated as cycling in GO and genes that do not belong to this category. Note that there is substantial overlap between the two distributions making it hard for a score only method to identify a large set of cycling human genes. In contrast, our graph based method was able to partially overcome this problem by relying on the graph neighborhoods as we discuss below. Another issue is the possible homology bias of GO annotations. To account for this, we repeated the validation procedure using a smaller set of GO annotated human cell cycle genes. Specifically, we removed the 256 human genes that are annotated in GO as "cell cycle" based on homology evidence. Even with this reduced set of GO cycling genes our method outperforms the score based method by a similar margin. See website for details.

To further explore the differences between score based and graph based methods we examined the differences in cell cycle assignments between the two. We generated two lists. The first contained genes that appear in the top 1000 using our method but were not in the top 1000 of the score based method and the second contained genes in the top 1000 of the scoring method but not in our method. To test which of these list is more relevant we used GO to analyze both lists. On the supporting website (Lu *et al.*, 2006) we present a number of figures comparing the GO enrichment p-values of both lists. As we show there, the majority of cell cycle related categories are more enriched for genes in the list derived based on our method when compared with the score based list.

Identifying cycling yeast genes We have used a dataset for protein-DNA binding (Lee *et al.*, 2002) to compare our budding yeast results

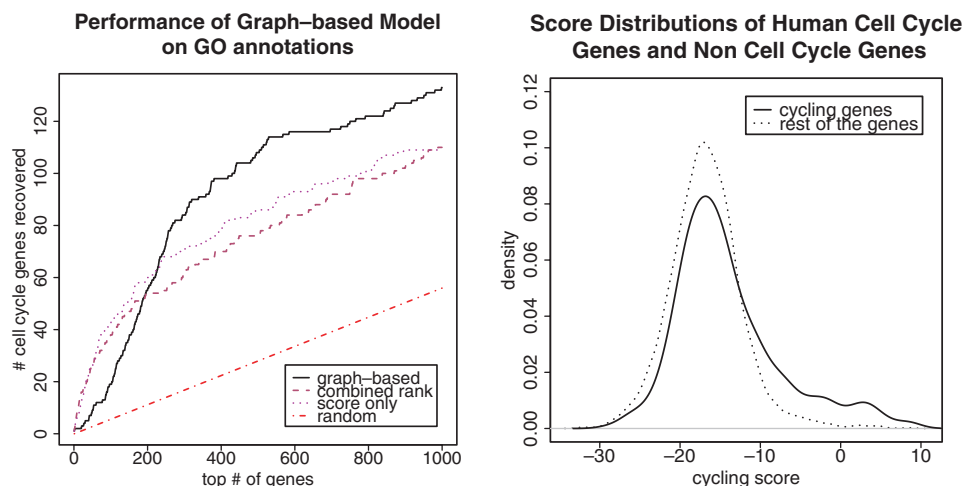


Fig. 3. Left: Identification of Human Cell Cycle Genes. The Y axis is the number of GO annotated human cell cycle genes in the top 1000 genes with highest posteriors. Our method (solid line) performs better than the score only method (dashed line) and the naive method for combining sequence and expression data (dotted line). Specifically for lower score thresholds our method achieves an improvement of over 20% over both other methods in terms of the number of accurately recovered cell cycle genes. Right: Score distribution for genes annotated as cycling in GO and the rest of the genes. As can be seen, these two distributions significantly overlap, making it hard to infer cyclic status from the expression score alone. The score distribution of the cell cycle genes has a heavy tail, and looks more like an Extreme Value Distribution than a normal distribution.

with the original list of Spellman *et al.* which was based on score alone. We extracted the binding information (p -value < 0.005 for the nine transcription factors that have been previously shown to play key roles in regulating cell cycle progression (Simon *et al.*, 2001). We found 2.5% more interactions between these nine TFs and the top 800 genes on our list when compared with the Spellman list (621 vs. 606, note that a gene could be counted multiple times if more than one TF interacts with it). We also tested a stricter version of the binding data (p -value < 0.001). As with the higher p -value, our method still resulted in slightly more interactions (477 vs. 474) when compared to the score based list. While these improvements are far less dramatic than the results presented for the human data above, it still implies that our method can improve cell cycle assignment even for high quality datasets, like the yeast cell cycle expression data (Wichert *et al.*, 2004).

Graph neighborhoods To further explore how our method helps in correct assignment of cell cycle status we have plotted two of the subgraphs in our graph. The shape of the nodes in each subgraph represents the species, and the different shades of node color represent the cycling expression score of the gene. Darker shades represent higher *expression* scores, and the darkest shade means that the expression score is within the top 1000 for human and top 800 for yeast. The first subgraph (Figure 4) contains members of the Rho family of genes in yeast and humans. These genes are involved in cell wall formation which is an integral part of the cell cycle system. Specifically, the cell wall integrity signaling pathway is controlled by Rho1 (Levin, 2005). Based on its expression score Rho1 was not in the top 800 yeast genes. However, since its expression score is high enough, and since its local neighborhood is all assigned cyclic status, our algorithm assigns a posterior score that is at the top 800 for yeast genes allowing us to correctly recover this gene.

Why expression scores are not sufficient? Expression value, especially in time series experiments which usually do not contain repeats for individual time points, are very noisy. To determine why

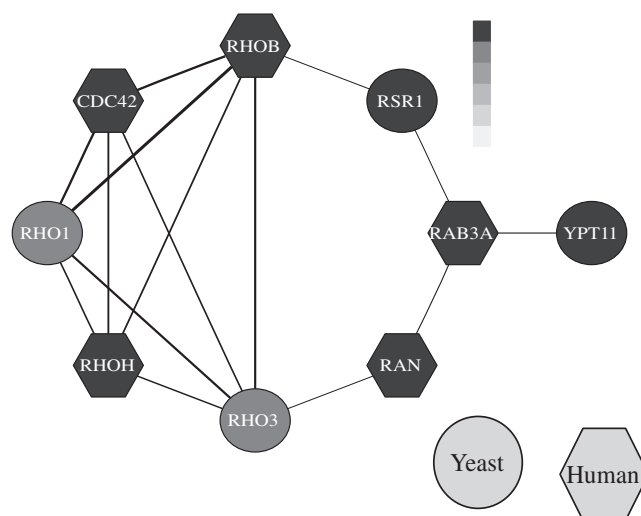


Fig. 4. Cluster containing the yeast cell wall gene Rho1. Node shades correspond to expression derived scores. Circles correspond to yeast genes and hexagons to human genes. Rho1 is not in the top 800 genes based on its score, but was identified by our algorithm because of its neighborhood.

our algorithm is able to correctly identify genes that cannot be detected using their expression score we looked at a number of genes that received high posterior scores and low expression scores.

One such gene is Mcm3, shown in Figure 5. Human Mcm3 is essential for the initiation of DNA replication and also participates in a checkpoint that ensures DNA replication is initiated once per cell cycle (Madine *et al.*, 1995; Takei and Tsujimoto, 1998). On the top of Figure 5 we plot the graph neighborhood of Mcm3. As can be seen, it contains many known cycling genes from both species. On the bottom we plot the expression of Mcm3 in three different human cell cycle datasets (each done using a different

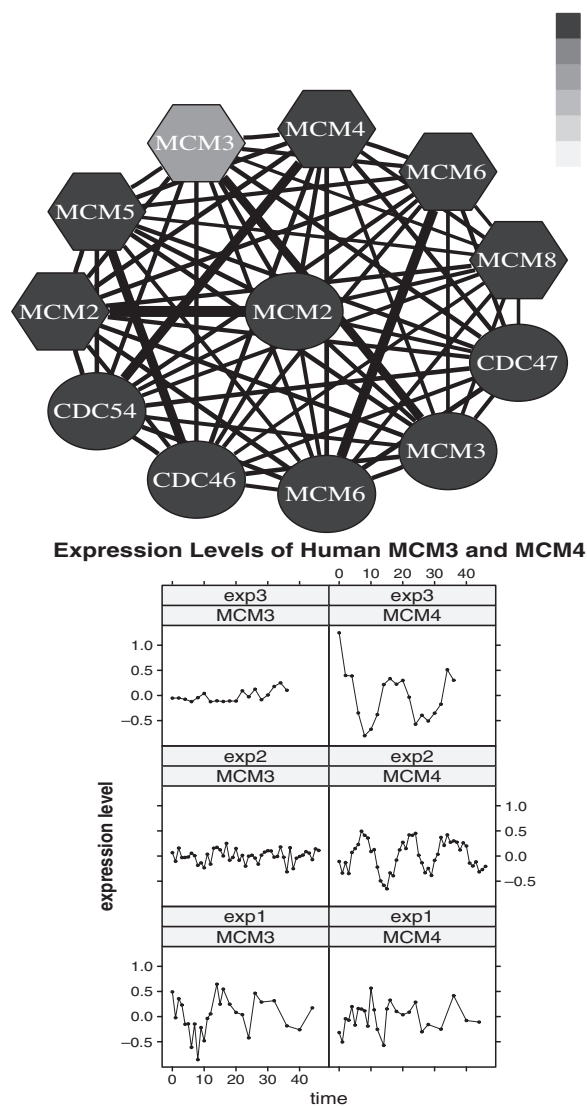


Fig. 5. Top: Gene cluster containing the human gene Mcm3, which is essential for the initiation of DNA replication. Bottom: plots showing the expression time series for both Mcm3 and Mcm4. Mcm4 scored in the top 15% but Mcm3 did not. Our algorithm was able to recover both genes.

arrest method). As can be seen, in at least one of these conditions Mcm3 seems to be cycling (bottom left). However, either because its expression levels are low in the other experiments or because of other experimental problems, it does not seem to be cycling in the other conditions. Using expression data alone, we would not assign a cyclic status to this gene. However, because of its medium expression score and its strong neighborhood score, our algorithm was able to correctly determine that it is a cycling human gene.

5 CONCLUSIONS AND FUTURE WORK

Many researchers have used gene expression experiments to study biological systems in various species. We presented an algorithm that combines information from studies in multiple species for the task of identifying cycling genes. Our algorithm constructs a graph where nodes represent genes and edges represent sequence

similarity. We then use belief propagation to update the status of genes based on their graph neighborhood.

We applied our algorithm to combine cell cycle expression data from budding yeast and humans. Using our approach we were able to recover a more accurate set of cycling human genes when compared to the score based methods that have been used in the past. We have also shown that by looking at the neighborhood extracted from the graph we can infer properties that cannot be determined using expression alone.

While this paper focuses on cell cycle analysis, our algorithm is general and can work with any expression data as long as an expression score can be extracted from that data. An obvious future direction is to apply it to other biological systems that have been studied in multiple species such as immune response and circadian rhythm. Another direction is to combine regulatory data with the sequence data we currently use to infer sets of genes that are conserved in terms of regulation, sequence and function across multiple species.

ACKNOWLEDGEMENTS

This work was partially supported by NSF CAREER award 0448453 to ZBJ and by a tobacco settlement grant from the Pennsylvania department of health.

REFERENCES

- Altschul,F.S., Madden,T.L., Schaffer,A.A., Zhang,J., Zhang,Z., Miller,W. and Lipman,D.J. (1997) Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**(17), 3389–3402.
- Bar-Joseph,Z., Farkash,S., Gifford,D., Simon,I. and Rosenfeld,R. (2004) Deconvolving cell cycle expression data with complementary information. *Bioinformatics*, **20** Suppl 1, 123–130.
- Bar-Joseph,Z. (2004) Analyzing time series gene expression data. *Bioinformatics*, **20**(16), 2493–2503.
- Bergmann,S., Ihmels,J. and Barkai,N. (2004) Similarities and differences in genome-wide expression data of six organisms. *PLoS Biol.*, **2**(1), e9.
- Bilmes,J.A. (1998) A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical Report TR-97-021, UC Berkeley.
- de Lichtenberg,U., Jensen,L.J., Fausboll,A., Jensen,T.S., Bork,P. and Brunak,S. (2005) Comparison of computational methods for the identification of cell cycle-regulated genes. *Bioinformatics*, **21**, 1164–1171.
- Laub,M., McAdams,H., Feldblyum,T., Fraser,C. and Shapiro,L. (2000) Global analysis of the genetic network controlling a bacterial cell cycle. *Science*, **290**(5499), 2144–8.
- Lee,T., Rinaldi,N., Robert,F., Odom,D. and Bar-Joseph,Z. *et al.* (2002) Transcriptional regulatory networks in *saccharomyces cerevisiae*. *Science*, **798**, 799–804.
- Letovsky,S. and Kasif,S. (2003) Predicting protein function from protein/protein interaction data: a probabilistic approach. *Bioinformatics*, **19**(Suppl 1), 197–204.
- Levin,D. (2005) Cell wall integrity signaling in *saccharomyces cerevisiae*. *Microbiol Mol Biol Rev.*, **69**, 262–91.
- Lu,X., Zhang,W., Qin,Z., Kwast,K. and Liu,J. (2004) Statistical resynchronization and bayesian detection of periodically expressed genes. *Nucl. Acids. Res.*, **32**, 447–455.
- Lu,Y., Rosenfeld,R. and Bar-Joseph,Z. (2006) Supporting website. <http://www.cs.cmu.edu/~lyongu/pub/cellcycle/>.
- Madine,M.A., Khoo,C.Y., Mills,A.D. and Laskey,R.A. (1995) Mcm3 complex required for cell cycle regulation of dna replication in vertebrate cells. *Nature*, **375**, 6530.
- Menges,M., Hennig,L., Gruissem,W. and Murray,J. (2002) Cell cycle regulated gene expression in *arabidopsis*. *J Biol Chem.*, **277**(44), 41987–42002.
- Pearl,J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.
- Rustici,G., Mata,J., Kivinen,K., Lio,P., Penkett,C., Burns,J., Hayles,G., Brazma,A., Nurse,P. and Bahler,J. (2004) Periodic gene expression program of the fission yeast cell cycle. *Nat. Genet.*, **36**(8), 809–17.
- Schliep,A., Schonhuth,A. and Steinhoff,C. (2003) Using hidden markov models to analyze gene expression time course data. *Bioinformatics*, **19**, i264–i272.

Sharan,R., Suthram,S., Kelley,R., Kuhn,T., McCuine,S., Uetz,P., Sittler,T., Karp,R. and Ideker,T. (2005) Conserved patterns of protein interaction in multiple species. *Proc Natl Acad Sci USA*, **102**(6), 1974–9.

Shedden,K. and Cooper,S. (2002) Analysis of cell-cycle-specific gene expression in human cells as determined by microarrays and double-thymidine block synchronization. *PNAS*, **99**(7), 4379–84.

Simon,L., Barnett,J., Hannett,N., Harbison,C., Rinaldi,N., Volkert,T., Wyrick,J., Zeitlinger,J., Gifford,D., Jaakkola,T. and Young,R. (2001) Serial regulation of transcriptional regulators in the yeast cell cycle. *Cell*, **106**, 697–708.

Spellman,P.T., Sherlock,G., Zhang,M., Iyer,V., Anders,K., Eisen,M.B., Brown,P.O., Botstein,D. and Futcher,B. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast *saccharomyces cerevisia* by microarray hybridization. *Mol. Biol. of the Cell*, **9**, 3273–3297.

Stuart,J.M., Segal,E., Koller,D. and Kim,S.K. (2003) A gene-coexpression network for global discovery of conserved genetic modules. *Science*, **302**(5643), 249–55.

Takei,Y. and Tsujimoto,G. (1998) Identification of a novel mcm3-associated protein that facilitates mcm3 nuclear localization. *J Biol Chem*, **273**, 22177–22180.

Whitfield,M., Sherlock,G., Saldanha,A., Murray,J., Ball,C. et al. (2002) Identification of genes periodically expressed in the human cell cycle and their expression in tumors. *Mol. Biol. Cell*, **13**(6), 1977–2000.

Wichert,S., Fokianos,K. and Strimmer,K. (2004) Identifying periodically expressed transcripts in microarray time series data. *Bioinformatics*, **20**, 5–20.

Yeang,C., Ideker,T. and Jaakkola,T. (2004) Physical network models. *J Comput Biol*, **11**(2–3), 243–62.

Yedidia,J.S., Freeman,W.T. and Weiss,Y. (2003) Understanding belief propagation and its generalizations. *Exploring Artificial Intelligence in the New Millennium*, pp. 236–239.

Zhao,L.P., Prentice,R. and Breeden,L. (2001) Statistical modeling of large microarray data sets to identify stimulus-response profiles. *PNAS*, **98**, 5631–5636.

6 APPENDIX

6.1 Derivation of update rules for EVD mixture model

Here are some facts of the Type I EVD, or the Gumbel distribution. The CDF and PDF of the EVD are

- CDF

$$F(x) = \exp\left\{-\exp\left[-\left(\frac{x-a}{b}\right)\right]\right\}, \quad -\infty < x < \infty$$

- PDF

$$p(x) = \frac{1}{b} \exp\left\{-\exp\left[-\left(\frac{x-a}{b}\right)\right]\right\} \exp\left\{-\left(\frac{x-a}{b}\right)\right\}$$

In the EM algorithm, we define the Q function to be $Q(\Theta, \Theta^{(i-1)}) = E[\log p(\mathcal{X}, \mathcal{Y} | \Theta) | \mathcal{X}, \Theta^{(i-1)}]$ where \mathcal{X} is the observed data, i.e. expression scores, and \mathcal{Y} represents the hidden variables, i.e. the cycling status of the genes. In each E-step, we evaluate the above expectation, and in each M-step we maximize this expectation. For mixture models, the expectation can be written as (Bilmes, 1998)

$$Q(\Theta, \Theta^{(i-1)}) = \sum_{l=1}^M \sum_{i=1}^N \log(\alpha_l) p(l | x_i, \Theta^g) + \sum_{l=1}^M \sum_{i=1}^N \log(p_l(x_i | \theta_l)) p(l | x_i, \Theta^g)$$

where x_i is the i -th observed data point (i.e. the score S_i for the i -th gene), α_l is the mixing coefficient of the l -th component, p_l is the PDF for the l -th component, and $p(l | x_i, \Theta^g)$ from now on denoted as

P_{il} for simplicity, is the probability x_i being generated by the l -th component, given the parameters Θ^g .

To maximize this expression, we can maximize the two terms independently. Using Lagrange multipliers, we solve for α_l that maximizes the first term, and get

$$\alpha_l = \frac{1}{N} \sum_{i=1}^N P_{il}$$

The maximization of the second term depends on the PDF of the component distributions. For the EVD mixture model, the second term becomes

$$B = \sum_{l=1}^M \sum_{i=1}^N P_{il} \log(p_l(x_i | \theta_l)) = \sum_{l=1}^M \sum_{i=1}^N P_{il} \left[-\log b_l - \frac{x_i - a_l}{b_l} - \exp\left\{-\frac{x_i - a_l}{b_l}\right\} \right]$$

Note that we also require the two components to have the same scale parameter, so we can drop the subscript and denote b_l as b . Now we maximize B by want to solve

$$\begin{aligned} \frac{\partial B}{\partial b} &= \sum_{l=1}^M \sum_{i=1}^N [-\beta + \beta^2(x_i - a_l) - \beta^2(x_i - a_l) \exp\{-\beta(x_i - a_l)\}] P_{il} \\ &= -\beta \sum_{l=1}^M \sum_{i=1}^N P_{il} + \beta^2 \sum_{l=1}^M \sum_{i=1}^N x_i P_{il} - \beta^2 \sum_{l=1}^M e^{\beta a_l} \sum_{i=1}^N e^{-\beta x_i} x_i P_{il} \\ &= 0 \end{aligned}$$

where $\beta = 1/b$. The above equation can be transformed to

$$\begin{aligned} \frac{1}{\beta} &= \frac{\sum_{l=1}^M \sum_{i=1}^N x_i P_{il}}{\sum_{l=1}^M \sum_{i=1}^N P_{il}} - \sum_{l=1}^M \left[\frac{\sum_{i=1}^N P_{il} \frac{\sum_{i=1}^N e^{-\beta x_i} x_i P_{il}}{\sum_{i=1}^N e^{-\beta x_i} P_{il}}}{\sum_{i=1}^N P_{il}} \right] / \sum_{l=1}^M \sum_{i=1}^N p(l | x_i, \Theta^g) \end{aligned}$$

Define

$$f(\beta) = \frac{1}{\beta} - \frac{\sum_{l=1}^M \sum_{i=1}^N x_i P_{il}}{\sum_{l=1}^M \sum_{i=1}^N P_{il}} + \sum_{l=1}^M \left[\frac{\sum_{i=1}^N P_{il} \frac{\sum_{i=1}^N e^{-\beta x_i} x_i P_{il}}{\sum_{i=1}^N e^{-\beta x_i} P_{il}}}{\sum_{i=1}^N P_{il}} \right] / \sum_{l=1}^M \sum_{i=1}^N P_{il}$$

and the β we are looking for is the root of $f(\beta) = 0$. In this case, since the root is near a local extremum ($\lim_{x \rightarrow +0} f(x) \rightarrow +\infty$), the Newton-Raphson method can fail. Fortunately, we can simply use a root bracketing algorithm to search for it, because we don't expect the variances of the distribution to be too big.