BIOINFORMATICS ORIGINAL PAPER
Vol. 21 no. 10 2005, pages 2488–2495
doi:10.1093/bioinformatics/bti339

Systems biology

Selective integration of multiple biological data for supervised network inference

Tsuyoshi Kato[1,*], Koji Tsuda[1,2] and Kiyoshi Asai[1,3]

[1]Computational Biology Research Center, National Institute of Advanced Industrial Science and Technology (AIST), 2-43 Aomi Koto-ku, Tokyo, Japan, [2]Max Planck Institute for Biological Cybernetics, Spemannstrasse 38, 72076 Tübingen, Germany and [3]Graduate School of Frontier Sciences, University of Tokyo, 5-1-5, Kashiwanoha, Kashiwa, Chiba 277–8562, Japan

Received on November 30, 2004; revised on February 15, 2005; accepted on February 17, 2005
Advance Access publication February 22, 2005

ABSTRACT

Motivation: Inferring networks of proteins from biological data is a central issue of computational biology. Most network inference methods, including Bayesian networks, take unsupervised approaches in which the network is totally unknown in the beginning, and all the edges have to be predicted. A more realistic supervised framework, proposed recently, assumes that a substantial part of the network is known. We propose a new kernel-based method for supervised graph inference based on multiple types of biological datasets such as gene expression, phylogenetic profiles and amino acid sequences. Notably, our method assigns a weight to each type of dataset and thereby selects informative ones. Data selection is useful for reducing data collection costs. For example, when a similar network inference problem must be solved for other organisms, the dataset excluded by our algorithm need not be collected.

Results: First, we formulate supervised network inference as a kernel matrix completion problem, where the inference of edges boils down to estimation of missing entries of a kernel matrix. Then, an expectation–maximization algorithm is proposed to simultaneously infer the missing entries of the kernel matrix and the weights of multiple datasets. By introducing the weights, we can integrate multiple datasets selectively and thereby exclude irrelevant and noisy datasets. Our approach is favorably tested in two biological networks: a metabolic network and a protein interaction network.

Availability: Software is available on request.

Contact: kato-tsuyoshi@aist.go.jp

Supplementary information: A supplementary report including mathematical details is available at www.cbrc.jp/∼kato/faem/faem.html

1 INTRODUCTION

Recent advances in generating comprehensive assays have clarified that most biological functions are realized through the cooperation of multiple proteins. To understand complicated relations of many proteins, it is handy to use a network representation (i.e. protein network) in which a protein corresponds to a node and the relation of two proteins is described as an edge. There are several types of networks whose edges have different meanings. This paper specifically addresses metabolic networks (Kanehisa *et al*., 2004) and networks of physical interactions (von Mering *et al*., 2002).

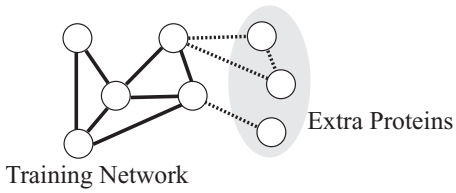


Fig. 1. The supervised network prediction; for a substantial number of proteins, the network is assumed to be known (training network). Our task is to identify the hidden edges involving extra proteins (shown as broken lines) using multiple datasets, such as the gene expression data.

Several statistical methods have been proposed to infer such networks, but most of them are unsupervised: they estimate the edges solely from a dataset [Friedman (2004); Saito *et al*. (2003)]. In contrast, the supervised approach (Yamanishi *et al*., 2004) assumes that a part of the network is known. When the network has ℓ proteins, the presence or absence of edges is completely known for the first $n < \ell$ proteins (Fig. 1). This assumption is becoming more realistic as high confidence networks have become increasingly available (von Mering *et al*., 2002). The remaining $m := \ell - n$ proteins are extra proteins. Our task is to predict the edges including extra proteins from datasets.

Properties of proteins are characterized from many aspects. It is for this reason, that proteins are usually represented by multiple types of data such as gene expression, amino acid sequences and phylogenetic profiles. It is essential to exploit all available datasets to achieve reliable prediction of networks. Nevertheless, to control data collection costs, it is also important to select informative data types. Once informative data types are identified, one need not collect unnecessary dataset when solving similar network inference problem for other sets of proteins or for other organisms.

This paper presents a new network inference method with automatic data selection. Following Yamanishi *et al*. (2004), proteins are represented as a 'kernel matrix' whose (i, j) element describes a similarity between two proteins i and j. With a kernel matrix, one can construct a network from the individual values. Edges are assigned

*To whom correspondence should be addressed.

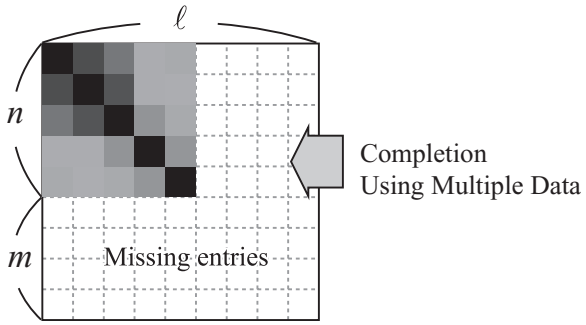


Fig. 2. Illustration of kernel matrix completion. Kernel values for the first n proteins are derived from the training network. Our task is to estimate the missing parts of the matrix using multiple types of data related to proteins.

to the protein pairs whose kernel values are above a certain threshold. One can control the number of edges by changing the threshold. The network inference problem is a huge combinatorial problem because many 0/1 binary variables must be determined to describe the edges. The kernel matrix representation lends reasonable relaxation to this difficult problem. Subsequently, the network inference problem boils down to the problem of inferring kernel values (i.e. similarity) among the proteins.

Our method is based on kernel matrix completion (Tsuda *et al*., 2003), which works as follows. First, the kernel matrix among n proteins in the training network is obtained by diffusion kernels (Kondor and Lafferty, 2002). We call this the incomplete kernel matrix because kernel values involving m extra proteins are unknown (Fig. 2). Next, each biological dataset is converted to a kernel matrix. Typically, we have $n_{\mathrm{K}}(\geq 2)$ different biological datasets, which we call auxiliary data. The RBF kernel and the linear kernel are common choices for vectors such as gene expressions. For sequences, trees or graphs, one can use kernel derivation methods for structured objects (Schölkopf *et al*., 2004). To predict the edges involving extra nodes, we must estimate the missing parts of the incomplete kernel matrix using a number of kernel matrices derived from auxiliary data (Fig. 2). We then introduce a weight parameter to each kernel matrix to select informative datasets. Consequently, our expectation–maximization (EM) algorithm estimates the missing part and the weights at the same time. The auxiliary matrices assigned with low weights are considered as unnecessary. In a related work, Lanckriet *et al*. (2004) addressed a similar kernel selection problem for protein function prediction, but that study does not assume any missing entry.

The salient technical difficulty in dealing with multiple kernel matrices is that the data weights are added as new unknown variables. When the EM algorithm is formulated in a straightforward manner, the optimization problem of the M-step is not convex, introducing a local minima problem into a step. We avoid this situation by adding extra hidden variables and reformulating the EM algorithm. In our algorithm, the M-step is convex and can be solved efficiently through one-dimensional line search.

Our method is applied to metabolic network and protein interaction networks in held-out tests. The accuracy of predicting edges was better than kernel CCA (KCCA). Furthermore, our method was able to exclude noisy datasets without losing accuracy.

2 KERNEL MATRIX REPRESENTATION OF A NETWORK

As mentioned, the network inference problem is a large combinatorial problem. Typically, one must minimize a non-convex loss function (e.g. negative marginal likelihood of a Bayesian network) over many binary variables. This optimization problem is very difficult to solve because of numerous local minima. Further more, as a result of these minima, a reasonable network may not be found when the initial network is not good enough.

The kernel matrix representation simplifies the problem radically (Yamanishi *et al*., 2004). To describe the network with ℓ nodes, we take an $\ell \times \ell$ symmetric positive definite kernel matrix Q and a threshold δ. The kernel matrix is assumed to be normalized, namely $Q_{ii} = 1$ for any i. We place edges between the node pairs whose kernel values are $>\delta$. More precisely, the set of edges is described as

$$E = \{(i, j) \mid Q_{ij} \geq \delta, 1 \leq i < j \leq \ell\}.$$

The restriction that the matrix Q is positive definite is helpful for developing algorithms for network inference because Q can be regarded as a Gram matrix in a vector space [Yamanishi *et al*. (2004)] or a covariance matrix of a Gaussian distribution (Section 3).

Our task is to obtain Q from the training network and multiple datasets. Let us decompose Q as

$$Q = \begin{bmatrix} K_I & Q_{vh} \\ Q_{vh}^{\top} & Q_{hh} \end{bmatrix}. \quad (1)$$

For the first n nodes, K_I is determined solely from the training network using the diffusion kernel (Kondor and Lafferty, 2002). Let A denote the adjacency matrix of the training network and let D denote the diagonal matrix of degrees of nodes (i.e. the number of edges connected to a node). Under those assumptions, the graph Laplacian matrix $L := D - A$ is normalized so that all the diagonal elements are one:

$$L_{ij} \leftarrow \frac{L_{ij}}{\sqrt{L_{ii} L_{jj}}}.$$

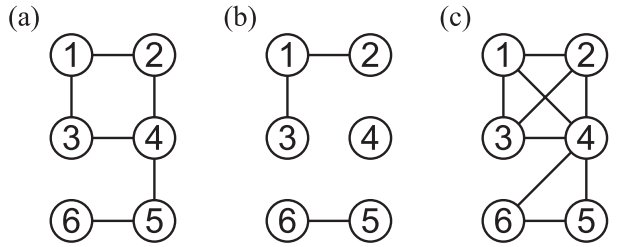
The diffusion kernel is defined as

$$K = \exp(-\beta L),$$

where exp is a matrix exponential operation and β is a parameter to control the degree of diffusion (Fig. 3). Diagonal elements of the diffusion kernel are not normalized to 1. Therefore, we normalize the matrix K again and obtain the resulting matrix K_I. When elements in K_I larger than a threshold are picked up, the training network, if not exactly recovered, is approximately recovered. If the network should be recovered exactly, a constrained version of diffusion kernels can be used with additional computational cost (Tsuda and Noble, 2004). However, we do not require the exact recovery here because our primary objective is to predict the edges involving the extra proteins (Q_{vh} and Q_{hh}). The closest problem is the estimation of Q_{vh} and Q_{hh} using multiple kernel matrices $P_1, \ldots, P_{n_{\mathrm{K}}} \in \Re^{\ell \times \ell}$ derived from different types of datasets.

3 KERNEL MATRIX COMPLETION FROM SINGLE DATASET

In this section, we first consider the case in which we have only one auxiliary matrix P. In the next section, we extend the algorithm to deal with multiple matrices.



(d)

$$\begin{bmatrix} 1.000 & 0.460 & 0.460 & 0.175 & 0.023 & 0.004 \\ 0.460 & 1.000 & 0.183 & 0.385 & 0.073 & 0.017 \\ 0.460 & 0.183 & 1.000 & 0.385 & 0.073 & 0.017 \\ 0.175 & 0.385 & 0.385 & 1.000 & 0.366 & 0.124 \\ 0.023 & 0.073 & 0.073 & 0.366 & 1.000 & 0.603 \\ 0.004 & 0.017 & 0.017 & 0.124 & 0.603 & 1.000 \end{bmatrix}$$

Fig. 3. The diffusion kernel gives similarities among nodes on an undirected graph. The 6×6 matrix **(d)** represents the kernel values of the graph with six nodes **(a)** using $\beta = 1.0$. The values between nodes that are close to each other are larger, and vice versa. The adjacency matrix is approximately recovered by selecting elements in the kernel matrix larger than some threshold. For instance, graph **(b)** is reconstructed from the kernel matrix with threshold 0.4, whereas graph **(c)** is reconstructed with threshold 0.1. In this graph, threshold 0.2 reproduces graph (a) perfectly, but such is generally not the case.

To pose the problem as one of statistical inference, let us consider an ℓ-dimensional random variable comprising two parts $\mathbf{x} = (\mathbf{v}^\top, \mathbf{h}^\top)^\top, \mathbf{v} \in \Re^n, \mathbf{h} \in \Re^m$, and two Gaussian distributions $p(\mathbf{x})$ and $q(\mathbf{x})$ defined on $\mathbf{x}$. Distributions p and q correspond to kernel matrices P and Q, respectively. The mean of $p = 0$. Therefore, $E_p[\mathbf{x}] = 0$. The covariance is set to P, i.e. $E_p[\mathbf{x}\mathbf{x}^\top] = P$. The mean of q is also 0 and the covariance for the first part is known as $E_q[\mathbf{v}\mathbf{v}^\top] = K_I$. Our task is to estimate the covariances involving the second part $\mathbf{h}$, namely $Q_{vh} := E_q[\mathbf{v}\mathbf{h}^\top]$ and $Q_{hh} := E_q[\mathbf{h}\mathbf{h}^\top]$.

The relationship between $\mathbf{v}$ and $\mathbf{h}$ is learnt from p by deriving conditional distribution $p(\mathbf{h}|\mathbf{v})$. Then, the joint distribution of q is estimated as $\hat{q}(\mathbf{v}, \mathbf{h}) := p(\mathbf{h}|\mathbf{v})q(\mathbf{v})$. Finally, the covariance matrices are computed from $\hat{q}$ as follows:

$$Q_{vh} = K_I P_{vv}^{-1} P_{vh}, \tag{2}$$

$$Q_{hh} = P_{hh} - P_{vh}^\top P_{vv}^{-1} P_{vh} + P_{vh}^\top P_{vv}^{-1} K_I P_{vv}^{-1} P_{vh} \tag{3}$$

where P_{vv}, P_{vh} and P_{hh} denote the partition matrices of P:

$$P = \begin{bmatrix} P_{vv} & P_{vh} \\ P_{vh}^\top & P_{hh} \end{bmatrix}.$$

The equivalent algorithm has been used successfully in different contexts by Kin *et al*. (2004) and Tsuda *et al*. (2003). To compute the inverse matrices appearing in Equations (2) and (3) in practice, P must be regular. When P is singular or close to singular, one has to regularize P as $P' = P + \sigma^2 I$.

To further clarify this process, it is meaningful to view this algorithm in terms of information geometry (Amari and Nagaoka, 2000). A central concept in information geometry is Kullback–Leibler (KL) divergence, which is a distance measure between two distributions. Distance between two zero mean Gaussian distributions with covariances P and Q is described as

$$D[Q, P] = \tfrac{1}{2} \operatorname{tr}(P^{-1}Q) + \tfrac{1}{2} \log\det P - \tfrac{1}{2} \log\det Q - \tfrac{1}{2}\ell.$$

The covariance matrices (2) and (3) are represented simply as the optimal solution of the following problem (Tsuda *et al*., 2003):

$$\min_{Q_{vh}, Q_{hh}} D[Q, P].$$

Hence Q_{vh} and Q_{hh} are determined such that Q matches P in terms of the KL divergence.

4 SELECTIVE INTEGRATION OF MULTIPLE DATASETS

When n_{K} kinds of datasets are available, one has n_{K} auxiliary kernel matrices $P_1, \ldots, P_{n_{\mathrm{K}}} \in \mathcal{R}^{\ell \times \ell}$. We take the weighted combination of these kernel matrices. The combined kernel matrix with weights $\mathbf{b} = \{b_1, \ldots, b_{n_{\mathrm{K}}}\}$, $\sum_{k=1}^{n_{\mathrm{K}}} b_k = 1$, involving a regularization term described as

$$P(\mathbf{b}) = \sum_{k=1}^{n_{\mathrm{K}}} b_k P_k + \sigma^2 I. \tag{4}$$

The weights b_k for matrices are estimated together with missing parts Q_{vh} and Q_{hh} by minimizing the KL divergence,

$$\min_{Q_{vh}, Q_{hh}, \mathbf{b}} D[Q, P(\mathbf{b})]. \tag{5}$$

Unfortunately, this optimization problem is not convex, so only local minima are obtainable. Nevertheless, the local minima obstacle can be alleviated using the EM algorithm (Dempster *et al*., 1977). The aim of the solution method is to minimize the objective function by alternating solutions of two partial convex problems. For purposes of illustration, the EM algorithms are applied successfully to learning with Gaussian mixture models and hidden Markov models. When we apply the EM algorithm to our problem, we have the following two steps:

- E-step: Fixing $\mathbf{b}$, minimize $D(Q, P)$ with respect to Q_{vh} and Q_{vv} and
- M-step: Fixing Q, minimize $D(Q, P)$ with respect to $\mathbf{b}$.

However, one technical problem arising here is that the optimization problem of the M-step is not convex. We propose to solve this problem by adding extra hidden variables and reformulating the EM algorithm.

The optimization problem posed in the M-step is called structural covariance matrix estimation. Several methods have been proposed (Burg *et al*., 1982). Among them, the most familiar method would be the one using Malley (1994), which also employs extra hidden variables in a different manner. Its computational cost is much higher because it involves inversion of a huge matrix of size $O(\ell^2) \times O(\ell^2)$.

4.1 Extended covariance matrix

We extend the random vector as $\mathbf{c} = (\mathbf{v}^\top, \mathbf{h}^\top, \mathbf{z}^\top)^\top$, where additional variables are denoted as $\mathbf{z} \in \Re^{n_{\mathrm{K}}\ell}$, and define a new Gaussian distribution $r(\mathbf{c}|\mathbf{b})$ whose mean is zero and covariance is $R(\mathbf{b})$. Instead of

Equation (5), we solve the following:

$$\min_{Q_{vh},Q_{hh},Q_{xz},Q_{zz},\mathbf{b}} D[\tilde{Q}, R(\mathbf{b})], \tag{6}$$

where $\tilde{Q}$ is the extended version of the covariance Q, i.e.

$$\tilde{Q} = \begin{bmatrix} Q & Q_{xz} \\ Q_{xz}^\top & Q_{zz} \end{bmatrix}. \tag{7}$$

We design $R(\mathbf{b})$ such that optimal solutions of Equation (6) for Q_{vh}, Q_{hh} and $\mathbf{b}$ coincide with those of the original problem. Furthermore, in the EM algorithm derived for Equation (6), both E- and M-steps are convex optimization problems.

The definition of $R(\mathbf{b})$ requires the Cholesky decomposition of each correlation matrix by $P_k = \Lambda_k \Lambda_k^\top$. Defining

$$\Lambda = [\Lambda_1, \ldots, \Lambda_{n_\mathrm{K}}] \in \Re^{\ell \times \ell n_\mathrm{K}} \tag{8}$$

and

$$B = \mathrm{diag}\{b_1, \ldots, b_{n_\mathrm{K}}\} \otimes I_\ell, \tag{9}$$

we get

$$P(\mathbf{b}) = \Lambda B \Lambda^\top + \sigma^2 I_\ell, \tag{10}$$

where $\otimes$ denotes the Kronecker product.

We define $\mathbf{z}$ as an $n_\mathrm{K}\ell$-dimensional random vector. The extended Gaussian distribution is defined as $r(\mathbf{c}|\mathbf{b}) = r(\mathbf{x}|\mathbf{z})r(\mathbf{z}|\mathbf{b})$, where $r(\mathbf{z}|\mathbf{b}) \sim \mathcal{N}(0, B)$ and $r(\mathbf{x}|\mathbf{z}) \sim \mathcal{N}(\Lambda\mathbf{z}, \sigma^2 I)$. Then, one can show that the covariance matrix of r that is limited to $\mathbf{x}$, $R_{xx}(\mathbf{b})$, coincides with P in Equation (10). The optimal solutions of Equation (6) coincide with those of the original problem (see the Supplementary data for proof).

4.2 EM algorithm

In M-step, we find optimal $\mathbf{b}$ to minimize $D[\tilde{Q}, R(\mathbf{b})]$ using fixed $\tilde{Q}$. Ignoring the terms irrelevant to $\mathbf{b}$, the KL divergence is written as

$$D(\tilde{Q}, P(\mathbf{b})) = \tfrac{1}{2}\,\mathrm{tr}\left(B^{-1}Q_{zz}\right) + \tfrac{1}{2}\log\det(B) + \mathrm{const.} \tag{11}$$

Defining $q_k = (1/\ell)\sum_{j=(k-1)\ell+1}^{k\ell}\left[Q_{zz}\right]_{jj}$, the optimization problem is rewritten as

$$\min_{\mathbf{b}} \sum_{k=1}^{n_\mathrm{K}} \frac{q_k}{b_k} + \log b_k, \quad \sum_{k=1}^{n_\mathrm{K}} b_k = 1.$$

Employing the Lagrange multiplier α, this convex problem is rewritten as

$$\max_{\alpha}\min_{\mathbf{b}} \sum_{k=1}^{n_\mathrm{K}} \frac{q_k}{b_k} + \log b_k + \alpha\left(\sum_{k=1}^{n_\mathrm{K}} b_k - 1\right). \tag{12}$$

The minimization problem inside can be solved as

$$b_k = \frac{-1 + \sqrt{1 + 4\alpha q_k}}{2\alpha} \quad k = 1, \ldots, n_\mathrm{K}. \tag{13}$$

Substituting Equation (13) in Equation (12), we obtain a concave maximization problem with respect to one parameter α, which can be solved easily by a line search algorithm. Thereby, the optimal $\mathbf{b}$ can be obtained by substituting the optimal α in Equation (13).

In the E-step, we find submatrices of $\tilde{Q}$ to minimize the divergence using fixed $R(\mathbf{b})$. It is well known that the optimal solutions can be obtained by conditional expectation (Amari and Nagaoka, 2000). We estimate the joint distribution as $\tilde{q}(\mathbf{c}) = r(\mathbf{z}|\mathbf{x},\mathbf{b})r(\mathbf{h}|\mathbf{v},\mathbf{b})q(\mathbf{v})$, and then derive its covariance matrix as $\tilde{Q}$. Here, the posterior distribution of $\mathbf{z}$ is written as $r(\mathbf{z}|\mathbf{x},\mathbf{b}) \sim \mathcal{N}(\mathbf{m}_z, V_z)$, where

$$V_z = B - B\Lambda^\top(\Lambda B \Lambda^\top + \sigma^2 I)^{-1}\Lambda B \tag{14}$$

and

$$\mathbf{m}_z = \sigma^{-2} V_z \Lambda^\top \mathbf{x}.$$

Estimation of Q, the submatrix for variable $\mathbf{x}$, can be done as Equations (2) and (3) by simply replacing the submatrices of P with those of $R(\mathbf{b})$. Thereby, the submatrix Q_{zz} is estimated as

$$Q_{zz} = V_z + \frac{1}{\sigma^4} V_z \Lambda^\top Q \Lambda V_z. \tag{15}$$

Calculation of the other submatrices is unnecessary because they are not used in the M-step. Moreover, only diagonal elements of Q_{zz} are required in Equation (11), which are computed as

$$[Q_{zz}]_{ii} = [B]_{ii} - ([B]_{ii})^2 \mathbf{l}_i^\top \left(P_0 + \sigma^2 I\right)^{-1} \mathbf{l}_i + \sigma^{-4}\mathbf{g}_i^\top Q \mathbf{g}_i,$$

where $\mathbf{l}_i \in \Re^\ell$ is the i-th column vector of Λ, and $\mathbf{g}_i \in \Re^\ell$ is the i-th column vector of matrix $G \equiv \Lambda V_z$ given by

$$G = \sum_{k=1}^{n_k} b_k \Lambda_k + P_0\left(P_0 + \sigma^2 I\right)^{-1}\left(\sum_{k=1}^{n_\mathrm{K}} b_k \Lambda_k\right),$$

where $P_0 = \sum_{k=1}^{n_\mathrm{K}} b_k \Lambda_k \Lambda_k^\top$.

4.3 Computational complexity

The memory complexity of our algorithm is $O(\ell^2 n_\mathrm{K})$ due to matrices Λ and G. In terms of time complexity, the computation in the E-step is dominant, where the computation of the diagonals of Q_{zz} takes $O(\ell^3 n_\mathrm{K})$.

5 EXPERIMENTS

We apply our algorithm to two kinds of protein networks in yeast. One is a metabolic network produced from the KEGG/PATHWAY database (Kanehisa *et al*., 2004). This network, which contains 769 proteins and 3702 undirected edges, is identical to the one used in Yamanishi *et al*. (2004). In addition, we used the protein interaction network provided by von Mering *et al*. (2002). This interaction network is produced from the results of various biological experiments. Each edge is rated with a confidence level: high, middle or low. We only used high confidence edges because they are supported by multiple experiments. By removal of proteins without any edge, we obtained a network of 984 proteins and 2438 edges.

The edges of the two networks have fundamentally different meanings. In our metabolic network, all proteins are enzymes. An edge is established if two enzymes catalyze successive reactions in any metabolic pathway (Yamanishi *et al*., 2004), whereas the edge in the interaction network indicates the physical interaction of two proteins. Actually, the two networks are quite different: they have 188 proteins

in common, and there are 311 and 159 edges among these proteins in metabolic and interaction networks, respectively. Nevertheless, the number of common edges is only 41. Given the same auxiliary data, no unsupervised method can perform well for both networks, though such a method might be good for one of them. One aim of this experiment is to show how well our supervised method is adaptable to different kinds of networks.

Another point is to investigate the ability of selecting informative data. Here we employed the following four datasets: gene expression (exp), protein interaction data derived from yeast two hybrid (y2h) experiments only, protein localization (loc) and phylogenetic profiles (phy) (Supplementary data). Since it is not inherently clear which one is informative for network prediction, we included several decoy data generated from 10-dimensional Gaussian white noise (rnd) to be excluded as irrelevant data. The range of this artificial noise does not affect prediction results since all kernel matrices are normalized.

Gene expression data have 157D vectors [77 experiments in Spellman *et al*. (1998) and 80 experiments in Eisen *et al*. (1998)]. They are transformed into an RBF kernel matrix with radius 5. The yeast two-hybrid network is a summation of two assays provided from Ito *et al*. (2001) and Uetz *et al*. (2000); its diffusion kernel is computed with $\beta = 1$. The localization data (Huh *et al*., 2003) are 23D vectors with binary values. Each value indicates whether or not the protein is at the corresponding location. Similarly, phy derived from the KEGG database gives 145D binary vectors. The linear kernel is used for both data.

We are also interested in comparing the performance of our algorithm with KCCA (Yamanishi *et al*., 2004). The KCCA constructs the kernel matrix Q through greedy extraction of relatively low-dimensional features from the training network and auxiliary data. Their method can solve the same problem, but it does not have a mechanism to select kernel matrices. It might be possible to extend the KCCA to allow kernel selection, but it is beyond the scope of this paper.

Accuracy of edge prediction is measured by a 10-fold cross-validation. The proteins are divided randomly into 10 subgroups. In each iteration, one subgroup is taken as extra proteins; the network that is restricted to the remaining nine subgroups is used as the training network. After applying the EM algorithm, the estimated entries Q_{vh} and Q_{hh} are used as scores for predicting edges. The predicted edges at different thresholds are compared with the true edges of the subgroup. We employ two measures to summarize the performance: the sensitivity at 95% specificity and the receiver operating characteristic (ROC) score. Sensitivity is defined as the ratio of true positive edges that are correctly identified whereas specificity is the ratio of true negative edges that are correctly identified. The ROC score is the area under the ROC curve, which plots the ratio of true positives against the ratio of false positives for different possible thresholds (Gribskov and Robinson, 1996). We repeat the same procedure for all 10 subgroups and report averaged scores.

For computing diffusion kernels, we set $\beta = 1.0$ for the metabolic network and $\beta = 3.0$ for the protein interaction network. If we pick up elements in the diffusion kernel matrices K_I larger than a threshold, the training networks (Fig. 1) are recovered almost perfectly: at 95% specificity, the sensitivity is 99.29% for the metabolic network and 100% for the protein interaction network. Since we included four decoy data, the number of kernel matrices was eight and the EM algorithm always started from equal weighting (0.125 for all matrices). The regularization parameter is fixed as $\sigma^2 = 0.1$.

Experimental results are shown in Figure 4. The corresponding numerical scores are listed in Tables 1 and 2. The ROC scores are as high as 0.827 and 0.929 for the metabolic and interaction networks, respectively, indicating that our EM algorithm can adapt well to different kinds of networks by learning from the training networks. As a reference, we also evaluated the weighted sum of auxiliary matrices (P_{vv} and P_{vh}) as scores for edge prediction, but the prediction accuracy was much lower than by our method. This result suggests the difficulty of inferring networks in unsupervised settings. We have also shown scores for individual auxiliary data. In the metabolic network, the data combination did not help boost accuracy to a great extent, but a substantial gain was shown in the protein interaction network.

Regarding data selection, our algorithm successfully excluded the four decoy data. Furthermore, the yeast-two-hybrid data ('y2h') got a very small weightage in the metabolic network. The result seems correct because the accuracy of edge prediction using only this data was very poor.

However, assigned weights do not always reflect prediction performance of individual dataset. It is sometimes true that a dataset, which performs poorly when alone, has important complementary information, which is not found in the other datasets. Such a dataset should have a large weightage in principle. Therefore, one cannot easily conclude that a dataset is needless even if it has a relatively low ROC score when used alone. In fact, in the metabolic network, the weight for loc is larger than exp, though exp is better in terms of individual performance.

5.1 Robustness

In the experiments with four decoy datasets, no significant performance changes are found between weighted and uniform combination. For illustrating the advantage of our algorithm clearly, we show the case where more decoy datasets are used. We plotted the sensitivity scores of our EM algorithm against the number of decoy datasets (Fig. 5). The score decreases as the number of decoys increases. Without kernel selection, however, a considerably poor performance is observed.

5.2 Comparison with KCCA

We have also reported the results of KCCA. Since KCCA cannot determine weights by itself, we used weights obtained by our algorithm. For the metabolic network, the parameters of KCCA are set according to Yamanishi *et al*. (2004): the number of features is $d = 40$ and both regularization parameters are $\lambda_1 = \lambda_2 = 0.1$, where λ_1 is the regularization parameter for the training network and λ_2 is the one for auxiliary data. We performed a three-dimensional grid search to determine the best parameter values for the protein interaction network. To be precise, the parameters are tuned such that the sensitivity score is maximized for the unweighted combination of four kernels, exp, y2h, loc and phy. Consequently, the best parameter values were $d = 150$, $\lambda_1 = 1.0$ and $\lambda_2 = 0.01$. For both networks and all weight settings, scores of KCCA were slightly worse than those obtained by our method but the difference was not enormous. This result shows that the performance of our algorithm is at a competitive level.

5.3 Train–extra edges and extra–extra edges

The edges to be predicted can be classified into two categories, namely those between the training network and extra proteins

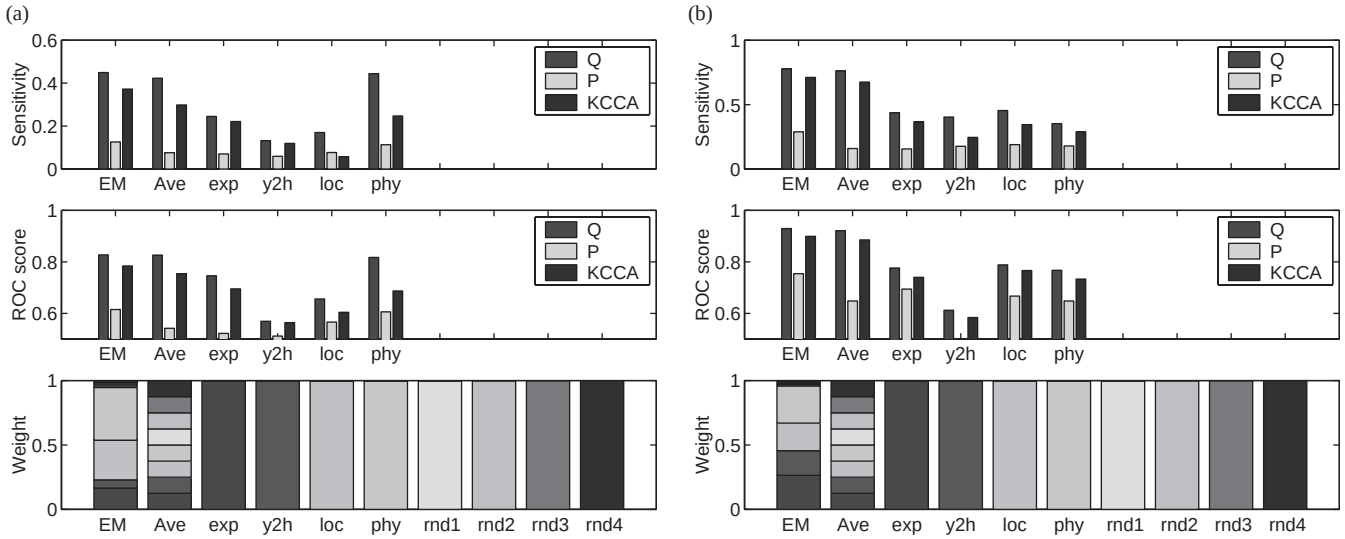


Fig. 4. Accuracy of network prediction for (**a**) metabolic and (**b**) protein interaction networks. Three plots are shown for each network. The top plots illustrate the sensitivity of different prediction algorithms at 95% specificity. The middle plots restate the same results using ROC scores (i.e. the area under ROC curves). Results obtained using our algorithm are shown as blue bars ('Q') in the leftmost column ('EM'). Green bars ('P') show the accuracy based on the matrix P (see the text). Red bars show the results of KCCA using the weights determined by our algorithm. The other columns correspond to results when the weights are prefixed. In column Ave, all kernel matrices are mixed with the same weight. Results using each individual kernel matrix are also shown in columns exp,y2h,loc and phy. Weights that are obtained automatically are shown in the EM column in the bottom plots, while other columns show prefixed weights.

Table 1. Prediction accuracy for the metabolic network

Weight								Sensitivity			ROC score		
exp	y2h	loc	phy	rnd1	rnd2	rnd3	rnd4	*Q*	*P*	KCCA	*Q*	*P*	KCCA
0.168	0.065	0.292	0.421	0.014	0.014	0.013	0.014	0.430	0.125	0.370	0.816	0.616	0.782
0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.413	0.099	0.255	0.813	0.576	0.728
0.168	0.063	0.328	0.441	0	0	0	0	0.436	0.126	0.392	0.827	0.615	0.792
0.250	0.250	0.250	0.250	0	0	0	0	0.425	0.119	0.422	0.828	0.609	0.800
1	0	0	0	0	0	0	0	0.245	0.070	0.221	0.746	0.522	0.695
0	1	0	0	0	0	0	0	0.132	0.059	0.119	0.569	0.512	0.564
0	0	1	0	0	0	0	0	0.170	0.077	0.057	0.656	0.566	0.604
0	0	0	1	0	0	0	0	0.444	0.113	0.247	0.817	0.606	0.687

The first row shows the weights of kernel matrices determined by the EM algorithm, along with the corresponding sensitivity scores and ROC scores. The other rows show results obtained when the weights are prefixed.

Table 2. Prediction accuracy for the protein interaction network

Weight								Sensitivity			ROC score		
exp	y2h	loc	phy	rnd1	rnd2	rnd3	rnd4	*Q*	*P*	KCCA	*Q*	*P*	KCCA
0.264	0.192	0.213	0.289	0.011	0.010	0.011	0.011	0.777	0.288	0.711	0.929	0.754	0.899
0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.762	0.159	0.675	0.921	0.648	0.885
0.279	0.198	0.219	0.304	0	0	0	0	0.796	0.288	0.737	0.939	0.756	0.911
0.250	0.250	0.250	0.250	0	0	0	0	0.796	0.296	0.742	0.940	0.758	0.913
1	0	0	0	0	0	0	0	0.437	0.156	0.366	0.776	0.694	0.740
0	1	0	0	0	0	0	0	0.403	0.176	0.245	0.612	0.495	0.584
0	0	1	0	0	0	0	0	0.454	0.189	0.344	0.788	0.667	0.766
0	0	0	1	0	0	0	0	0.352	0.179	0.289	0.767	0.648	0.733

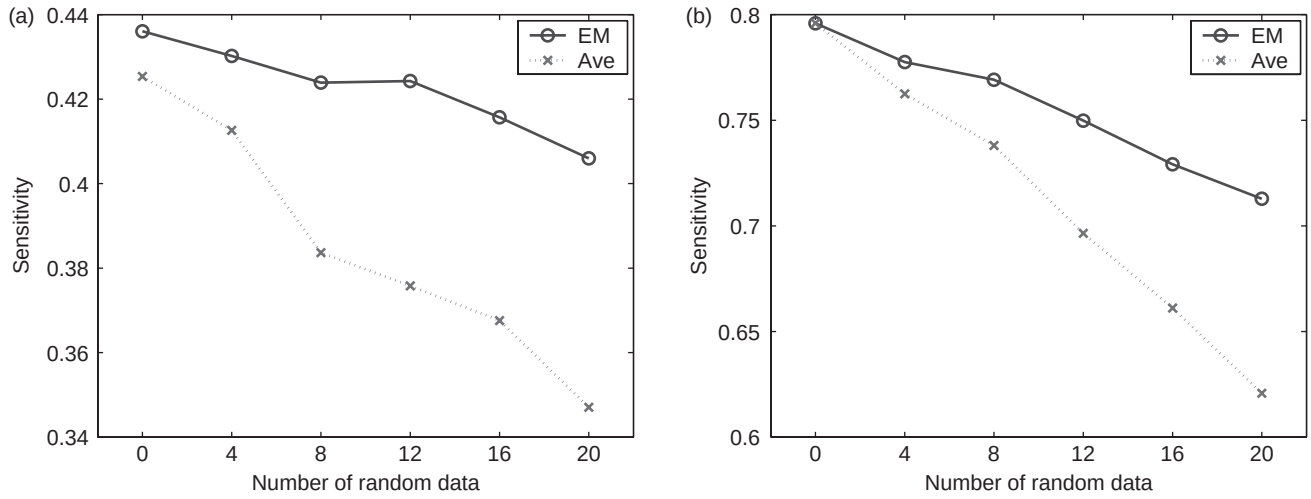


Fig. 5. Accuracy of network prediction for (**a**) metabolic and (**b**) protein interaction networks against the number of decoy kernel matrices. Our algorithm (EM) is superior to the simple kernel average (Ave) because it can automatically exclude decoy kernel matrices.

Table 3. Prediction performance of train–extra edges and extra–extra edges

	Sensitivity			ROC score		
	Q	*P*	KCCA	*Q*	*P*	KCCA
Metabolic network						
All	0.430	0.125	0.370	0.816	0.616	0.782
Train–extra	0.442	0.127	0.382	0.821	0.617	0.789
Extra–extra	0.225	0.102	0.165	0.693	0.599	0.651
Protein interaction network						
All	0.777	0.288	0.711	0.929	0.754	0.899
Train–extra	0.787	0.286	0.719	0.934	0.755	0.903
Extra–extra	0.623	0.314	0.533	0.858	0.749	0.840

'Train–extra' denotes edges between the training network and extra proteins corresponding to Q_{vh}. Extra–extra denotes edges among extra proteins corresponding to Q_{hh}. All shows the average over all edges.

(train–extra edges) and those among extra proteins (extra–extra edges), which corresponds to the submatrices Q_{vh} and Q_{hh}, respectively. We investigated the prediction performance separately for each category (Table 3), and found that the extra–extra edges are much harder to predict.

5.4 Remarks on yeast two-hybrid network

The yeast two-hybrid network is considered as an important information source for predicting the ground truth protein interaction network, because they share a large overlap. Actually, the two-hybrid network is a subgraph of the ground truth network: all the 1112 edges in the two-hybrid network are included in the ground truth network having 2438 edges. However, the experimental result did not agree with this consideration, namely the two-hybrid network performed poorly (ROC score 0.612), and the weight assigned by our algorithm was the smallest among the four data.

This result reflects the difficulty in predicting new edges only from known edges in the subgraph. In the experiment, the diffusion kernel

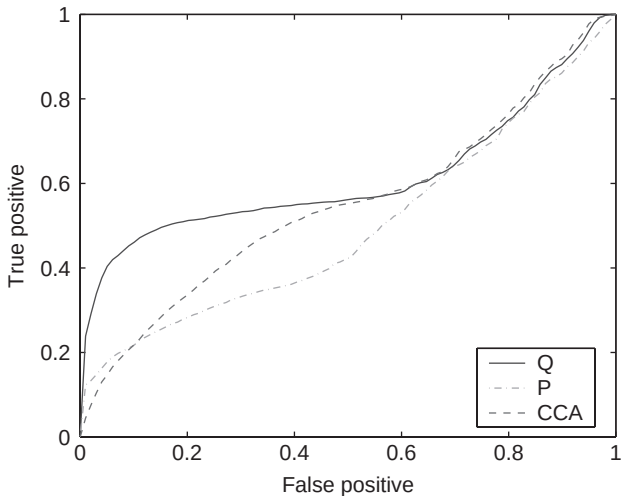


Fig. 6. ROC curves for the protein interaction network using only the y2h network.

made from the two-hybrid network was very informative for predicting its own edges, but almost useless for predicting new edges. Figure 6 shows the complete ROC curve of our algorithm using the two-hybrid network only. The ROC curve rises very sharply when the false positive rate is low. It means that the high confidence predictions, corresponding to known edges, are mostly accurate. However, the curve becomes almost flat around 20% false positive rate due to poor prediction of new edges. As a result, the ROC score (i.e. the area under the curve) turned out to be smaller than those from the other types of data.

6 DISCUSSION AND CONCLUSION

This paper presents a novel method for network inference with automatic data selection. We adopted the kernel matrix representation of networks, which is rather simple compared with a Bayesian network

characterizing conditional independent relationships. The advantage of taking a simpler framework is that the learning problem becomes much simpler. Moreover, it is easier to incorporate additional variables to allow data selection. Such data selection techniques are crucial, as more and more biological observations become available.

High ROC scores in our cross-validation experiments imply that the training network is very informative for predicting edges outside the network. This would not occur if the network were generated randomly. In fact, protein networks have the small world property and contain many small cliques (Goldberg and Roth, 2003). Our method seems to benefit from the small world property, but the underlying mechanism of that benefit remains unclear.

In our algorithm, we assumed that the subnetwork is perfectly known for the first n nodes. However, one can also consider a more general situation where a set of known edges are distributed on the whole network. In this case, Q does not have a block structure like Equation (1), but the elements of Q corresponding to known edges are individually constrained. The minimization of $D[Q, P]$ with respect to Q is still a convex problem but the solution cannot be given in closed form. However, the problem can be solved iteratively, for example, by applying the coordinate descent algorithm to its dual problem (Speed and Kiiveri, 1986).

We have described our method as a network prediction method, but it can also be used in combination with support vector machines, e.g. for protein function prediction (Tsuda and Noble, 2004). Theoretically, our method is somewhat related to Gaussian process classifiers (Williams and Barber, 1998) because they regard the kernel matrix as a covariance matrix of a Gaussian distribution. It would also be interesting to combine our method with the Gaussian process to make a fully integrated Bayesian classifier.

ACKNOWLEDGEMENTS

The authors wish to thank T. Natsume, K. Tomii and T. Kin for the helpful discussions. Thanks are also due to two anonymous refrees for useful suggestions. We appreciate Y. Yamanishi and J.P. Vert for providing their datasets.

REFERENCES

Amari,S.-I. and Nagaoka,H. (2000) *Methods of Information Geometry*. Oxford University Press, Oxford, UK.

Burg,J.P. *et al*. (1982) Estimation of structured covariance matrices. *Proc. IEEE*, **70**, 963–974.

Dempster,A.P. *et al*. (1977) Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc., B*, **39**, 1–38.

Eisen,M.B. *et al*. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl Acad. Sci. USA.* **95**, 14863–14968.

Friedman,N. (2004) Inferring cellular networks using probabilistic graphical models. *Science*, **303**, 799–805.

Goldberg,D.S. and Roth,F.P. (2003) Assessing experimentally derived interaction in a small world. *Proc. Natl Acad. Sci. USA*, **100**, 4372–4376.

Gribskov,M. and Robinson,N.L. (1996) Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Comput. Chem.* **20**, 25–33.

Huh,W.K. *et al*. (2003) Global analysis of protein localization in budding yeast. *Nature*, **425**, 686–691.

Ito,T. *et al*. (2001) A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc. Natl Acad. Sci. USA*, **98**, 4569–4574.

Kanehisa,M. *et al*. (2004) The KEGG resource for deciphering the genome. *Nucleic Acids Res.* **1**, D277–D280.

Kin,T. *et al*. (2004) Protein classification via kernel matrix completion. In Schölkopf,B., Tsuda,K. and Vert,J.P, (eds), *Kernel Methods in Computational Biology*, MIT Press, Cambridge, MA, pp. 261–274.

Kondor,R.I. and Lafferty,J. (2002) Diffusion kernels on graphs and other discrete structures. In Sammut,C. and Hoffmann,A.G. (eds), *Machine Learning, Proceedings of the 19th International Conference (ICML 2002)*, Morgan Kaufmann, San Francisco, pp. 315–322.

Lanckriet,G.R.G. *et al*. (2004) A statistical framework for genomic data fusion. *Bioinformatics*, **20**, 2626–2635.

Malley,J.D. (1994) *Statistical Applications of Jordan Algebras*. Springer Verlag, NY.

Saito,R. *et al*. (2003) Construction of reliable protein–protein interaction networks with a new interaction generality measure. *Bioinformatics*, **19**, 756–763.

Schölkopf,B., Tsuda,K. and Vert,J.P. (eds) (2004) *Kernel Methods in Computational Biology*. MIT Press, Cambridge, MA.

Speed,T. and Kiiveri,H. (1986) Gaussian Markov distributions over finite graphs. *Ann. Statist.*, **14**, 138–150.

Spellman,P.T. *et al*. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, **9**, 3273–3297.

Tsuda,K. and Noble,W.S. (2004) Learning kernels from biological networks by maximizing entropy. *Bioinformatics*, **20** (Suppl. 1), i326–i333.

Tsuda,K. *et al*. (2003) The EM algorithm for kernel matrix completion with auxiliary data. *J. Mach. Learn. Res.* **4**, 67–81.

Uetz,P. *et al*. (2000) A comprehensive analysis of protein–protein interactions in *Saccharomyces cerevisiae*. *Nature*, **403**, 623–627.

von Mering,C. *et al*. (2002) Comparative assessment of large-scale data sets of protein–protein interactions. *Nature*, **417**, 399–403.

Williams,C.K.I. and Barber,D. (1998) Bayesian classification with Gaussian processes. *IEEE Trans. PAMI.* **20**, 1342–1351.

Yamanishi,Y. *et al*. (2004) Protein network inference from multiple genomic data: a supervised approach. *Bioinformatics*, **20**(Suppl. 1), i363–i370.