Finding Regulatory Elements Using Joint Likelihoods for Sequence and Expression Profile Data

Ian Holmes(*) and **William J. Bruno**
ihh@fruitfly.org billb@t10.lanl.gov
Theoretical Biology & Biophysics, T-10, MS K-710, Los Alamos National Laboratory, NM 87545
(*) Present address: Berkeley Drosophila Genome Project, LSA Rm 539
UC Berkeley, CA 94720-3200 Tel (510) 643-9944 Fax (510) 643-9947

Keywords: microarrays, clustering, promoters, Gibbs sampling, Gaussian processes

Abstract

A recent, popular method of finding promoter sequences is to look for conserved motifs upstream of genes clustered on the basis of expression data. This method presupposes that the clustering is correct. Theoretically, one should be better able to find promoter sequences and create more relevant gene clusters by taking a unified approach to these two problems. We present a likelihood function for a "sequence-expression" model giving a joint likelihood for a promoter sequence and its corresponding expression levels. An algorithm to estimate sequence-expression model parameters using Gibbs sampling and Expectation/Maximization is described. A program, called **kimono**, that implements this algorithm has been developed: the source code is freely available on the Internet.

Introduction

A promising use for data from expression profiling experiments is as a signpost for finding transcription factor binding sites. For example, groups of putatively co-regulated transcripts may be identified by clustering the expression profiles generated by a series of microarray experiments (Bucher 1999). The promoter sequences for each transcript in a cluster may then be fed to a motif discovery algorithm. Motifs that are common to a set of apparently co-expressed genes are plausible candidates for binding sites implicated in transcriptional regulation (Wasserman & Fickett 1998).

Such was the approach taken by Church and co-workers (Tavazoie *et al.* 1999), who used the k-means algorithm to cluster the *Saccharomyces* mitotic cell cycle expression data generated by Cho *et al.* (1998) (k-means is an algorithm that clusters points in a multi-dimensional vector space by repeatedly assigning each point to the nearest of k cluster "centroids", calculating newly optimal centroids for each group, and iterating until convergence). The Church group then looked for ungapped motifs in the corresponding genomic sequence using a Gibbs sampler (which samples from the posterior probability distribution over ungapped alignments by repeatedly sampling one row at a time, picking a new indentation by making a random choice weighted by the probability of the resulting aligned motif (Lawrence *et al.* 1993)).

Copyright © 2000, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

It is acknowledged that this endeavor is fraught with potential pitfalls: noise in the microarray datasets, cryptic promoter elements, interactions between binding proteins and alternative modes of transcriptional regulation are among the aspects of the underlying biological process that are poorly understood. However, even quite simple algorithms that ignore such issues appear to generate interesting results (Spellman *et al.* 1998; Eisen *et al.* 1998; Tavazoie *et al.* 1999) and so it seems worth investigating improved methods and formalisms that can be of practical benefit.

It is possible to map the two-stage algorithm described by Church and co-workers (Tavazoie *et al.* 1999) into an integrated likelihood framework. The two stages—k-means clustering of expression profiles followed by Gibbs sampling of sequences—can be viewed as operating on the marginal distributions of a joint probabilistic model for sequence and expression data.

Why is it useful to have an integrated likelihood model? One reason is that the two-stage algorithm and the integrated algorithm are solving subtly different problems. Our integrated algorithm can be summarized as:

> *Find clusters of genes that have:*
> *(a) similar expression patterns* **and**
> *(b) similar promoters.*

On the other hand, the two-stage algorithm can be summarized as:

> *Find genes with similar expression patterns,*
> **then** *see if they have similar promoters.*

In particular, with the two-stage algorithm, the presence or absence of a promoter motif can have no influence on which cluster a gene is assigned to. This

is reminiscent of a situation sometimes encountered in medicine, where several distinct genetic diseases can cause thoroughly similar symptoms and thus be naively diagnosed as the same disease. Furthermore, the two-stage algorithm cannot easily apply any penalty for finding the same motif in two different clusters. In the integrated algorithm, however, motifs can play a key role in determining the overall clustering pattern.

The two-stage algorithm is suited for validating the hypothesis that similar expression profiles can imply transcriptional co-regulation via shared transcription factor binding sites. Once this is taken as a fact, the integrated method may be used as a more effective way to find meaningful clusters and promoters. The hope is that using an integrated approach and a better formulated optimization problem will result in significantly improved discriminative power.

In this abstract we describe how to use just such an integrated model for identifying transcription factor (TF) binding motifs. We start, following Bishop (1995), by noting that the k-means algorithm is itself a special case of a parameter estimation algorithm (namely the Expectation/Maximization or EM algorithm) applied to a mixture density where the mixture components are spherical Gaussian distributions. Generalizing, we allow arbitrary variance (and covariance): this gives us a prior model for expression profiles within the framework of "Gaussian processes". The description of the sequence model closely follows that of Lawrence *et al.* (1993), since their development is already probabilistic. We then show how to combine the two models and suggest an adaptation of the Gibbs sampling and EM algorithms to the resulting joint model.

Finally, we direct the reader to an implementation of this algorithm: the **kimono** program (k-means **i**ntegrated **m**odels for **o**ligo**n**ucle**o**tide arrays) which, despite the acronym, can be used with any source of quantitative expression data—not just microarrays! We illustrate the characteristic behavior of the program with data from simulations. The source code for **kimono** is covered by the GNU Public License and will remain available to all users free of charge. We strongly encourage interested parties to contact us to discuss performance, suggest improvements or just offer praise.

Definitions

We begin by introducing some notation for the experimental data. Let the number of gene transcripts be G. The observed sequence and expression data for the g'th gene are $\bar{S}^{(g)}$ and $\vec{E}^{(g)}$ respectively (we will use the notational convention that x is a scalar, $\bar{x}$ is a sequence, $\vec{x}$ is a vector and $\mathbf{x}$ is a matrix).

We suppose that each upstream sequence is the same length, L. Thus, sequence $\bar{S}^{(g)}$ is defined to be the L bases upstream of the transcription start site for gene g, and the i'th nucleotide of this sequence is denoted by $S_i^{(g)}$.

The expression profile $\vec{E}^{(g)}$ for gene g is a vector of real numbers, typically log intensity-ratios (Spellman *et al.* 1998), presumed to be related to mRNA abundance levels. Thus $E_x^{(g)}$ is the "expression level" measured in experiment x, with $1 \le x \le X$. We suppose that the X experiments form a time series (although this is not a restriction—see below), so that experiment x corresponds to time t_x and $E_x^{(g)} \equiv E^{(g)}(t_x)$. Note that this allows for multiple sampling runs if $t_x = t_y$ for some $x \neq y$.

The notation of the previous paragraph could still be considered valid for experiments that did not correspond to a time series. For example, t_x could be a discrete-valued variable representing the tissue from which the cells in experiment x were taken. The point is that we want to be able to model correlations between different expression experiments; if the experiments are actually independent (or more likely, if it's too hard to figure out which experiments should be correlated with which) the usual practice of assuming the experiments are independent can readily be carried out.

The expression model, $\mathcal{E}$

In the k-means algorithm, each cluster may be considered to have a single vector-valued parameter, the "center" of the cluster. The goodness-of-fit of a cluster is measured by an error function defined as the total squared distance (i.e. the sum-of-squares differences) from the cluster center to each of the member points (Tavazoie *et al.* 1999). The optimal position for the cluster center is clearly the mean (centroid) of all the member points.

We seek a likelihood model $\mathcal{E}$ that is analogous to a cluster. Comparing the sum-of-squares error function to a log-likelihood, the most obvious candidate would be a multivariate Gaussian density, i.e.

$$\Pr\left[\vec{E}^{(g)} \pm \Delta | \mathcal{E}, \vec{\mu}, \sigma\right] \simeq \left(2\pi\sigma^2\right)^{-X/2} \exp\left[-\frac{|\vec{E}^{(g)} - \vec{\mu}|^2}{2\sigma^2}\right] \cdot (2\Delta)^X \quad (1)$$

where $\vec{\mu}$ and σ are the (vector-valued) mean and (scalar-valued) standard deviation of the cluster of expression profiles generated by the model $\mathcal{E}$. Note that $\vec{\mu}$ is an adjustable cluster-specific parameter (equivalent to the cluster centroid vector in the k-means algorithm) whereas σ is treated as a fixed, global parameter.

The Δ's arise because, strictly, we have to integrate over an (arbitrarily small) volume of $\vec{E}^{(g)}$-space to get a finite probability; from now on, we will drop these terms from our notation.

The connection to k-means can be made more explicit. If we posit that each of the observed $\vec{E}^{(g)}$ vectors were generated by one of k alternative models, denoting by $\mathcal{E}^{(m^{(g)})}$ the model which gave rise to the g'th vector, and if we treat the $m^{(g)}$ as missing data and apply the EM algorithm, then the k-means algorithm

emerges naturally in the limit $\sigma \to 0$, since the posterior distribution $\Pr\left[\mathcal{E}^{(m^{(g)})}|\vec{E}^{(g)},...\right]$ tends to one for the cluster nearest to $\vec{E}^{(g)}$ and zero otherwise (Bishop 1995). We will elaborate on this treatment below.

We can generalize equation (1) by replacing the variance σ^2 with a covariance matrix $\mathbf{N}$ (for "noise", since the scale of this matrix should reflect the expected dispersion of the cluster):

$$\Pr\left[\vec{E}^{(g)}|\mathcal{E},\vec{\mu},\mathbf{N}\right] = (2\pi|\mathbf{N}|)^{-X/2}\exp\left[-\frac{1}{2}(\vec{E}^{(g)}-\vec{\mu})^T\mathbf{N}^{-1}(\vec{E}^{(g)}-\vec{\mu})\right] \quad (2)$$

We complete the generalized model by writing down a prior distribution for the adjustable parameter $\vec{\mu}$. We use another multivariate Gaussian, with mean $\vec{\nu}$ and covariance matrix $\mathbf{C}$:

$$\Pr[\vec{\mu}|\vec{\nu},\mathbf{C}] = (2\pi|\mathbf{C}|)^{-X/2}\exp\left[-\frac{1}{2}(\vec{\mu}-\vec{\nu})^T\mathbf{C}^{-1}(\vec{\mu}-\vec{\nu})\right] \quad (3)$$

For later convenience, we also introduce a "multiplicity" $\rho^{(g)}$ for each gene, representing the number of exact duplicates of each profile there are in the dataset (one can also think of this value as a weight for the gene). The total log-probability (including the likelihood and the prior) is then

$$\mathcal{L}\left[\vec{\mathbf{E}},\vec{\mu}\mid\mathcal{E},\Theta,\vec{\rho}\right] = \log\Pr[\vec{\mu}|\vec{\nu},\mathbf{C}] + \sum_{g=1}^{G}\rho^{(g)}\log\Pr\left[\vec{E}^{(g)}|\mathcal{E},\vec{\mu},\mathbf{N}\right] \quad (4)$$

where $\Theta = \{\mathbf{N},\vec{\nu},\mathbf{C}\}$ is the global expression model parameter set, $\vec{\rho} = \{\rho^{(g)}\}$ is the multiplicities vector and we have introduced the shorthand $\vec{\mathbf{E}}$ to represent the entire set of expression vectors $\{\vec{E}^{(g)}\}$. Note that the vector $\vec{\rho}$ is indexed by gene g, as contrasted with vectors like $\vec{\mu}$ or $\vec{\nu}$ which are indexed by experiment x.

The posterior distribution of the signature expression profile for the cluster, $\Pr\left[\vec{\mu}|\vec{\mathbf{E}},...\right]$, is also a multivariate Gaussian. From (4), the optimal value for $\vec{\mu}$ is given by the matrix equation:

$$\vec{\mu}^{(\mathrm{opt})} = \left(\mathbf{C}^{-1}+\sum_{g=1}^{G}\rho^{(g)}\mathbf{N}^{-1}\right)^{-1} \times\left(\mathbf{C}^{-1}\vec{\nu}+\mathbf{N}^{-1}\sum_{g=1}^{G}\rho^{(g)}\vec{E}^{(g)}\right) \quad (5)$$

In the limit of large G (or, strictly speaking, as $\sum\rho^{(g)}$ tends to infinity), this tends towards the $\rho^{(g)}$-weighted centroid of the $\vec{E}^{(g)}$ vectors, as with k-means.

Can we find an interpretation for these Gaussian priors? Recalling that $E_x^{(g)}$ is the expression level at time t_x, equation (3) implies that the probability distribution over all possible cluster expression profiles $\{\mu(t)\}$ is uniquely specified by the covariance between any two points, i.e. the entries of the covariance matrix:

$$C_{xy} = \langle(\mu(t_x)-\nu(t_x))\cdot(\mu(t_y)-\nu(t_y))\rangle_{\{\mu(t)\}}$$

Such probability distributions over function spaces are described by the theory of Gaussian processes (GPs), mathematical models for randomly varying signals (MacKay 1997). A GP has the property that if it's sampled at any number of points, the joint distribution of the sample levels is a multivariate Gaussian. This is because a GP may be viewed as an infinite-dimensional Gaussian distribution (one dimension for every possible value of t) and the marginal distribution of any subspace of a multidimensional Gaussian is itself Gaussian.

An almost-trivial example is white noise: if samples are taken at a set of time points, each sampled value will be independently normally distributed. In this case, the covariance matrix is just the identity matrix. This is therefore the implicit process underlying equation (1) and the k-means algorithm.

The great boon of Gaussian processes is the availability of well-studied covariance functions corresponding to e.g. Brownian motion, short-range fluctuations, quasi-periodic oscillations or indeed any given power spectrum (Abrahamsen 1997). Covariance functions appropriate to a system of interest can be chosen: for example, when clustering cell-cycle transcript levels, one might choose a $\mathbf{C}$ matrix representing a weakly periodic prior for the mean cluster signal and an $\mathbf{N}$ matrix representing quickly-damped fluctuations from this signal, plus uncorrelated experimental errors, for the individual gene profiles. In the event that the experimental protocol induces correlated errors, these can also be modeled in this framework. Suitable covariance functions for this example are described by MacKay (1997).

The sequence model, $\mathcal{S}$

We use a fixed-length, ungapped, nonrepeating model for TF binding motifs, following Lawrence *et al.* (1993). This model is simplistic and has been improved upon (Neuwald, Liu, & Lawrence 1995; Zhu, Liu, & Lawrence 1998; Roth *et al.* 1998), but it is a plausible first approximation. Since it has been described extensively elsewhere, we will keep our (slightly modified) definition terse.

Denote the motif model by $\mathcal{S}$ and the "null" model by $\mathcal{S}_\emptyset$. The probability of nucleotide n under the null model is p_n (i.e. $\vec{p}$ is a vector indexed by type of nucleotide), so that the null model likelihood is:

$$\Pr\left[\bar{S}^{(g)}|\mathcal{S}_\emptyset,\vec{p}\right] = \prod_{i=1}^{L}p_{S_i^{(g)}} \quad (6)$$

Suppose that the weight matrix for the motif model is $\mathbf{q}$, so that the probability of seeing nucleotide n at position j in the motif is q_{jn}. Fix the motif length at η. The prior probability of the motif is given by a product of independent identical Dirichlet distributions for each position j, with pseudocounts $\vec{D} = \{D_n\}$ (these counts will typically be proportional to the null model probabilities p_n):

$$\Pr\left[\mathbf{q}|\vec{D}\right] = \prod_{j=1}^{\eta} \mathcal{D}(\vec{q}_{[j]}|\vec{D}) \tag{7}$$

Here $\vec{q}_{[j]}$ is the vector of nucleotide probabilities for position j and $\mathcal{D}(\vec{\theta}|\vec{\alpha})$ is the Dirichlet distribution with pseudocounts $\vec{\alpha}$ (Durbin *et al.* 1998).

Consider the first base in sequence g that is aligned to the motif. Let the index of this position be $a^{(g)}$ (so that the complete alignment is specified by the gene-indexed vector $\vec{a} = \{a^{(g)}\}$). Then the odds-ratio of the aligned motif to the null model is:

$$\frac{\Pr\left[\bar{S}^{(g)}, a^{(g)}|\mathcal{S}, \vec{p}, \mathbf{q}\right]}{\Pr\left[\bar{S}^{(g)}|\mathcal{S}_\emptyset, \vec{p}\right]} = \prod_{j=1}^{\eta} \frac{q_{j,S^{(g)}_{j+a^{(g)}-1}}}{p_{S^{(g)}_{j+a^{(g)}-1}}} \tag{8}$$

We can optimize the total log-likelihood, including the Dirichlet prior, by setting the weight matrix entries proportional to the column nucleotide frequencies plus the pseudocounts. For convenience we first recall the multiplicities $\rho^{(g)}$ introduced for expression data in equation (4) to write down an analogous total log-probability for the sequence data:

$$\mathcal{L}\left[\bar{\mathbf{S}}, \vec{a}, \mathbf{q} \mid \mathcal{S}, \Theta', \vec{\rho}\right] = \log \Pr\left[\mathbf{q}|\vec{D}\right] + \sum_{g=1}^{G} \rho^{(g)} \log \Pr\left[\bar{S}^{(g)}, a^{(g)}|\mathcal{S}, \vec{p}, \mathbf{q}\right] \tag{9}$$

where $\Theta' = \{\vec{p}, \vec{D}\}$ is the global sequence model parameter set and $\bar{\mathbf{S}}$ is a shorthand for the entire sequence dataset $\{\bar{S}^{(g)}\}$.

The optimal weight matrix entries are then given by the (multiplicity-weighted) nucleotide frequencies, plus pseudocounts:

$$q_{jn}^{(\mathrm{opt})} = \frac{D_n + \sum_{\{g: S^{(g)}_{j+a^{(g)}-1} = n\}} \rho^{(g)}}{\sum_{n'} D_{n'} + \sum_{g=1}^{G} \rho^{(g)}} \tag{10}$$

The update procedure for Gibbs sampling is as follows:

- For each gene g (picked in a random order)
  - For each possible alignment $a = a^{(g)}$, calculate:
    - the optimal weight matrix $\mathbf{q}_a^{(\mathrm{opt})}$
    - the corresponding alignment probability $\Pr\left[\bar{S}^{(g)}, a, \mathbf{q}_a^{(\mathrm{opt})}|\mathcal{S}, \vec{p}\right]$
  - Sample an alignment from the posterior distribution $\Pr\left[a, \mathbf{q}_a^{(\mathrm{opt})}|\bar{S}^{(g)}, \mathcal{S}, \vec{p}\right]$

Numerous generalizations of this algorithm—for example, to look for multiple motifs (Neuwald, Liu, & Lawrence 1995), or to search the complementary strand of DNA (Fickett 1996)—have been published. Systematic studies of the convergence properties of the algorithm, whether empirical or theoretical, are less numerous. A heuristic rule is to use the expected information content ratio of sequence data to motif as a guide to the number of iterations to run. Another useful rule of thumb is frequently to start sampling from scratch with a random seed alignment (but how frequently is "frequently"?). Arrival at the same solution from independent random seeds seems to be a good indicator of having found the global optimum (Lawrence 1996); we call this the *déja vu* heuristic.

The sequence-expression model, $\mathcal{SE}$

Combining equations (2) and (8), we can now write down a joint likelihood for a transcript's promoter sequence alignment and expression profile:

$$\Pr\left[\bar{S}\vec{E}^{(g)}, a^{(g)}|\mathcal{SE}, \Lambda, \Theta''\right] = \Pr\left[\bar{S}^{(g)}, a^{(g)}|\mathcal{S}, \vec{p}, \mathbf{q}\right] \Pr\left[\vec{E}^{(g)}|\mathcal{E}, \vec{\mu}, \mathbf{N}\right] \tag{11}$$

where $\bar{S}\vec{E}^{(g)}$ is the combined sequence-expression information for gene g, $\Lambda = \{\mathbf{q}, \vec{\mu}\}$ are the variable parameters, $\Theta'' = \{\Theta, \Theta'\} = \{\mathbf{N}, \vec{\nu}, \mathbf{C}, \vec{p}, \vec{D}\}$ are the fixed global parameters and $\mathcal{SE}$ is our notation for the sequence-expression model.

We can also write down the total joint log-probability for the entire set of transcripts by combining equations (4) and (9):

$$\mathcal{L}\left[\bar{\mathbf{S}}\vec{\mathbf{E}}, \vec{a}, \Lambda \mid \mathcal{SE}, \Theta'', \vec{\rho}\right] = \mathcal{L}\left[\bar{\mathbf{S}}, \vec{a}, \mathbf{q} \mid \mathcal{S}, \Theta', \vec{\rho}\right] + \mathcal{L}\left[\vec{\mathbf{E}}, \vec{\mu} \mid \mathcal{E}, \Theta, \vec{\rho}\right] \tag{12}$$

Clustering using multiple $\mathcal{SE}$ models

We now move to the question of how to cluster sequence and expression data simultaneously using multiple competing $\mathcal{SE}$ models. The prototype optimization algorithms (Expectation/Maximization for clustering, Gibbs sampling for alignment) have already been introduced, so it only remains to combine them. We will describe just one of several ways that this can be done.

We must introduce some more notation (the last!). Suppose that we have k sequence-expression models; the m'th model is $\mathcal{SE}_{[m]}$ and its parameters are $\Lambda_{[m]} = \{\vec{a}_{[m]}, \mathbf{q}_{[m]}, \vec{\mu}_{[m]}\}$.

We also allow a null sequence-expression model, $\mathcal{SE}_\emptyset$. The null sequence likelihood is given by (6) and the null expression likelihood by setting $\vec{\mu}$ equal to $\vec{E}^{(g)}$ in (3).

The joint null likelihood is the product of these two likelihoods

$$\Pr\left[\bar{S}\vec{E}^{(g)}|\mathcal{SE}_{\emptyset},\Theta''\right] = \Pr\left[\bar{S}^{(g)}|\mathcal{S}_{\emptyset},\vec{p}\right]\Pr\left[\vec{E}^{(g)}|\mathcal{E}_{\emptyset},\vec{\mu},\mathbf{N}\right] \quad (13)$$

by analogy with equation (11).

The null sequence model generates independent unrelated sequences and the null expression model generates expression profiles each of which is effectively an independent one-member cluster.

The basic job of our clustering algorithm is to assign each gene to a cluster. We handle this in our formalism by having a different multiplicities vector for each model, denoting the vector for the m'th model by $\vec{\rho}_{[m]}$ (and for the null model, $\vec{\rho}_{\emptyset}$). Then the act of assigning gene g to model $\mathcal{SE}_m$ is equivalent to setting

$$\rho^{(g)}_{[m']} = \begin{cases} 1 & \text{if } m' = m \\ 0 & \text{if } m' \neq m \end{cases} \quad (14)$$

In our Gibbs/EM algorithm, the $\rho^{(g)}_{[m]}$ are the missing data. The E-step of the EM algorithm involves setting these missing data to their expectation values according to the posterior probability distribution $\Pr\left[\mathcal{SE}_{[m]}|\bar{S}\vec{E}^{(g)},...\right]$ (i.e. multiplying equation (14) by this probability and summing over m).

In other words, as with k-means, we effectively force the models to compete to explain the genes. At the same time, by allowing the $\rho^{(g)}_{[m]}$ to be continuously valued expectations rather than zero or one, we allow uncertainty where k-means does not. See e.g. Bishop (1995) for an alternative explanation of this approach.

We are now ready to describe our Gibbs/EM update procedure, which is:

- For each model m
  - For each gene g (picked in a random order)
    - Sample an alignment $a = a^{(g)}_{[m]}$ from the posterior distribution $\Pr\left[a, \mathbf{q}^{(\mathrm{opt})}_{[m],a}|\bar{S}^{(g)},\mathcal{S}_{[m]},\vec{p}_{[m]}\right]$ *(the Gibbs sampling step)*
- Set the multiplicities $\rho^{(g)}_{[m]}$ equal to the posteriors $\Pr\left[\mathcal{SE}_{[m]},a^{(g)}_{[m]},\Lambda_{[m]}|\bar{S}\vec{E}^{(g)},\Theta''\right]$ *(the Expectation step)*
- Set the cluster centers $\vec{\mu}_{[m]}$ equal to the optimal values $\vec{\mu}^{(\mathrm{opt})}_{[m]}$ *(the Maximization step)*

Relevant equations (apart from Bayes' rule) are (7), (8) and (10) for the Gibbs sampling step, (12) and (13) for the Expectation step (we can also introduce prior model probabilities $\Pr\left[\mathcal{SE}_{[m]}\right]$ here) and (5) for the Maximization step.

Another way of viewing this algorithm is as a mixture density estimation problem, where a k-component mixture of $\mathcal{SE}$ models must be fit to an observed scattering of sequence-expression data. Viewed this way, the oft-quoted limitation of k-means (that the parameter k must be known in advance) seems not insurmountable, as mixture density estimation when the number of components is unknown is a familiar problem in Bayesian statistics.

A note on scaling of expression data

Consider what happens if we take a couple of real numbers, $\gamma^{(g)}$ and $\beta^{(g)}$, and use them to scale/offset the expression data for gene g by applying the mapping

$$E^{(g)}_x \rightarrow \exp\left[-\gamma^{(g)}\right](E^{(g)}_x - \beta^{(g)})$$

to all likelihood formulae involving expression profiles.

If we choose

$$\beta^{(g)} = \frac{1}{X}\sum_{x=1}^{X} E^{(g)}_x$$

$$\gamma^{(g)} = -\frac{1}{2}\log\left[\frac{1}{X}\sum_{x=1}^{X}(E^{(g)}_x - \tau^{(g)})^2\right]$$

then this is equivalent to the "normalization" of profile vectors that is commonly applied to expression data preceding analysis. While this normalization is effective as a correction for biases in the experimental protocol, it is also crude: it does not distinguish between systematic experimental errors and genuine differences in transcription levels.

Our likelihood framework permits a more principled approach. We can incorporate prior probabilities for the model-independent scaling parameters $\beta^{(g)}$ and $\gamma^{(g)}$. Suitable priors might be the Gaussian distributions $\beta^{(g)} \sim \mathcal{N}(0,\tau^2)$ and $\gamma^{(g)} \sim \mathcal{N}(0,\upsilon^2)$. The values of $\beta^{(g)}$ and $\gamma^{(g)}$ can then be sampled at some point during the Gibbs/EM update procedure, e.g. between the Gibbs and Expectation steps. One way to do the sampling is to generate values from the prior distribution, then choose one of these values randomly according to the likelihoods given by equation (12).

It is equally straightforward to incorporate experiment-dependent parameters into the scaling, e.g.

$$E^{(g)}_x \rightarrow \exp\left[-\gamma^{(g)} - \delta^{(x)}\right](E^{(g)}_x - \beta^{(g)} - \epsilon^{(x)})$$

with $\delta^{(x)} \sim \mathcal{N}(0,\phi^2)$ and $\epsilon^{(x)} \sim \mathcal{N}(0,\varphi^2)$.

When such additional parameters are allowed, we may constrain them (by keeping τ, υ, ϕ and φ small) to avoid overfitting when the dataset is sparse.

Implementation

We have developed a program, called **kimono**, to implement the Gibbs/EM algorithm for sequence-expression clustering with competing $\mathcal{SE}$ models. The

source code for **kimono** is freely available, under the terms of the GNU Public License (GPL 2000), from the following URL:

`http://www.okchicken.com/~yam/kimono/`

kimono implements all the features described in this abstract, including arbitrary covariance priors, null sequence and expression models, prior probabilities for models, reverse strand sampling, multiple re-seeding and the *déja vu* heuristic. It also offers some useful features not covered here, including automatic null model optimization and an experimental implementation of a simulated annealing version of Gibbs/EM.

The program is easy to use: it understands standard file formats (FASTA format for sequence data (Pearson & Lipman 1988), Stanford format tab-delimited tables for expression data (Spellman *et al.* 1998)). In addition, it uses a customizable multi-priority logging system and displays friendly help messages and errors.

The acronym **kimono** stands for k-means **i**ntegrated **m**odels for **o**ligo**n**ucle**o**tide arrays. We would however like to point out that nothing in the present formalism prohibits the use of quantitative expression data from large-scale functional genomics experiments using technologies other than microarrays.

kimono is implemented in C++ using the Standard Template Library and compiles cleanly on a RedHat 6.0 GNU/Linux system using the `gcc` compiler, version egcs-2.91.66.

Figure 1 shows sample paths of log-likelihood over time illustrating **kimono**'s typical behavior on simulated data. The paths shown were selected for illustrative purposes from ten randomly seeded runs on each of four simulated datasets. The variables distinguishing the four plots are the expression signal-to-noise ratio ε defined by

$$\varepsilon = \mathrm{tr}(\mathbf{C})/\mathrm{tr}(\mathbf{N})$$

and the sequence signal-to-noise ratio ψ defined by

$$\log L\psi = \sum_j \sum_n q_{jn} \log \frac{q_{jn}}{p_n}$$

These signal-to-noise ratios refer to the actual parameters used in generating the data. The **kimono** program was run using its default settings, except for k which was set to 3 to match the simulated data. The default **kimono** parameters are the same as the simulation parameters listed in the caption to Figure 1, with the exception of the **C** and **N** matrices (which are equal to the identity matrix by default); also, the pseudocounts D_n are set to $0.1p_n$, where the null nucleotide probabilities p_n are automatically estimated from composition of the sequence database.

The graphs in Figure 1 exhibit several features characteristic of this algorithm. Whereas the Gibbs sampler for a single alignment initially spends most of its time exploring a more-or-less flat log-likelihood landscape, or "plateau", before finding a seed for the true signal and then rapidly converging to the global optimum (Lawrence *et al.* 1993), the multiple-model sampler steps through several plateaux at different levels as the distinct clusters converge semi-independently on their respective solutions. The convergence of different clusters is not expected to be completely independent, as the arrival of each cluster at its optimal alignment helps to fix some of the weights $\vec{\rho}$ correctly, and thus limits the introduction of noise into the other alignments.

Sequence length	Rounds to convergence	Time (μs) per residue cluster
20	12 ± 6	29 ± 4
30	36 ± 19	26 ± 2
50	49 ± 23	28 ± 0.8
70	61 ± 25	30 ± 0.6
110	66 ± 18	29 ± 0.2
150	155 ± 36	28 ± 0.07
230	412 ± 55	28 ± 0.06
310	674 ± 47	28 ± 0.05
470	984 ± 13	27 ± 0.05

Table 1: **kimono** performance on a 450MHz Red Hat Linux Pentium system. In each round, the alignment of every sequence to every model was sampled once. The simulation parameters were: number of genes $G = 24$, number of models $k = 3$, nucleotide probability $p_n = \frac{1}{4}$, motif length $\eta = 10$, expected expression profile $\vec{\nu} = \vec{0}$ and covariance matrices **N** and **C** equal to multiples of the identity matrix, with expression signal-to-noise $\varepsilon = 50$. Sampling was terminated after 1000 rounds regardless of convergence.

For the data shown, the simulated expression signal-to-noise ratio ε is quite large and so the assignments of genes to their correct clusters is reasonably well constrained. However, as ε or ψ is reduced, the clustering becomes more uncertain, with the observable effect that the plateaux become bumpier and start to blur into one another. One corollary of this is that one starts to see the algorithm step downwards to less favorable plateaux. Of course, this will happen (albeit less frequently) even in the simple case, when there's only one plateau separated from the optimum by a large jump in the log-likelihood. This behavior can be seen in plot c in Figure 1.

A characteristic feature of Gibbs sampling in general is that the distribution of convergence times has a long tail. This is apparently due to the algorithm occasionally getting stuck in an appealing local optimum. Convergence is generally slower when ε and ψ are small.

An idea of the running speed of **kimono** can be gained by studying Table 1. Extrapolating to 30 clusters—the number used by Tavazoie *et al.*—and the full complement of roughly 6,000 yeast ORFs, convergence is expected in approximately one month on a 450MHz Pentium system. Our procedure is inherently

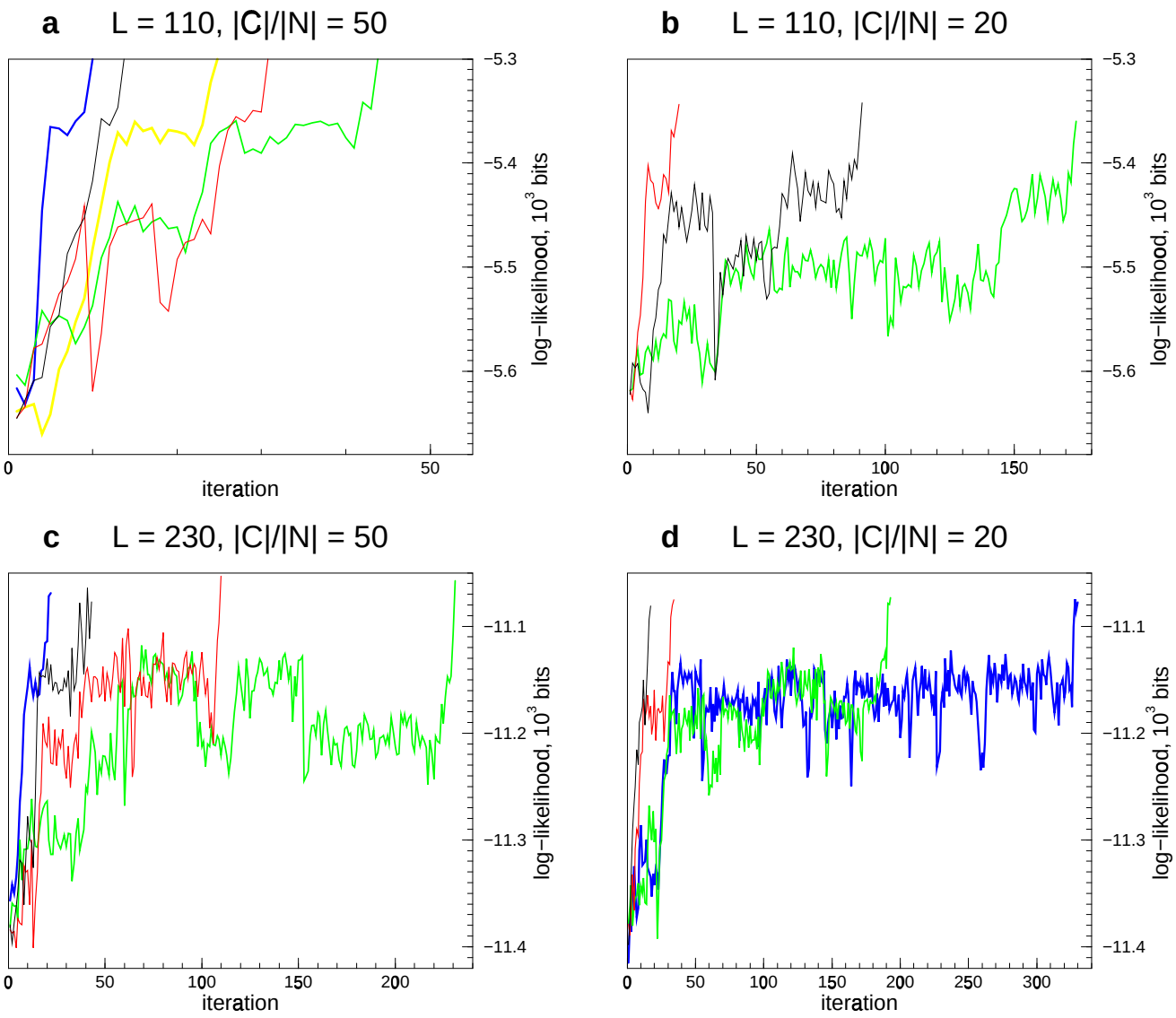

Figure 1: Sample paths of total log-likelihood versus number of iterations, illustrating **kimono**'s characteristic behavior on simulated data generated with number of experiments $X = 10$, number of genes $G = 24$, number of models $k = 3$, nucleotide probability $p_n = \frac{1}{4}$, motif length $\eta = 10$, expected expression profile $\vec{\nu} = \vec{0}$ and covariance matrices $\mathbf{N}$ and $\mathbf{C}$ equal to multiples of the identity matrix. The **kimono** parameters were left at their default settings, except for k which was set to 3. The Gibbs/EM algorithm was halted when the correct solution was approached to within a mildly error-tolerant margin. Different paths in each graph correspond to different random seeds. **a**: sequence length $L = 110$, expression signal-to-noise $\varepsilon = 50$, and sequence signal-to-noise $\psi \simeq 350$. **b**: $L = 110$, $\varepsilon = 20$, $\psi \simeq 350$. **c**: $L = 230$, $\varepsilon = 50$, $\psi \simeq 170$. **d**: $L = 230$, $\varepsilon = 20$, $\psi \simeq 170$.

slower than the two-stage algorithm of Tavazoie *et al.*, since their algorithm samples one alignment per gene where our algorithm has to sample k such alignments. It would be straightforward to parallelise this extra sampling duty on a multiprocessor system. At the time of writing, there is also plenty of room for optimizing the C++ code.

The two-stage algorithm is a special case of our algorithm that can be obtained formally by scaling the matrices $\mathbf{C}$ and $\mathbf{N}$ by some parameter λ and sampling the alignment for each gene to only the most plausible model with some probability p. As $\lambda \to 0$ and $p \to 1$, then the two-stage algorithm is approached.

Discussion

We have described a probabilistic model—the $\mathcal{SE}$ model—that relates a promoter sequence to a quantitative expression profile of the corresponding downstream message. We have shown how to use multiple $\mathcal{SE}$ models for identification of putative transcription factor binding sites, giving a Gibbs/EM algorithm for model training. The program **kimono** implements this algorithm; its source code is freely available from the Internet.

Clearly, probabilistic modeling is not limited to ungapped motif models and simple Gaussian process clusters. It is possible to adapt the approach that we have described in many ways. One could imagine extending the $\mathcal{S}$ and $\mathcal{E}$ models described here, replacing them with other models, adding new features or modeling different kinds of functional genomics data. Although we do not intend to list every possible modification, there are some that are worth mentioning.

One obvious extension to the expression half of the $\mathcal{SE}$ model is to include multiple classes of model. For example, when clustering a cell-cycle dataset one might allow for both periodic and non-periodic clusters. Related ideas include allowing the covariance matrix to be a model-specific variable and improving the scaling model for experimental and gene-specific bias in the light of more experimental data.

Promoter sequence models are also worth developing, for a broad range of applications. The most obvious probabilistic framework in which to do this seems to be the Hidden Markov model (HMM). In fact, the ungapped motif models that we use are special cases of profile HMMs with no insert states, delete states or internal cycles. One basic improvement, then, would be to allow loop transitions within the HMM (and thus multiple instances of the motif). Another would be to place a prior distribution over the motif length, allowing it to vary (note however that a geometric prior is far too broad for this job, leading mostly to long, rambling, weak pseudo-motifs; a narrower distribution, like a Gaussian, might work).

A familiar example—the "TATA box"—motivates development of better promoter models for use with our method. Since the TATA box motif is found in the majority of eukaryotic promoter regions regardless of the transcriptional behavior of the downstream gene, it will have some overall aggregating effect on the clustering. This example shows that, counterintuitively, the integration of more (sequence) data can potentially lead to coarser clusters if an incorrect model is used (although we expect this particular example to be handled somewhat better by our Gibbs/EM approach, which allows for uncertainty in the cluster assignments, than by other clustering algorithms, such as k-means, which make "hard" decisions). It is easy to become pessimistic about the chances of ever modeling promoters well, but it is good to remember that, pragmatically speaking, one doesn't need to have a perfect model in order to generate leads that experimentalists can pursue.

Another route to new models is to attempt to place procedures that are not explicitly probabilistic within a Bayesian framework, as has been done previously with the k-means algorithm and mixture density estimation using Gaussian basis functions. With many procedures, the Bayesian interpretation (or the nearest approximation) may be less obvious than the relatively simple mapping between k-means and Gaussian processes. An interesting candidate for this approach might be the logistic regression model for aggregating promoter sequence motifs that was proposed by Wasserman & Fickett (1998). With respect to the analysis of expression data, the support vector machine (SVM) approaches used by the Haussler group are promising (Brown *et al.* 2000); their work also differs from ours in taking a supervised learning approach, where at least some of the cluster membership assignments must be pre-specified by the user.

In addition to these improvements to the probabilistic model, there may be many potential improvements to the Gibbs/EM algorithm that we use to train the model (i.e. to identify and characterize clusters). Examples of such improvements could include simulated annealing approaches (Gilks, Richardson, & Spiegelhalter 1996) or incremental Expectation Maximization (Neal & Hinton 1993). It would also be useful to be able to sample over unknown parameters, such as the number of clusters k, using Markov Chain Monte Carlo methods.

An exciting prospect for the future is the automatic identification of regulatory networks on the basis of functional genomics data (D'haeseleer *et al.* 1999). In a probabilistic framework, this is equivalent to relaxing the independence assumption for transcripts. This presents many challenges to the statistical analyst, including issues of how much data is needed to identify interactions, a problem common in MCMC analysis.

Acknowledgements

WJB and IHH were supported by DOE grant/contract W-7405-ENG-36. IHH was supported by the Fulbright Zeneca research fellowship.

We gratefully acknowledge making use of source code that is freely available on the Internet, including the linear algebra package `Newmat09` (Davies 1999), the random number generation library `randlib` (Brown *et al.*

1997), a set of C++ regular expression classes (Gascoigne *et al.* 1998) and routines from the `HMMER` package (Eddy 1996) due to Sean Eddy and Graeme Mitchison.

This work benefited from discussions with Bill Hlavacek (who originally brought to our attention the work of Tavazoie *et al.*) and Roger Sayle. We would like to thank Oli Bye for providing web hosting services for downloading of the **kimono** program and the Santa Fe Institute for providing additional computing resources. IHH would also like to thank Kristina Ecker.

References

Abrahamsen, P. 1997. A review of Gaussian random fields and correlation functions. Technical Report 917, Norwegian Computing Center, Box 114, Blindern, N-0314 Oslo, Norway.

Bishop, C. M. 1995. *Neural Networks for Pattern Recognition*. Oxford, UK: Oxford University Press.

Brown, B.; Lovato, J.; Russell, K.; and Venier, J. 1997. randlib: Library of routines for random number generation. http://odin.mdacc.tmc.edu/. Department of Biomathematics, Box 237, The University of Texas, M.D. Anderson Cancer Center, 1515 Holcombe Boulevard, Houston, TX 77030.

Brown, M.; Grundy, W.; Lin, D.; Cristianini, N.; Sugnet, C.; Furey, T.; Ares, M.; and Haussler, D. 2000. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proceedings of the National Academy of Sciences of the USA* 97(1):262–267.

Bucher, P. 1999. Regulatory elements and expression profiles. *Current Opinion in Structural Biology* 9(3):400–407.

Cho, R.; Campbell, M.; Winzeler, E.; Steinmetz, L.; Conway, A.; Wodicka, L.; Wolfsberg, T.; Gabrielian, A.; Landsman, D.; Lockhart, D.; and Davis, R. 1998. A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell* 2(1):65–73.

Davies, R. 1999. Newmat09: C++ matrix library. http://webnz.com/robert/nzc_nm09.html.

D'haeseleer, P.; Wen, X.; Fuhrman, S.; and Somogyi, R. 1999. Linear modeling of mRNA expression levels during CNS development and injury. In *Pacific Symposium on Biocomputing*, 41–52. World Scientific Publishing Co.

Durbin, R.; Eddy, S.; Krogh, A.; and Mitchison, G. 1998. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press.

Eddy, S. R. 1996. Hidden Markov models. *Current Opinion in Structural Biology* 6:361–365.

Eisen, M.; Spellman, P.; Brown, P.; and Botstein, D. 1998. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences of the USA* 95:14863–14868.

Fickett, J. 1996. Quantitative discrimination of MEF-2 sites. *Molecular Cell Biology* 16:437–441.

Gascoigne, G.; Noer, G.; Woods, J.; Gilmore, J.; and Spencer, H. 1998. Regexp: C++ regular expression classes.

Gilks, W.; Richardson, S.; and Spiegelhalter, D. 1996. *Markov Chain Monte Carlo in Practice*. Chapman & Hall.

2000. The GNU Public License. Available in full from http://www.fsf.org/copyleft/gpl.html.

Lawrence, C. E.; Altschul, S. F.; Boguski, M. S.; Liu, J. S.; Neuwald, A. F.; and Wootton, J. C. 1993. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science* 262(5131):208–214.

Lawrence, C. 1996. Personal communication.

MacKay, D. J. C. 1997. Introduction to Gaussian processes. Available from http://wol.ra.phy.cam.ac.uk/mackay/.

Neal, R. M., and Hinton, G. E. 1993. A new view of the EM algorithm that justifies incremental and other variants. Preprint, Dept. of Computer Science, Univ. of Toronto, available from ftp://archive.cis.ohio-state.edu/pub/neuroprose/neal.em.ps.Z.

Neuwald, A.; Liu, J.; and Lawrence, C. 1995. Gibbs motif sampling: detection of bacterial outer membrane protein repeats. *PROTSCI* 4(8):1618–1632.

Pearson, W. R., and Lipman, D. J. 1988. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the USA* 4:2444–2448.

Roth, F.; Hughes, J.; Estep, P.; and Church, G. 1998. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nature Biotechnology* 16(10):939–945.

Spellman, P. T.; Sherlock, G.; Zhang, M. Q.; Iyer, V. R.; 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 cerevisiae by microarray hybridization. *Molecular Biology of the Cell* 9:3273–3297.

Tavazoie, S.; Hughes, J.; Campbell, M.; Cho, R.; and Church, G. 1999. Systematic determination of genetic network architecture. *Nature Genetics* 22:281–285.

Wasserman, W., and Fickett, J. 1998. Identification of regulatory regions which confer muscle-specific gene expression. *Journal of Molecular Biology* 278(1):167–181.

Zhu, J.; Liu, J. S.; and Lawrence, C. E. 1998. Bayesian adaptive sequence alignment algorithms. *Bioinformatics* 14:25–39.