BIOINFORMATICS

Vol. 19 Suppl. 1 2003, pages i34–i43
DOI: 10.1093/bioinformatics/btg1003

Predicting bacterial transcription units using sequence and expression data

Joseph Bockhorst[1, 2,*], Yu Qiu[3], Jeremy Glasner[3], Mingzhu Liu[3], Frederick Blattner[3] and Mark Craven[2, 1]

[1]Department of Computer Sciences, [2]Department of Biostatistics & Medical Informatics and [3]Laboratory of Genetics, University of Wisconsin, Madison, Wisconsin 53706, USA

Received on January 6, 2003; accepted on February 20, 2003

ABSTRACT

Motivation: A key aspect of elucidating gene regulation in bacterial genomes is identifying the basic units of transcription. We present a method, based on probabilistic language models, that we apply to predict operons, promoters and terminators in the genome of *Escherichia coli* K-12. Our approach has two key properties: (i) it provides a coherent set of predictions for related regulatory elements of various types and (ii) it takes advantage of both DNA sequence and gene expression data, including expression measurements from inter-genic probes.

Results: Our experimental results show that we are able to predict operons and localize promoters and terminators with high accuracy. Moreover, our models that use both sequence and expression data are more accurate than those that use only one of these two data sources.

Availability: Our *E.coli* transcription-unit predictions are available from http://www.biostat.wisc.edu/gene-regulation/.

Contact: joebock@biostat.wisc.edu; craven@biostat.wisc.edu

INTRODUCTION

A central challenge in computational biology is to uncover the complete gene regulation network of an organism. This challenge can now be profitably attacked given the availability of complete genomes and high-throughput technologies for interrogating the states of cells. A key step in addressing the challenge is to assemble a 'parts list' of the regulatory elements for a given genome. We present an approach, based on probabilistic language models, that uses sequence and expression data to predict a variety of regulatory elements in prokaryotic genomes. We apply this approach to the task of predicting transcription units in the genome of *Escherichia coli* K-12. Our approach has two key properties. First, the approach can provide a coherent set of predictions for related regulatory elements of various types. Second, the approach can take advantage of both DNA sequence data and gene expression data.

The prediction task that we consider here is the following. Given the genome sequence and gene expression data for a prokaryotic organism, predict all of the *transcription units* (TUs) in the genome. The definition of a transcription unit that we assume is: the complete extent of a sequence of DNA that is transcribed to produce a single mRNA transcript. That is, each of our predicted transcription units consists of a sequence of co-transcribed genes along with the associated polymerase binding site and transcription termination signal. Our approach is to use a probabilistic grammar for DNA sequence and expression data. We train this model from a set of known operons, promoters and terminators. The definition of *operon* we use is a set of genes that are transcribed as a unit under some condition. Note that, in contrast to our definition of a transcription unit, our operon definition does not include the exact start and end of transcription points.

The gene expression data we use comes from Affymetrix oligonucleotide arrays for *E.coli* that include probes for both coding and inter-genic regions. Because they contain probes corresponding to inter-genic regions, these arrays are quite advantageous for identifying the 5′ and 3′ ends of transcription units. However, our approach is general enough that it does not require expression data from this type of array. Our approach can use expression data from any type of microarray, and is still applicable even when expression data is not available.

There is a large body of research on using computational methods to recognize regulatory elements in prokaryotic DNA. This work includes methods that learn models to recognize promoters (Pedersen *et al.*, 1996; Thieffry *et al.*, 1998), terminators (Brendel *et al.*, 1986; Carafa *et al.*, 1990; Ermolaeva *et al.*, 2000), and operons (Salgado *et al.*, 2000; Moreno-Hagelsieb and Collado-Vides, 2002). Our

*To whom correspondence should be addressed.

work differs from these efforts in several respects. First, in making our predictions we use not only sequence data, but also expression data. Second, our model simultaneously predicts promoters, terminators and operons instead of treating these as separate prediction tasks. Third, we predict the entire extent of transcription units.

In earlier research (Craven *et al.*, 2000; Bockhorst *et al.*, 2003), we developed probabilistic methods for predicting operons using sequence and expression data. The work presented herein extends our earlier work by predicting the complete extent of transcription units, as opposed to predicting just which sequences of genes are in the same operons. Additionally, our new approach extends our earlier work by simultaneously predicting a coherent set of promoter, terminator and operon predictions for a complete genome. The notion of coherency here is that a set of predictions satisfies the relationships that necessarily hold among the regulatory elements being predicted. For example, each predicted TU must be bracketed by a promoter and a terminator. Also, the expression data used previously contained expression intensities only for genes while here we also utilize intergenic probe measurements.

Yada *et al.* (1999) developed an approach to predict transcription units from sequence data, and Tjaden *et al.* (2002) developed an approach to predict transcription units using expression data from Affymetrix *E.coli* arrays. However, in contrast to our approach, these methods use only a single source of data. As our experiments show, the combination of DNA sequence data and expression data results in more accurate predictions than either alone. Additionally, the use of DNA sequence data allows our method to make predictions for genes that are not expressed under the measured conditions. Finally, we predict transcription terminators whereas Tjaden *et al.* do not, and our approach can naturally be extended to predict other regulatory elements, such as transcription factor binding sites, as well.

Several other research groups have addressed the task of predicting operons. Ermolaeva *et al.* (2001) use comparisons of gene order in multiple genomes to predict operons. Zheng *et al.* (2002) use information about biochemical pathways to predict metabolism-related operons. Since these methods use different sources of evidence than we do, we view them as complementary.

DATA REPRESENTATION

Our approach to predicting transcription units is based on a probabilistic language model, which describes TUs in terms of three sequences of observations. The first sequence of observations, which we denote by x, consists of the nucleotides of the DNA sequence. The second sequence of observations, which we denote by y, is composed of the similarities between the codon usage biases of neighboring ORFs. The third sequence of observations, which we denote by z, is based on the expression measurements made by the probes on the oligonucleotide array. These three sequences are aligned by associating each codon usage and expression observation with a specific position in the DNA sequence. Table 1 provides a toy example illustrating each these sequences.

For the analysis considered here, we assume that we are given the (predicted) coordinates of every ORF in the genome. The basic units that our model processes when making predictions are *runs* of genes. We define a run to be a sequence of DNA that (i) contains genes only on one strand, and (ii) is bracketed by genes on the opposite strand. Given such a run, we would like our model to predict all of the TUs contained within it. That is, we want the model to identify the sequences of genes that are co-transcribed, along with the precise $5'$ and $3'$ ends of the DNA sequence corresponding to each transcript.

In the remainder of this section we describe in detail how we derive codon-usage and expression observation sequences.

Codon usage sequence

A variety of factors influence an ORF's codon usage properties (Karlin *et al.*, 1998) including some, such as gene function, expression level and evolutionary history, that also influence the grouping of ORFs into transcription units. In previous work we found consideration of codon usage properties to be beneficial to an operon prediction task (Bockhorst *et al.*, 2003).

To derive a sequence of codon usage based observations from a run we first associate each ORF o in the run with a set of codon bias vectors $\{\vec{b}^o_a\}$, one for each amino acid a. Let uvw be a codon that codes for a and n^o_{uvw} be the number of times codon uvw appears in o. Then, the elements of the bias vectors are given by:

$$b^o_{a,uvw} = \hat{f}^o_{(uvw|a)} - \bar{f}_{(uvw|a)}$$

where $\bar{f}_{(uvw|a)}$ is the frequency with which a is encoded by uvw (relative to other codings for a) over the whole genome and

$$\hat{f}^o_{(uvw|a)} = \frac{n^o_{uvw} + \bar{f}_{(uvw|a)}}{\sum_{xyz \in \text{codons}(a)} n^o_{xyz} + 1}$$

is the smoothed frequency with which a is coded for by uvw in o. The sum in the denominator ranges over all codons that code for amino acid a.

Next, we calculate the codon usage similarity between each pair of neighboring ORFs in the run. The codon usage similarity between the two ORFs o and q is defined as:

$$y_i = \text{Sim}(o, q) = \sum_a \vec{b}^o_a \cdot \vec{b}^q_a.$$

Table 1. A toy example of an observation sequence for a region of DNA at an ORF boundary

		intergenic										ORF					
	probes		•			•				•			•			•	
x	**DNA sequence**	c	c	c	g	a	g	a	a	g	a	A	T	G	C	G	T
y	**ORF-ORF codon usage similarity**	–	–	–	–	–	–	–	–	–	–	0.4	–	–	–	–	–
	ORF-ORF expression correlation	–	–	–	–	–	–	–	–	–	–	0.6	–	–	–	–	–
z	**upstream expression correlation**	–	0.8	–	–	0.7	–	–	–	0.4	–	–	–	–	–	–	–
	downstream expression correlation	–	0.3	–	–	0.4	–	–	–	0.2	–	–	–	–	–	–	–

Nucleotides shown in lower case correspond to inter-genic positions and those shown in upper case correspond to coding positions. Bullets on the 'probes' line indicate locations of the center positions of probes. Entries on the 'downstream expression correlation' and 'upstream expression correlation' lines represent correlations between expression measurements at probes and their downstream and upstream ORFs respectively. Entries on these two lines align with probes in inter-genic regions only. The entries on the 'ORF-ORF correlation' and 'ORF-ORF codon usage' lines are aligned with the first nucleotide of ORFs and represent the expression correlation and codon usage similarity between the ORF and the next ORF downstream respectively

This measure is symmetric and reflects both the consistency and degree to which the bias vectors are correlated. We align each such codon usage similarity value with the first nucleotide of the more upstream ORF. In summary, the codon usage based sequence associated with a run is the sequence of codon usage similarities between each pair of neighboring ORFs in the run.

Expression sequence

In addition to DNA and codon usage observation sequences, our model also explains observation sequences based on gene expression data. Affymetrix *E.coli* high-density oligonucleotide arrays were used to obtain the gene expression data we consider here. The design of the array is described elsewhere (Selinger *et al.*, 2000). Briefly, each array contains 295 936 25mer oligonucleotide probes. Half of the probes exactly match oligonucleotides in the *E.coli* genome, and are called *perfect match* (PM) probes. Each PM probe is paired with a *mismatch* (MM) probe which has the same 25mer sequence except that the 13th base is complemented. Every annotated ORF and most inter-genic regions are assayed by a set of PM-MM probe pairs on the array.

Given expression measurements from a set of experiments, we construct a sequence of observations as follows. First, for each probe in a non-coding region, p, and each experiment, k, we determine the probe's expression measurement as the difference between the perfect match intensity and the mismatch intensity.

$$e_p^k = PM_p^k - MM_p^k.$$

Second, for each ORF o and each experiment, we compute an expression value e_o^k by considering all probes contained within it and taking the average of the measurements for these probes.

$$e_o^k = \frac{1}{m_o} \sum_{l=1}^{m_o} \left(PM_{o_l}^k - MM_{o_l}^k \right)$$

Here, m_o represents the number of probes for ORF o, $PM_{o_l}^k$ is the intensity of the lth perfect match probe in o for experiment k, and $MM_{o_l}^k$ is the intensity of the lth mismatch probe in o for experiment k. Third, for each probe that corresponds to a non-coding region, we compute its correlation across all experiments with both of its neighboring ORFs (i.e. the next ORF upstream in the sequence and next ORF downstream).

The correlation between the expression measurements for probe p and ORF o is computed as follows:

$$z_i = \text{corr}(e_p, e_o) = \frac{\sum_{k=1}^{n} (e_p^k - \mu_p)(e_o^k - \mu_o)}{\sqrt{\sum_{k=1}^{n} (e_p^k - \mu_p)^2 \sum_{k=1}^{n} (e_o^k - \mu_o)^2}}$$

where n represents the number of experiments, μ_p is defined as:

$$\mu_p = \frac{1}{n} \sum_{k=1}^{n} e_p^k,$$

and μ_o is defined analogously.

Also, we compute correlations between the expression values for pairs of neighboring ORFs. For each ORF o, except the last one in a run, we compute the correlation between expression measurements for o and expression measurements for the next ORF downstream. In summary, the expression observation sequence for a run consists of probe-ORF and ORF-ORF correlations. We align ORF-ORF correlations with the more upstream ORF of the pair, and we align probe-ORF correlations with the position of the probe in the sequence. For the latter, we consider the position of a probe to be the sequence position that corresponds to the middle (*ie* the 13th) nucleotide of the probe.

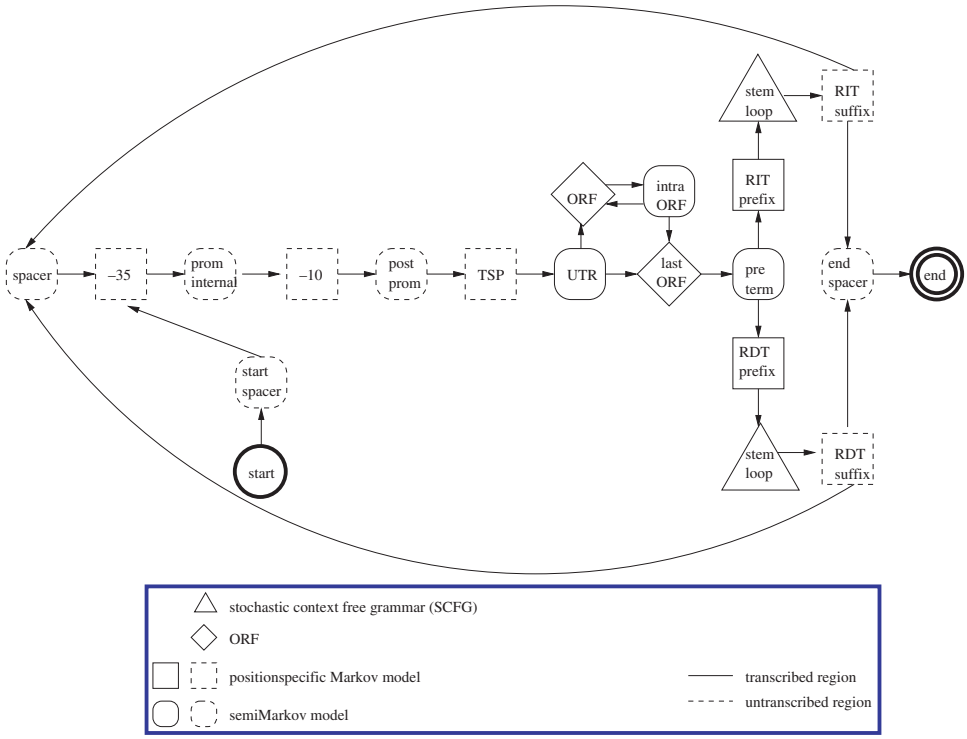

Fig. 1. The structure of our model. Closed shapes represent submodels of the overall model, and arrows represent allowable transitions among the submodels. Each submodel explains DNA sequences of a specified type. The key at the bottom of the figure shows the type of submodel represented by each shape. Solid lines indicate submodels that correspond to transcribed regions of sequence, and dashed lines indicate submodels for untranscribed regions.

THE MODEL

Figure 1 illustrates the overall structure of our model. This model is designed to process a single *run*, as described above. The start and end states, represented by circles in the figure, are silent. Each of the other closed shapes, which we refer to as *submodels*, represents a specific TU feature. For example, the submodel labeled prom-internal represents the region between a promoter's -35 and -10 boxes. The arrows represent allowable transitions among the submodels. The submodels drawn with solid lines correspond to regions transcribed under some conditions and those drawn with dashed lines correspond to regions that are not transcribed.

The submodels contain probability distributions over emissions of nucleotides, codon usage similarities and expression correlations. Since we assume a submodel's DNA sequence, codon usage and expression emissions are conditionally independent given the submodel, a joint distribution over aligned sequences can be represented with separate probability distributions over the three types of sequences.

A submodel's shape in Figure 1 indicates the parameterization of its DNA sequence probability distribution. Each submodel maintains a separate set of parameters for its DNA sequence distribution. However, various submodels share their expression parameters.

Our model has a set of parameters, $\vec{\theta}$, that we use to characterize $\Pr(x, y, z|\vec{\theta})$, the joint probability distribution of various observation sequences. A key assumption of our model is that x, y and z are conditionally independent given an associated path through the model.

DNA sequence model component

There are four types of DNA sequence submodels in our overall model. We do not model the DNA sequence in ORFs because we assume that ORF coordinates are given. We discuss each of the other submodels in turn.

Position specific Markov models: These are first-order Markov models that represent fixed-length segments of sequence. To compute the probability of a given subsequence $x_i, \ldots, x_j$, (denoted $x_{i:j}$) with one of these submodels, we perform the following calculation:

$$\Pr(x_{i:j}|\vec{\theta}) = \Pr(x_i|\vec{\theta}_1) \prod_{l=i+1}^{j} \Pr(x_l|x_{l-1}, \vec{\theta}_{l-i+1}).$$

Table 2. The structure of the terminator grammar

TERM	→	PREFIX STEM_LOOP SUFFIX
PREFIX	→	P_1 P_2 P_3 P_4 P_5 P_6 P_7 P_8 P_9 P_{10}
STEM_LOOP	→	b_l STEM_BOT b_r
STEM_BOT	→	b_l^* STEM_MID b_r^* | b_l^* STEM_TOP2 b_r^*
STEM_MID	→	b_l^* STEM_MID b_r^* | b_l^* STEM_TOP2 b_r^*
STEM_TOP2	→	b_l^* STEM_TOP1 b_r^*
STEM_TOP1	→	b_l LOOP b_r
LOOP	→	B B B LOOP_END
LOOP_END	→	B LOOP_END | B
SUFFIX	→	S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8 S_9 S_{10}
B	→	a | c | g | t
P_i	→	a | c | g | t
S_i	→	a | c | g | t

Nonterminals are capitalized and the terminals are a, c, g and t. The notation X → Y | Z is shorthand for the two productions X → Y and X → Z. Productions containing b_l and b_r are shorthand for the family of 16 productions where b_l and b_r can be any of the four terminals. Productions containing b_l^* and b_r^* have a similar interpretation except that b_l^* and b_r^* can also be null allowing for unpaired bases in the interior of the stem

Here, $\vec{\theta}$ is the set of parameters that characterizes the model and $\vec{\theta}_i$ is the subset of parameters that specifies the nucleotide probabilities at the ith position in the model; $\vec{\theta}_1$ comprises four parameters, and each of the $\vec{\theta}_i$, for $i > 1$, comprises 16 parameters.

Semi-Markov models: These are first-order Markov models that represent variable-length segments of sequence. With these submodels, we compute the probability of a given subsequence $x_{i:j}$ of length L as follows:

$$\Pr(x_{i:j}|\vec{\theta}) = \Pr(L|\vec{\theta}_L)\Pr(x_i|\vec{\theta}_0)\prod_{l=i+1}^{j}\Pr(x_l|x_{l-1},\vec{\theta}_0).$$

Here we have a set of parameters, $\vec{\theta}_L$, to model the length of the sequence being considered, and set of parameters, $\vec{\theta}_0$, to model the DNA bases (irrespective of their position) in the sequence.

Stochastic context free grammars (SCFGs): We use SCFGs to model the two classes of transcription terminators. The key feature of a terminator sequence is that, when it is transcribed to RNA, a stem-loop structure is formed. In order to detect potential stem-loop structures in genomic DNA sequences, we need to be able to represent dependencies between non-neighboring pairs of bases that are separated by variable distances. SCFGs can naturally represent such dependencies, and thus have been used for a number of RNA modeling tasks (Eddy and Durbin, 1994; Sakakibara *et al.*, 1994; Brown, 2000).

We use the same grammar structure, shown in Table 2, separately parameterized to model rho-independent (RIT) and rho-dependent (RDT) terminators. Note that the grammar in this table includes productions for the prefix, stem loop and suffix submodels shown in Figure 1. We designed the structure of our grammar to represent many of the prototypical properties of rho-independent terminators identified in the literature (Brendel *et al.*, 1986; Carafa *et al.*, 1990). These features include: (i) a variable length stem with a minimum of four base pairs and allowing bulges, (ii) a variable length loop with a minimum of four bases, (iii) a preference for specific base pairs at the top and base of the stem, and (iv) the individual preferences of the 10 bases preceding and following the stem–loop.

Abstractly, our SCFG models compute the probability of a subsequence $x_{i:j}$ by summing over all possible parses of the sequence:

$$\Pr(x_{i:j}|\vec{\theta}) = \sum_{\pi}\Pr(x_{i:j},\pi|\vec{\theta}_s).$$

Here, $\vec{\theta}_s$ represents the parameters of the SCFG, and π ranges over the possible parses of $x_{i:j}$. The SCFG parameters are the probabilities associated with the grammar's productions. We do not explicitly enumerate the possible parses but use the Inside algorithm (Lari and Young, 1990) to calculate $\Pr(x_{i:j}|\vec{\theta})$.

Codon usage model component

The observations explained by the codon usage component of our model are the ORF-ORF codon usage similarities. These observations are only emitted by the ORF and last ORF submodels. The ORF submodel emits codon usage similarity measures between ORFs in the same TU while the last ORF submodel emits codon usage similarity measures between ORFs that are in different TUs. We model these two cases with separate probability distributions parameterized by $\vec{\theta}_{c,S}$ and $\vec{\theta}_{c,N}$ where:

$$\vec{\theta}_{c,S} = \Pr(\mathrm{Sim}(o,q) \mid o \text{ and } q \text{ in same TU})$$
$$\vec{\theta}_{c,N} = \Pr(\mathrm{Sim}(o,q) \mid o \text{ and } q \text{ NOT in same TU}).$$

Here o and q represent ORFs. We use histograms to represent the probability distributions so the $\vec{\theta}$ parameters refer to bin-specific probabilities.

Expression model component

The observations explained by the expression component of our model are the ORF-ORF and probe-ORF expression correlations. Like the codon usage observations, the ORF-ORF expression correlations are only emitted by the ORF submodel and last ORF submodel. The ORF submodel emits correlations between ORFs in the same TU and last ORF emits correlations between ORFs in different

TUs. As above, we model these two cases with separate probability distributions parameterized by $\vec{\theta}_{ORF,S}$, and $\vec{\theta}_{ORF,N}$ where

$$\vec{\theta}_{ORF,S} = \Pr(\text{corr}(e_o, e_q) \mid o \text{ and } q \text{ in same TU})$$
$$\vec{\theta}_{ORF,N} = \Pr(\text{corr}(e_o, e_q) \mid o \text{ and } q \text{ NOT in same TU})$$

and $\text{corr}(e_o, e_q)$ is the ORF-ORF expression correlation between ORFs o and q.

All submodels *except* ORF and last ORF may emit probe-ORF expression correlations. We separately model the distribution of probe-ORF expression correlations for the cases that the probe and ORF are in, and are not in, the same TU. We use the latter case both when the probe position is in a different TU than the ORF and when the probe position is not in any TU.

We parameterize these distributions with $\vec{\theta}_{\text{probe},S}$, $\vec{\theta}_{\text{probe},N}$ where

$$\vec{\theta}_{\text{probe},S} = \Pr(\text{corr}(e_p, e_o) \mid p \text{ and } o \text{ in same TU})$$
$$\vec{\theta}_{\text{probe},N} = \Pr(\text{corr}(e_p, e_o) \mid p \text{ and } o \text{ NOT in same TU})$$

and $\text{corr}(e_p, e_o)$ is the probe-ORF expression correlation between probe p and ORF o.

We associate each submodel which emits probe-ORF expression correlations with the appropriate parameter set. For example, the UTR submodel uses $\vec{\theta}_{\text{probe},N}$ for upstream probe-ORF expression correlations and $\vec{\theta}_{\text{probe},S}$ for downstream probe-ORF expression correlations.

We assume that expression observations are conditionally independent given the submodel. Thus, given expression measurements for some subsequence, $z_{i:j}$, we compute $Pr(z_{i:j}|\vec{\theta})$ by taking the product of the appropriate θ parameters. Here, $z_{i:j}$ refers to all expression observations aligned to positions in $x_{i:j}$.

Training

Our training data consists of runs whose TUs are partially annotated from known operons, promoters and terminators. Here, an operon refers to a set of co-transcribed genes, a promoter indicates a transcription start site (TSS) and a terminator indicates the end point of a TU. Often, the annotation associated with a TU is incomplete. For example, the TSS associated with a known operon may be unknown.

We train the sequence and expression models and their components independently by extracting training examples appropriate for the individual components. From the operons, promoters and terminators associated with the runs in a training set, we extract sequences that correspond to various submodels or sets of submodels. These sequences are used to train the submodels to which they are aligned. The Baum-Welch and Inside-Outside algorithms are used for training the Markov models and SCFGs respectively. These training algorithms are instances of the expectation-maximization algorithm and as such converge to a local maximum. Consequently, for our terminator SCFGs we use domain knowledge to set strong priors on its parameters in an attempt to start the Inside-Outside training procedure in a region of parameter space with a good local maximum.

We represent the conditional probability distributions in our codon usage and gene expression components of our models with histograms. To set the bin cut-points and probabilities we first use known operons and transcription start and end sites to assemble a set of codon usage and expression observations known to be either same TU or non-same TU observations. Next, we set the cut-points such that an (approximately) equal number of observations fall in each bin. Finally, we set the bin probabilities using Laplace estimates.

Inference

We wish to find the most probable partitioning of a run into transcription units given its DNA sequence, x, codon usage observations, y, expression observations, z, and ORFs. Inference begins by filling the matrices α_x^T, α_x^N, α_y^T, α_y^N, α_z^T, and α_z^N which hold the probabilities of subsequences of x, y and z given the extent of TUs and untranscribed regions. Let $x_{i:j}$ refer to the DNA subsequence $x_i, \ldots, x_j$, $y_{i:j}$ refer to all codon usage observations aligned to positions in $x_{i:j}$ and $z_{i:j}$ refer to all expression observations aligned to positions in $x_{i:j}$. The α values are defined as follows:

$$\alpha_x^T(i, j) = \Pr(x_{i:j} | x_{i:j} \text{ is complete TU})$$
$$\alpha_x^N(i, j) = \Pr(x_{i:j} | x_{i:j} \text{ is complete untranscribed region})$$
$$\alpha_y^T(i, j) = \Pr(y_{i:j} | x_{i:j} \text{ is complete TU})$$
$$\alpha_y^N(i, j) = \Pr(y_{i:j} | x_{i:j} \text{ is complete untranscribed region})$$
$$\alpha_z^T(i, j) = \Pr(z_{i:j} | x_{i:j} \text{ is complete TU})$$
$$\alpha_z^N(i, j) = \Pr(z_{i:j} | x_{i:j} \text{ is complete untranscribed region}).$$

We compute them using a method similar to the Inside algorithm for SCFGs. Under the assumption that the DNA sequence and expression observations are conditionally independent given the locations of the transcription units, the most probable parse can be computed efficiently from the α values with a dynamic program similar to the Viterbi algorithm for HMMs. Note that in computing the α values, we consider all possible sub-parses that could account for a subsequence either being a TU or being an untranscribed region. However, in the second step of the inference algorithm, in which we are deciding how to partition a run into transcription units, we compute and return only the most probable parse.

EMPIRICAL EVALUATION

In this section we empirically evaluate how accurately our approach predicts transcription units. Our experiments are designed to test two hypotheses. First, we hypothesize that we can obtain more accurate predictions when we use both sequence and expression data than when we use either data source alone. Second, we hypothesize that our approach is as accurate as a state-of-the-art operon prediction method, even though we consider a more involved prediction task here.

Methodology

Our initial data set consists of 1292 runs, 545 operons, 401 promoters and 211 terminators. Much of this data is from the annotated sequence of *E.coli* (Blattner *et al.*, 1997) and the EcoCyc database (Karp *et al.*, 2002). From this set, we exclude 62 promoters and 38 terminators that overlap ORFs, since our current model restricts its parses to inter-genic regions and is unable to represent such signals.

We use expression data from 53 experiments in our experiments. Many of the experiments were replicated, resulting in data from 110 arrays. For these experiments, *E.coli* strain MG1655 and several mutant strains were grown under various conditions and RNA from each culture was hybridized on an array. Cells were generally grown aerobically in Neidhardt's MOPS minimal medium (Neidhardt *et al.*, 1974) with glucose as the carbon source. Variations included cultures grown anaerobically, with various carbon sources, under different stress conditions (e.g. heat, cold, acid and antibiotic treatments), and time series experiments such as the transition from mid-log phase to late stationary phase.

Additionally, we have three arrays to which genomic DNA was hybridized. Using expression measurements from these arrays, we attempted to find probes that could mislead our models by identifying those probes whose expression measurements on all three arrays deviated from the array-wide average by greater than a specified number of standard deviations. The accuracy of our models when such identified probes were removed from our analysis was always equal to, or worse than, the accuracy of the models when all probes were used. In this article, we report results from runs of our method that use all probes.

We associate each operon, promoter and terminator with the run in which it occurs. We partition the runs into ten disjoint sets (folds), and use a 10-fold cross-validation methodology. Thus, each run and its associated regulatory elements are in a single test set. For each run in a test set, our method predicts the operons, promoters and terminators that are present in the run (all of the known ones are hidden). We evaluate the accuracy of our predictions using the known operons, promoters and terminators in each test set. The results we report are aggregates from all ten folds.

We run our method three times varying the types of observation sequences available as input data.

- sequence-only: The models are given only those observations that are derived from the DNA sequence, that is, the codon usage and DNA sequence observations.
- expression-only: The models are given only observations that are derived from the expression arrays.
- sequence+expression: The models are given the observations derived from the DNA sequence and the observations derived from the expression arrays.

RESULTS

The first evaluation criterion we consider is classification accuracy. For each input-data variant we consider, we have our models find the most probable parse for each run in the test sets. Each of these parses provides a set of predictions for the run, since the parse specifies where each predicted transcription unit begins and ends.

Table 2 shows classification accuracy results for the sequence-only, expression-only and sequence+expression models. A *positive* instance of an operon is a sequence of genes that is transcribed as a unit under some conditions, whereas a *negative* instance is a sequence of genes that shares at least one gene with a known operon but is itself not a known operon. A correct positive operon prediction involves exactly identifying a sequence of genes in a known operon. Note that under this stringent criterion, a classification is considered either right or wrong; there is no partial credit for cases in which we predict the extent of an operon almost correctly. A *positive* promoter (terminator) instance is an inter-genic region that contains a promoter (terminator), whereas a *negative* instance is an inter-genic region, between two genes in the same known operon, that does not contain a known promoter (terminator). Note that given these definitions, some predictions, such as a predicted operon whose ORFs do not belong to any known operon, cannot be classified as either correct or incorrect.

For the case of operons, the results in Table 3 show that it is clearly beneficial to use both sequence and expression data. The sequence+expression models are superior to the sequence-only models in terms of sensitivity, specificity and precision. A paired t-test indicates that the differences between the two sets of models for all three measures are significant at greater than a 95% confidence level. The expression-only models have relatively good sensitivity but they make more false positive predictions, as exhibited by their lower precision.

For the case of promoters and terminators, the expression-only models again provide good sensitivity, but make many more false positive predictions than the

Table 3. Classification accuracy for the models that use only sequence data, only expression data and both sources of data

		sequence-only	expression-only	sequence+expression
operons	TP	274	322	306
	FN	271	223	239
	FP	213	257	191
	TN	11612	11568	11634
	sensitivity	50.3 %	59.1 %	56.1 %
	specificity	98.2 %	97.8 %	98.4 %
	precision	56.3 %	55.6 %	61.6 %
promoters	TP	285	304	297
	FN	54	35	42
	FP	8	47	10
	TN	245	206	243
	sensitivity	84.1 %	89.7 %	87.6 %
	specificity	96.8 %	81.4 %	96.0 %
	precision	97.3 %	86.6 %	96.7 %
terminators	TP	132	150	141
	FN	41	23	32
	FP	9	48	11
	TN	247	208	245
	sensitivity	76.3 %	86.7 %	81.5 %
	specificity	96.5 %	81.3 %	95.7 %
	precision	93.6 %	75.8 %	92.8 %

TP = true positive, FN = false negative, FP = false positive, TN = true negative, sensitivity = $\frac{TP}{TP+FN}$, specificity = $\frac{TN}{TN+FP}$, precision = $\frac{TP}{TP+FP}$

other two models. This suggests a tendency of these models to predict TUs with relatively few ORFs. The sequence-only and the sequence+expression models have similar accuracy profiles for promoters and terminators. The sequence+expression models correctly identify 12 more promoters and nine more terminators than the sequence-only models, but also make two more false positive promoter and terminators predictions.

For the true-positive promoter and terminator predictions represented in Table 3, Figure 2 shows how accurately our models localize the true signals. The left side of the figure shows a histogram for promoter TP predictions, and the right side shows a histogram for terminator TP predictions. Each histogram shows the distribution in terms of where the predicted signal locations are relative to actual promoter/terminator locations. The reference point we use for actual promoter locations is the transcription start site (TSS), and the reference point for terminator locations is the $3'$ end of the stem loop. For example, the coordinate 0 in the left histogram corresponds to promoter predictions that exactly identify the TSS, and the coordinate 10 corresponds to predictions that are 10 bases downstream from the true TSS.

Figure 2 shows that the expression-only models are not able to accurately localize the position of predicted promoters and terminators. The models that use sequence data are clearly much better at localizing the positions of promoters and terminators. The sequence+expression models exactly localize more promoters than the sequence-only models. Of the 307 promoter predictions made by the sequence+expression model 131 (42.7%) are correct predictions and within $\pm$ 3 bases of the true transcription start site. This compares to 106 of 293 (36.2%) for the sequence-only model. However, the sequence-only models are slightly better at localizing terminators. This may be because the reference point used for terminators, the $3'$ end of the stem loop, may not be the exact point where transcription stops.

Finally, we compare the accuracy of our models' operon predictions to the predictions made by a state-of-the-art model (Bockhorst *et al.*, 2003) that predicts only operons, not complete transcription units. Figure 3 shows the ROC (receiver operating characteristic) curve (Eagen, 1975) for a Bayesian network operon prediction model that we developed in earlier work, along with points plotting the sensitivity vs. specificity behavior of our sequence-only, expression-only and sequence+expression models. Points on the ROC curve are generated by varying the threshold on the posterior probability computed by the Bayesian network; this threshold separates positive from negative predictions. A model that guessed randomly would result in an ROC 'line' defined by: TP rate = FP rate. This figure illustrates that both our expression-only and sequence+expression models

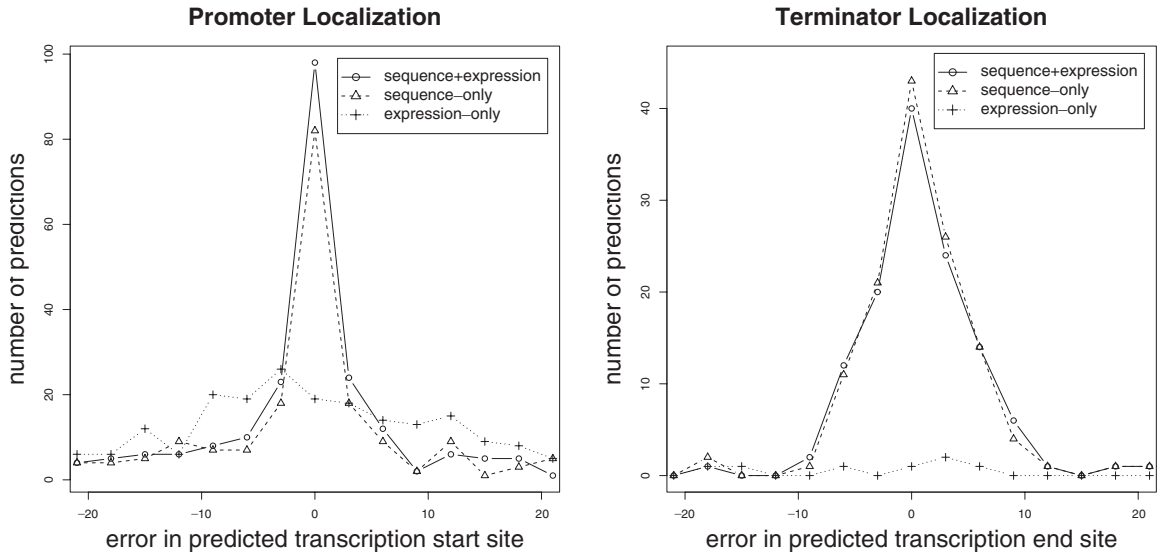

Fig. 2. Histograms of errors in predicted transcription start positions and predicted transcription termination positions. Histograms are shown for all three models considered in the experiments.

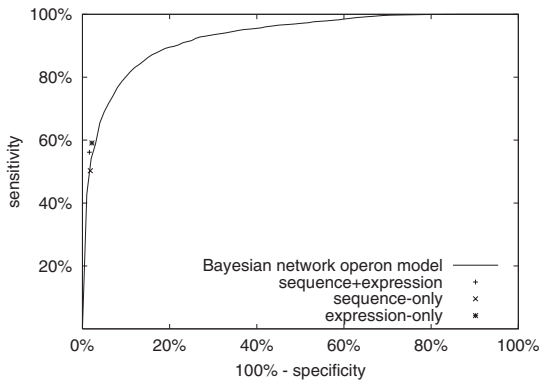

Fig. 3. ROC curve for our Bayes net operon model and operating points for the sequence-only, expression-only and sequence+expression models.

provide predictive accuracy that is slightly better than the Bayesian network operon model, even though these models are addressing a more detailed task. Note that we do not plot ROC curves for the models presented in this paper because we compute only the best parse for each test run. In future work, we plan to extend our inference procedure such that we can compute ROC curves.

CONCLUSIONS

We have presented a method for predicting transcription units in bacterial genomes using both sequence and expression data. Our method builds on our previous work (Craven *et al.*, 2000; Bockhorst *et al.*, 2003) in that it provides a coherent set of predictions for operons, promoters and terminators, and it predicts the complete extent of transcription units. Our method extends the recent work of Tjaden *et al.* (2002) in that is uses DNA sequence data, in addition to expression data, to predict transcription units. Similarly, our method extends that of Yada *et al.* (1999), which uses only sequence data to predict transcription units.

Our experiments indicate several key results. First, the combination of DNA sequence data and expression data results in more accurate predictions than either alone. Second, our models can predict promoters, terminators and operons with high accuracy. Third, the accuracy of our operon predictions is slightly better than a state-of-the-art method that predicts only which sequences of genes constitute operons, not the extent of transcription units.

A notable limitation of our current models is that they are based on a simplified view of transcription. In particular, they cannot represent multiple promoters in a single inter-genic region, transcription units that overlap each other, and regulatory elements that overlap ORFs or other regulatory elements. We are currently developing extensions to our approach to address these issues. Additionally, we are extending the 'vocabulary' of regulatory elements represented in our models. A key feature of our approach is that it can be easily extended to incorporate additional types of sequence elements. We envision developing our method into a comprehensive

model of the sequence-based aspects of prokaryotic gene regulation.

ACKNOWLEDGEMENTS

This research was supported in part by NSF grant IIS-0093016 (JB, MC) and NIH grants T15-LM07359-01 (JB), R01-GM35682-15A1 (YQ, JG, ML, FB), R44-HG02193-02 (YQ, JG, ML, FB), and R01-LM07050-01 (MC).

REFERENCES

Blattner,F.R., Plunkett,G., Bloch,C.A., Perna,N.T., Burland,V., Riley,M., Collado-Vides,J., Glasner,J.D., Rode,C.K. and Mayhew,G.F. (1997) The complete genome sequence of *Escherichia coli* K-12. *Science*, **277**, 1453–1474.

Bockhorst,J., Craven,M., Page,D., Shavlik,J. and Glasner,J. (2003) A Bayesian network approach to operon prediction. *Bioinformatics*, (in Press).

Brendel,V., Hamm,G. and Trifinov,E. (1986) Terminators of transcription with RNA polymerase from *Escherichia coli*: what they look like and how to find them. *J. Biomol. Struct. Dynamics*, **3**, 705–723.

Brown,M.P.S. (2000) Small subunit ribosomal RNA modeling using stochastic context-free grammars. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, La Jolla, CA, pp. 57–66.

Carafa,Y.A., Brody,E. and Thermes,C. (1990) Prediction of rho-independent *Escherichia coli* transcription terminators: a statistical analysis of their RNA stem-loop structures. *J. Mol. Biol.*, **216**, 835–858.

Craven,M., Page,D., Shavlik,J., Bockhorst,J. and Glasner,J. (2000) A probabilistic learning approach to whole-genome operon prediction. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, La Jolla, CA, pp. 116–127.

Eagen,J.P. (1975) *Signal Detection Theory and ROC analysis*. Academic Press, New York.

Eddy,S.R. and Durbin,R. (1994) RNA sequence analysis using covariance models. *Nucleic Acids Res.*, **22**, 2079–2088.

Ermolaeva,M., Khalak,H., White,O., Smith,H. and Salzberg,S. (2000) Prediction of transcription terminators in bacterial genomes. *J. Mol. Biol.*, **301**, 27–33.

Ermolaeva,M., White,O. and Salzberg,S. (2001) Prediction of operons in microbial genomes. *Nucleic Acids Res.*, **29**, 1216–1221.

Karlin,S., Mrázek,J. and Campbell,A. (1998) Codon usages in different gene classes of the *Escherichia coli* genome. *Mol. Microbiol.*, **6**, 1341–1355.

Karp,P., Riley,M., Saier,M., Paulsen,I., Collado-Vides,J., Paley,S., Pellegrini-Toole,A., Bonavides,C. and Gama-Castro,S. (2002) The EcoCyc database. *Nucleic Acids Res.*, **30**, 56–58.

Lari,K. and Young,S.J. (1990) The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, **4**, 35–56.

Moreno-Hagelsieb,G. and Collado-Vides,J. (2002) A powerful non-homology method for the prediction of operons in prokaryotes. *Bioinformatics*, **18** (Suppl. 1), S329–S336.

Neidhardt,F., Bloch,P. and Smith,D. (1974) Culture medium for enterobacteria. *J. Bacteriol.*, **119**, 736–747.

Pedersen,A., Baldi,P., Brunak,S. and Chauvin,Y. (1996) Characterization of prokaryotic and eukaryotic promoters using hidden Markov models. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, St. Louis, MO, pp. 182–191.

Sakakibara,Y., Brown,M., Hughey,R., Mian,I.S., Sjölander,K., Underwood,R.C. and Haussler,D. (1994) Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Res.*, **22**, 5112–5120.

Salgado,H., Moreno-Hagelsieb,G., Smith,T.F. and Collado-Vides,J. (2000) Operons in *Escherichia coli*: genomic analyses and predictions. *Proc. Natl Acad. Sci. USA*, **97**, 6652–6657.

Selinger,D., Cheung,K., Mei,R., Johansson,E., Richmond,C., Blattner,F., Lockhart,D. and Church,G. (2000) RNA expression analysis using a 30 base pair resolution *Escherichia coli* genome array. *Nat. Biotechnol.*, **18**, 1262–1268.

Thieffry,D., Salgado,H., Huerta,A. and Collado-Vides,J. (1998) Prediction of transcriptional regulatory sites in the complete genome sequence of *Escherichia coli* K-12. *Bioinformatics*, **14**, 391–400.

Tjaden,B., Haynor,D., Stolyar,S., Rosenow,C. and Kolker,E. (2002) Identifying operons and untranslated regions of transcripts using *Escherichia coli* RNA expression analysis. *Bioinformatics*, **18** (Suppl. 1), S337–S344.

Yada,T., Nakao,M., Totoki,Y. and Nakai,K. (1999) Modeling and predicting transcriptional units of *Escherichia coli* genes using hidden Markov models. *Bioinformatics*, **15**, 987–993.

Zheng,Y., Szustakowski,J., Fortnow,L., Roberts,R. and Kasif,S. (2002) Computational identification of operons in microbial genomes. *Genome Res.*, **12**, 1221–1230.