Entropic Graph-based Posterior Regularization: Extended Version

Maxwell W Libbrecht, Michael M Hoffman, Jeffrey A Bilmes, William S Noble

May 18, 2015

Abstract

Graph smoothness objectives have achieved great success in semi-supervised learning but have not yet been applied extensively to unsupervised generative models. We define a new class of entropic graph-based posterior regularizers that augment a probabilistic model by encouraging pairs of nearby variables in a regularization graph to have similar posterior distributions. We present a three-way alternating optimization algorithm with closed-form updates for performing inference on this joint model and learning its parameters. This method admits updates linear in the degree of the regularization graph, exhibits monotone convergence, and is easily parallelizable. We are motivated by applications in computational biology in which temporal models such as hidden Markov models are used to learn a human-interpretable representation of genomic data. On a synthetic problem, we show that our method outperforms existing methods for graph-based regularization and a comparable strategy for incorporating long-range interactions using existing methods for approximate inference. Using genome-scale functional genomics data, we integrate genome 3D interaction data into existing models for genome annotation and demonstrate significant improvements in predicting genomic activity.[1]

Graph-based methods have recently been successful in solving many types of semi-supervised learning problems (Chapelle et al., 2006; Das & Smith, 2011; Joachims, 1999; Subramanya et al., 2010; **?**; Subramanya & Bilmes, 2011; Zhu et al., 2004; Zhu & Ghahramani, 2002). These methods assume that data instances lie in a low-dimensional manifold that may be represented as a graph. They optimize a *graph smoothness* criterion, which states that data instances nearby in the graph should be more likely to receive the same label. In a semi-supervised learning setting, optimizing this criterion has the effect of spreading labels from labeled to unlabeled instances.

Despite the success of graph-based methods for semi-supervised learning, there has not been as much study of the use of graph smoothness objectives in an unsupervised setting. In unsupervised problems, we do not have labels but instead have a generative model that is assumed to explain the observed data given the latent labels. While some types of relationships between instances (for example, the relationship between neighboring words in a sentence or neighboring bases in a genome) can easily be incorporated into the generative model, it is often inappropriate to encode a graph smoothness assumption into the model this way, for two reasons. First, in some cases, it is not clear what probabilistic process generated the labels with respect to the graph. Some objectives and distance measures that are successful for semi-supervised learning do not have probabilistic analogues. Second, large models must obey factorization properties (e.g., a tree or chain as in hidden Markov models) to facilitate the use of efficient dynamic programming algorithms such as belief propagation. Graphs representing similarity between variables do not in general satisfy these structure requirements because they tend to be densely clustered, leading to very high-order factors.

In this paper, therefore, we propose a new regularization approach for expressing a graph smoothness objective over a probabilistic model. We employ the posterior regularization (PR) framework of Ganchev et al. (2010), in which a probabilistic model is regularized through a term defined on an auxiliary posterior distribution variable. We define a powerful posterior regularizer which encourages pairs of variables to have similar posterior distributions by adding a penalty based on their Kullback-Leibler (KL) divergence. The pairs of penalized variables are encoded in a regularization graph which may be entirely different from the graphical model on which inference is performed. This regularizer graph need not have low treewidth and admits efficient optimization even when fully connected. We call our strategy of adding KL regularization penalties *entropic graph-based posterior regularization* (EGPR).

We show that inference and learning using this regularizer can be performed efficiently using a three-way alternating optimization algorithm with closed-form updates. This algorithm alternates between (1) smoothing marginal posteriors according to a regularization similarity graph, (2) performing probabilistic inference in a graphical model with the same dependence structure as the unregularized model, and (3) updating model parameters. The updates are linear in the degree

[1] So that the supplementary material appears in context, this supplement includes the text that appears in the main version in grey. The text that appears only in the extended version is shown in black.

of the regularization graph and are easily parallelizable (**?**), in our experiments scaling to tens of millions of variables. We show that this procedure corresponds to a generalization of the EM algorithm.

We apply this approach to improve existing methods for annotating the human genome (Day et al., 2007; Hoffman et al., 2012a; Ernst & Kellis, 2010). Methods for genome annotation distill genomic data into a human-interpretable form by simultaneously partitioning the genome into non-overlapping segments and assigning labels to each segment. This type of analysis has recently had great success in interpreting the function of the human genome and formed an integral part of the analysis of the NIH-sponsored ENCODE project ((ENCODE Project Consortium, 2012; Hoffman et al., 2012b), `http://www.nature.com/encode`).

However, exiting annotation methods use temporal models such as hidden Markov models and therefore cannot efficiently incorporate data on the genome's 3D structure. This 3D structure has been shown to play a key role in gene regulation and other genomic processes. In our experiments on synthetic data, a model using EGPR outperforms comparable models using either other regularization strategies (e.g., squared error) or loopy belief propagation. On ENCODE data, a model using EGPR predicts genome activity much more accurately than the currently-used chain models as well as other forms of regularizer. Thus EGPR provides a method for jointly modeling genome activity and 3D structure.

1 Proposed Method

In an unsupervised learning problem, we are given a set of vertices V that index a set of $n = |V|$ random variables $X_V = \{X_1, \ldots, X_n\}$ and a conditional dependence graph $G = (V, E)$. The graphical model describes a probability distribution parameterized by θ that can be factorized as $p_\theta(x_V) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \phi_\theta^{(C)}(x_C)$ where each $C \subseteq V$ is a fully connected clique in G. We denote random variables with capital letters (e.g., X_H) and instantiations of variables with lower-case (e.g., $x_H \in \text{domain}(X_H)$). We also use capitals to denote sets and lowercase to denote set elements (e.g., X_h for $h \in H$).

Training graphical models involves a set of observed data $\bar{x}_O$, where a subset of variables $O \subseteq V$ is observed and the remainder $H = V \setminus O$ are hidden. In general we may have many samples of observed data, but for notational simplicity, in this work we assume the graphical model expresses the independence between samples, so it is sufficient to assume just one sample.

In this work we require that every hidden variable have the same domain $\mathcal{X}$, although our results can easily be extended to a case where we have several sets of variables, each with a common domain. When the probability distribution is governed by a set of parameters θ, penalized maximum likelihood training corresponds to the optimization

$$\text{maximize}_\theta \qquad J(\theta) \triangleq \mathcal{L}(\theta) + \mathcal{R}(\theta) \tag{1}$$

$$\text{where } \mathcal{L}(\theta) \triangleq \log p_\theta(\bar{x}_O) = \log \sum_{x_H} p_\theta(x_H, \bar{x}_O), \tag{2}$$

and where $\mathcal{R}(\theta)$ is a regularizer that expresses prior knowledge about the parameters. Many regularizers are used in practice, such as the ℓ_2 or ℓ_1 norms, which encourage parameters to be small or sparse, respectively.

Instead of placing a regularizer on the parameters themselves, it is often more natural to place a regularizer on the posterior distribution, a technique called *posterior regularization* (Ganchev et al., 2010). This is done by introducing an auxiliary joint distribution $q(X_H)$, placing a regularizer on $q(X_H)$, and encouraging q to be similar to p_θ via a KL divergence penalty. The regularizer is

$$\mathcal{R}_{\text{PR}}(\theta) \triangleq \max_q \mathcal{R}'_{\text{PR}}(\theta, q) \tag{3}$$

$$\mathcal{R}'_{\text{PR}}(\theta, q) \triangleq -D(q(X_H) \| p_\theta(X_H | \bar{x}_O)) + \mathcal{PR}(q), \tag{4}$$

where $D(\cdot\|\cdot)$ is the KL divergence $D(p(X)\|q(X)) = \sum_x p(x) \log(p(x)/q(x))$ and $\mathcal{PR}(q)$ is a penalty term that expresses some prior knowledge about the posterior distribution. For notational convenience, we also define $J'(\theta, q) \triangleq \mathcal{L}(\theta) + \mathcal{R}'(\theta, q)$. Ganchev et al. (2010) showed how to optimize this combined objective efficiently when $\mathcal{PR}(q)$ is a sum of terms over individual cliques in the model. Such regularizers can be used for constraining the posterior of individual variables in expectation, among other applications. However, graph smoothness objectives cannot be expressed this way, because they involve arbitrary pairs of variables.

When we have a graph smoothness assumption, we are given a weighted, undirected regularization graph over the hidden variables $G_R = (H, E_R)$, where $E_R \subseteq H \times H$ is a set of edges with non-negative similarity weights $w : E_R \to \mathbb{R}_+$, such that a large $w(u, v)$ indicates that we have strong belief that X_u and X_v should be similar. The regularization graph G_R is entirely separate from the conditional dependence graph G and, in particular, need not obey any decomposition or factorization properties to admit efficient inference.

He et al. (2013) introduced a regularizer of the following form. Let λ_G be a hyperparameter controlling the strength of regularization. The regularizer is

$$\mathcal{PR}_{\text{GPR}}(q) \triangleq -\lambda_G \sum_{(u,v)\in E_R} w(u,v)\|q(X_u) - q(X_v)\|_2^2.$$

He et al. showed how to optimize this regularizer using an exponentiated gradient descent method.

Although this regularizer shows good results for some problems, the use of squared error—or indeed any p-norm—to represent dissimilarity between probability distributions can be highly suboptimal, as we demonstrate empirically in Sections 5 and 6. Squared error is based on a Gaussian error model, which is not appropriate for probability values, and it under-penalizes differences between small probability values. p-norms are defined over all real numbers, while posteriors must lie within the range $[0, 1]$ (and live in a simplex).

A more justified way to measure divergence between probability distributions is to employ the KL divergence. The KL divergence measures the difference of exponents in the probability and so evaluates differences between small and large probabilities more uniformly. Also, Pinsker's inequality (Csiszár & Tusnády, 1984) combined with the relationship of ℓ-norms implies that $D(p\|q) \geq \frac{1}{2}\|p-q\|_1^2 \geq \frac{1}{2}\|p-q\|_\ell^2$, for all $\ell \geq 1$, where $\|\cdot\|_\ell$ is the ℓ-norm. Hence, minimizing KL divergence minimizes an upper bound on all ℓ-norms.

As a concrete example, consider two pairs of probability distributions over two events: $p_1 = [0.55, 0.45]$ vs. $q_1 = [0.45, 0.55]$, and the second pair $p_2 = [0.1, 0.9]$ vs. $q_2 = [10^{-10}, 1-10^{-10}]$. The first pair of distributions (p_1, q_1) are fairly similar, with both events being roughly equally likely. The second pair (p_2, q_2) is quite dissimilar, with the first event being reasonably likely in the first case p_2 and astronomically unlikely in the second q_2. Squared error actually regards the first pair as more dissimilar than the second pair, while KL divergence identifies the second pair as much more dissimilar. Despite the advantages of KL divergence, although all posterior regularization objectives include a KL term binding q to be similar to p_θ, to our knowledge, no existing methods define the posterior regularizer **itself** using KL. Results in Sections 5 and 6 will show, moreover, that KL significantly improves over squared error across the board.

In this work, we propose such a posterior regularizer, which we term *entropic graph-based posterior regularization* (EGPR). The posterior regularizer is

$$\begin{aligned}\mathcal{PR}_{\text{EGPR}}(q) &\triangleq \\ &-\lambda_G \sum_{(u,v)\in E_R} w(u,v) D(q(X_u)\|q(X_v)),\end{aligned} \tag{5}$$

and $J'_{\text{EGPR}}(\theta, q)$ and $\mathcal{R}'_{\text{EGPR}}(\theta, q)$ are defined according to Equations (2) and (4) respectively using the corresponding regularizers. That is,

$$\begin{aligned}&\text{maximize}_{\theta,q} \quad J'_{\text{EGPR}}(\theta, q) \triangleq \mathcal{L}(\theta) + \mathcal{R}'_{\text{EGPR}}(\theta, q),\\ &\mathcal{R}'_{\text{EGPR}}(\theta, q) \triangleq -D(q(X_H)\|p_\theta(X_H|\bar{x}_O)) + \mathcal{PR}_{\text{EGPR}}(q).\end{aligned}$$

The KL divergence in Equation 5 is symmetrized because G_R is undirected—that is $w(u,v) = w(v,u)$—so $D(q_u\|q_v)$ and $D(q_v\|q_u)$ appear with the same weight in the regularizer.

In the next section, we describe a novel alternating optimization algorithm for solving Equation (1) with an EGPR regularizer. Unlike other recent prominent examples of alternating optimization in machine learning (Wang et al., 2008), each update of this method has a closed-form solution. EGPR can be employed either as a regularizer for training the parameters or for inference directly. In the training case, an EM-like algorithm described in Section 2.3 is used to compute and output θ, which can then be used for inference either with or without EGPR. In the inference case, q is computed and output as the posterior marginals, as described in Section 3.

2 EGPR for Training

We first describe how to compute $\text{argmax}_q J'_{\text{EGPR}}(\theta, q)$, then describe how this algorithm can be used in combination with an EM-like algorithm for learning θ.

2.1 Optimizing q

The EGPR regularizer $\mathcal{R}'_{\text{EGPR}}(\theta, q)$ is convex in q; therefore, we could compute q using any convex optimization algorithm. However, general-purpose convex optimization algorithms do not scale to problems with millions or billions of variables

such as those present in genomics. Therefore, we instead propose a novel alternating optimization strategy for performing this optimization more efficiently.

To enable closed form updates for this objective, we reformulate $J'_{\text{EGPR}}(\theta, q)$ by introducing a new variable $r^M(X_H)$. Like q, r^M is a distribution over X_H, but we require that r^M be factorizable as a product of marginals—that is $r^M(x_H) = \prod_h r_h^M(x_h)$. (In this manuscript, we use the notation $p^M(X_H)$ to indicate that p is a product of marginals.) We define the graph regularizer over r^M and add an additional term $\lambda_{\text{R1}} D(q(X_H) \| r^M(X_H))$, which encourages q and r^M to be similar. As we show below, restricting r^M in this way means that the reformulated objective is a lower bound on the original rather than being equivalent. We maximize this lower bound as an approximation to maximizing the original. The reformulated regularizer is

$$\begin{aligned} \mathcal{PR}'_{\text{EGPR-R1}}(q, r^M) \triangleq & -\lambda_{\text{R1}} D(q(X_H) \| r^M(X_H)) \\ & + f_{\text{R1}}(r^M) \end{aligned} \tag{6}$$

and

$$\begin{aligned} & f_{\text{R1}}(r^M) \triangleq \\ & -\lambda_G \sum_{(u,v) \in F_{EGPR}} w(u,v) D(r^M(X_u) \| r^M(X_v)), \end{aligned} \tag{7}$$

where $J'_{\text{EGPR-R1}}(\theta, q, r^M)$ and $\mathcal{R}'_{\text{EGPR-R1}}(\theta, q, r^M)$ are defined according to Equations (2) and (4) respectively using the corresponding regularizers. That is,

$$\begin{aligned} & \text{maximize}_{\theta, q, r^M} \quad J'_{\text{EGPR-R1}}(\theta, q, r^M) \triangleq \\ & \qquad \mathcal{L}(\theta) + \mathcal{R}'_{\text{EGPR-R1}}(\theta, q, r^M), \end{aligned} \tag{8}$$

$$\begin{aligned} & \mathcal{R}'_{\text{EGPR-R1}}(\theta, q, r^M) \triangleq \\ & -D(q(X_H) \| p_\theta(X_H | \bar{x}_O)) + \mathcal{PR}_{\text{EGPR-R1}}(q, r^M). \end{aligned} \tag{9}$$

First, we show that $r^M \approx q$ for large values of λ_{R1}, so optimizing the reformulated regularizer is equivalent to optimizing a lower bound on the original.

Lemma 2.1. *For distributions $p \in \mathcal{P}$ and $q \in \mathcal{Q}$ where $\mathcal{P} \cap \mathcal{Q} \neq \emptyset$ and a continuous function $J(p, q)$, let $\tilde{J}(p, q; \lambda) = J(p, q) - \lambda D(p \| q)$, and $p^*_\lambda, q^*_\lambda \in \text{argmax}_{p \in \mathcal{P}, q \in \mathcal{Q}} \tilde{J}(p, q; \lambda)$. Then the following hold:*

$$\lim_{\lambda \to \infty} D(p^*_\lambda \| q^*_\lambda) = 0, \tag{10}$$

$$\lim_{\lambda \to \infty} \| p^*_\lambda - q^*_\lambda \|_\ell = 0 \tag{11}$$

for any ℓ, where $\| \cdot \|_\ell$ is the ℓ-norm, and

$$\lim_{\lambda \to \infty} \max_{p \in \mathcal{P}, q \in \mathcal{Q}} \tilde{J}(p, q; \lambda) \leq \max_{p \in \mathcal{P}} J(p, p). \tag{12}$$

Proof. Consider any $\epsilon > 0$ and any $p' \in \mathcal{P}, q' \in \mathcal{Q}$ such that $D(p' \| q') > \epsilon$. Let $\hat{p} \in \text{argmax}_{p \in \mathcal{P} \cap \mathcal{Q}} J(p, p)$ and consider any $\lambda' \geq (1/\epsilon)(J(p' \| q') - J(\hat{p} \| \hat{p}))$.

$$\begin{aligned} \tilde{J}(p', q'; \lambda') &= J(p', q') - \lambda D(p' \| q') && (13) \\ &< J(p', q') - \lambda \epsilon && (14) \\ &\leq J(p', q') - \cancel{\epsilon} \cancel{(1/\epsilon)} (J(p' \| q') - J(\hat{p} \| \hat{p})) && (15) \\ &= J(\hat{p}, \hat{p}) && (16) \end{aligned}$$

Therefore, $D(p^* \| q^*) \leq \epsilon$ when $\lambda \geq \lambda'$. This proves Proposition (10).

We have that

$$D(p \| q) \geq \frac{1}{2} \| p - q \|_1^2 \geq \frac{1}{2} \| p - q \|_\ell^2 \tag{17}$$

$$\tag{18}$$

for any ℓ-norm. The first inequality is Pinsker's inequality and the second follows from the relationship of ℓ-norms. Proposition (11) follows from this combined with Proposition (10).

Due to Proposition (11) and the continuity of $J(p, q)$,

$$\lim_{\lambda \to \infty} \max_{p \in \mathcal{P}, q \in \mathcal{Q}} \tilde{J}(p, q; \lambda) - \max_{p \in \mathcal{P} \cap \mathcal{Q}} J(p, p) = 0. \tag{19}$$

Proposition (12) follows from this and the fact that $\mathcal{P} \cap \mathcal{Q} \subseteq \mathcal{P}$. □

Therefore, for sufficiently large λ_{R1}, optimizing Equation (7) is equivalent to optimizing a lower bound on Equation (5). This form allows us to compute q efficiently, which is shown as follows.

Theorem 2.2. *Define* $q^*(X_H) \triangleq \operatorname{argmax}_q J'_{\text{EGPR-R1}}(\theta, q, r^M)$. *Then,*

$$q^*(x_H) = \frac{p_\theta(x_H, \bar{x}_O)^{1/(1+\lambda_{\text{R1}})} \prod_{h \in H} r_h^M(x_h)^{\lambda_{\text{R1}}/(1+\lambda_{\text{R1}})}}{\sum_{x'_H} p_\theta(x'_H, \bar{x}_O)^{1/(1+\lambda_{\text{R1}})} \prod_{h \in H} r_h^M(x'_h)^{\lambda_{\text{R1}}/(1+\lambda_{\text{R1}})}}. \tag{20}$$

Proof. For ease of notation, we group all terms that do not depend on q into one function $K_{2.2}(r)$. Since we must respect the sum-to-one property of q, we form the Lagrangian by adding the term $\lambda_{2.2}(1 - \sum_{x_H} q(x_H))$

$$L_{2.2}(q, \lambda_{2.2}) = -D(q(X_H) \| p_\theta(X_H | X_O)) - \lambda_{\text{R1}} D(q(X_H) \| r^M(X_H)) - \lambda_{2.2}(1 - \sum_{x_H} q(x_H)) + K_{2.2}(r) \tag{21}$$

$$= \sum_{x_H} q(x_H) \log \frac{p_\theta(x_H | \bar{x}_O) r^M(x_H)^{\lambda_{\text{R1}}}}{q(x_H)^{1+\lambda_{\text{R1}}}} - \lambda_{2.2}(1 - \sum_{x_H} q(x_H)) + K_{2.2}(r) \tag{22}$$

$$= \sum_{x_H} q(x_H) \log \frac{p_\theta(x_H, \bar{x}_O) r^M(x_H)^{\lambda_{\text{R1}}}}{q(x_H)^{1+\lambda_{\text{R1}}}} - \log p_\theta(\bar{x}_O) - \lambda_{2.2}(1 - \sum_{x_H} q(x_H)) + K_{2.2}(r) \tag{23}$$

$$0 = \frac{\partial L}{\partial q(x_H)} = -\log p_\theta(x_H, \bar{x}_O) r^M(x_H)^{\lambda_{\text{R1}}} + \log q(x_H)^{1+\lambda_{\text{R1}}} + 1 + \lambda_{\text{R1}} - \lambda_{2.2} \tag{24}$$

$$\implies q(x_H) \propto p_\theta(x_H, \bar{x}_O)^{\frac{1}{1+\lambda_{\text{R1}}}} r^M(x_H)^{\frac{\lambda_{\text{R1}}}{1+\lambda_{\text{R1}}}} \tag{25}$$

□

This is identical to the original model p_θ, but with one additional factor $r_h^M(x_h)^{\lambda_{\text{R1}}/(1+\lambda_{\text{R1}})}$ over each label. Critically, because r^M is factorizable such that each factor involves just one variable X_h, $q^*(X_H)$ is factorizable in the same way as the unregularized model $p_\theta(X_H, \bar{x}_O)$. For example, if the original model was an HMM, q still factors as a chain. Therefore, the normalization constant can be computed using any algorithm for exact or approximate probabilistic inference on factorized models, such as belief propagation, with similar computational cost as the unregularized model.

2.2 Optimizing r^M

While the last reformulation enabled closed form updates for q, the objective still does not admit closed-form updates for r^M. This is due to the fact that an objective of the form $D(p\|q) + D(p\|r)$ admits closed form updates for p, while $D(p\|q) + D(r\|p)$ does not. Therefore, we again reformulate $\mathcal{PR}_{\text{EGPR-R1}}(\theta, q, r^M)$ by adding a new variable s^M, where s^M is also a distribution over X_H restricted to be factorizable as a product of marginals. As before, we add a term $\lambda_{\text{R2}} D(s^M(X_H) \| r^M(X_H))$, which encourages $s^M \approx r^M$. We define the graph regularizer KL divergence terms to have s^M on the left and r^M on the right—that is, in the form $D(s_u^M(X_u) \| r_v^M(X_v))$—which will enable efficient optimization for both variables.

$$\begin{aligned} \mathcal{PR}'_{\text{EGPR-R2}}(q, r^M, s^M) &\triangleq -\lambda_{\text{R1}} D(q(X_H) \| r^M(X_H)) \\ &\quad + \max_{s^M} f_{\text{R2}}(r^M, s^M) \\ f_{\text{R2}}(r^M, s^M) &\triangleq -\lambda_{\text{R2}} D(s^M(X_H) \| r^M(X_H)) \\ &\quad - \lambda_G \sum_{(u,v) \in F_{EGPR}} w(u, v) D(s_u^M(X_u) \| r_v^M(X_v)). \end{aligned} \tag{26}$$

$J'_{\text{EGPR-R2}}(\theta, q, r^M, s^M)$ and $\mathcal{R}'_{\text{EGPR-R2}}(\theta, q, r^M, s^M)$ are defined according to Equations (2) and (4) respectively using the corresponding regularizers. That is,

$$\begin{aligned} &\text{maximize}_{\theta,q,r^M,s^M} \quad J'_{\text{EGPR-R2}}(\theta, q, r^M, s^M) \triangleq \\ &\qquad \mathcal{L}(\theta) + \mathcal{R}'_{\text{EGPR-R2}}(\theta, q, r^M, s^M), \end{aligned} \tag{27}$$

$$\begin{aligned} &\mathcal{R}'_{\text{EGPR-R1}}(\theta, q, r^M, s^M) \triangleq \\ &\qquad - D(q(X_H) \| p_\theta(X_H | \bar{x}_O)) + \mathcal{PR}_{\text{EGPR-R2}}(q, r^M, s^M). \end{aligned} \tag{28}$$

By Lemma 2.1, optimizing $\mathcal{R}_{\text{EGPR-R2}}(q)$ is equivalent to optimizing $\mathcal{R}_{\text{EGPR-R1}}(q)$ for large values of λ_{R2}. This regularizer can be optimized in r^M and s^M using closed-form updates, shown as follows.

Theorem 2.3. *For notational simplicity, define a new regularization graph with self-edges of weight $\lambda_{\text{R2}}/\lambda_G$, $E'_{\text{EGPR}} \triangleq E_{\text{R}} \cup \{(h,h) \mid h \in H\}$, and $w'(u,v) \triangleq w(u,v) + \delta(u=v)\lambda_{\text{R2}}/\lambda_G$. Let $r^{M^*}(X_H) \in \text{argmax}_{r^M} J'_{\text{EGPR-R2}}(\theta, q, r^M, s^M)$ and $s^{M^*}(X_H) \in \text{argmax}_{s^M} J'_{\text{EGPR-R2}}(\theta, q, r^M, s^M)$. Then,*

$$r_v^M(x_v) = \frac{\lambda_{\text{R1}} q_v^M(x_v) + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) s_u^M(x_v)}{\lambda_{\text{R1}} + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v)}, \tag{29}$$

$$s_u^{M^*}(x_u) = \frac{\exp \frac{\sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) \log r_v^M(x_u)}{\sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v)}}{\sum_{x'_u} \exp \frac{\sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) \log r_v^M(x'_u)}{\sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v)}}. \tag{30}$$

Proof. In its current form, $\mathcal{PR}_{\text{EGPR-R2}}(q)$ involves a sum over all values of $q(X_H)$. However, the following lemma shows how the factorizability of r^M facilitates expressing the objective in a form that involves only sum over values of each variable X_h.

Lemma 2.4. *For distribution $p(X_V)$ and factorizable distribution $q^M(X_V) = \prod_{v\in V} q_v^M(X_v)$, define $p_v^M(X_v) \triangleq \sum_{X_{V\setminus v}} p(X_V)$.*

$$D(p\|q^M) = \sum_{v\in V} D(p(X_v)\|q(X_v)) - H(p) + \sum_{v\in V} H(q(X_v)). \tag{31}$$

Proof.

$$\begin{aligned} D(p\|q^M) &= -H(p) - \sum_{x_V} p(x_V) \log \left(\prod_{v\in V} q_v^M(X_v) \right) && (32)\\ &= -H(p) - \sum_{x_V} p(x_V) \sum_{v\in V} \log q_v^M(X_v) && (33)\\ &= -H(p) - \sum_{v\in V} \sum_{x_v} \sum_{x_V \neq v} p(x_V) \log q_v^M(X_v) && (34)\\ &= -H(p) - \sum_{v\in V} \sum_{x_v} \left(\log q_v^M(X_v)\right) \sum_{x_V \neq v} p(x_V) && (35)\\ &= -H(p) - \sum_{v\in V} \sum_{x_v} \left(\log q_v^M(X_v)\right) p_v^M(x_v) && (36)\\ &= \sum_{v\in V} D(p(X_v)\|q(X_v)) - H(p) + \sum_{v\in V} H(q(X_v)) && (37) \end{aligned}$$

□

Define q_h^M to be the marginal distribution of q over X_h, $q_h^M(X_h) \triangleq \sum_{X_{H\setminus h}} q(X_H)$. Using Lemma 2.4,

$$\mathcal{PR}_{\text{EGPR-R2}}(q) = \max_{r^M} \left(-\lambda_{\text{R1}} \sum_{h\in H} D(q_h^M(X_h) \| r_h^M(X_h)) + H(q) - \sum_{h\in H} H(q_h^M(X_h)) + f_{\text{R2}}(r^M) \right). \tag{38}$$

We now proceed to derive the update steps. We first derive the update for r^M. The Lagrangian for the optimization of r_v^M is

$$\begin{aligned}
&L_{2.3\text{–}1}(r_v^M, \lambda_{2.3\text{–}1}) && (39)\\
&= \lambda_{\text{R1}} D(q_v^M(X_v) \| r_v^M(X_v)) + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) D(s_u^M(X_u) \| r_v^M(X_u)) && (40)\\
&\quad + \lambda_{2.3\text{–}1}(1 - \sum_{x_v} r_v^M(X_v)) + K_{2.3\text{–}1}(q, s^M, r_{H\setminus v}^M) && (41)\\
0 = \frac{\partial L}{\partial r_v^M(x_v)} &= -\left(\lambda_{\text{R1}} q_v^M(x_v) + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) s_u^M(x_v) \right) \frac{1}{r_v^M(x_v)} + \lambda_{2.3\text{–}1} && (42)\\
\implies r_v^M(x_v) &\propto \lambda_{\text{R1}} q_v^M(x_v) + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) s_u^M(x_v) && (43)\\
&\sum_{x_v} \left(\lambda_{\text{R1}} q_v^M(x_v) + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) s_u^M(x_v) \right) && (44)\\
&= \lambda_{\text{R1}} \sum_{x_v} q_v^M(x_v) \overset{1}{} + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) \sum_{x_v} s_u^M(x_v) \overset{1}{} && (45)\\
\implies r_v^M(x_v) &= \frac{\lambda_{\text{R1}} q_v^M(x_v) + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) s_u^M(x_v)}{\lambda_{\text{R1}} + \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v)} && (46)
\end{aligned}$$

We next derive the update for s^M. The Lagrangian for the optimization of s_u^M is

$$\begin{aligned}
&L_{2.3\text{–}2}(s_u^M, \lambda_{2.3\text{–}2}) && (47)\\
&= \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) D(s_u^M(X_u) \| r_v^M(X_u)) + \lambda_{2.3\text{–}2}(1 - \sum_{x_u} s_u^M(X_u)) + K_{2.3\text{–}2}(q, r^M, s_{H\setminus u}^M) && (48)\\
&= \lambda_G \sum_{x_u} s_u^M(x_u) \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) \log \frac{s_u^M(x_u)}{r_v^M(x_u)} + \lambda_{2.3\text{–}2}(1 - \sum_{x_u} s_u^M(X_u)) + K_{2.3\text{–}2}(q, r^M, s_{H\setminus u}^M) && (49)\\
0 = \frac{\partial L}{\partial s_u^M(x_u)} &= \lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) \log \frac{1}{r_v^M(x_u)} + \left(\lambda_G \sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) \right) (1 + \log s_u^M(x_u)) - \lambda_{2.3\text{–}2} && (50)\\
\implies s_u^M(x_u) &\propto \exp \frac{\sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v) \log r_v^M(x_u)}{\sum_{(u,v)\in E'_{\text{EGPR}}} w'(u,v)} && (51)
\end{aligned}$$

□

2.3 Updating θ

The preceding section described an algorithm for computing $\operatorname{argmax}_q \mathcal{R}_{\text{EGPR-R2}}$. This algorithm can be combined with an EM-like algorithm in order to learn a θ that (locally) optimizes $J_{\text{EGPR-R2}}$, as we describe in this section. We use an alternating EM-like algorithm to compute θ.

E-step:
$q^{(t+1)} \in \operatorname{argmax}_{q, r^M, s^M} J'_{\text{EGPR-R2}}(\theta^{(t)}, q, r^M, s^M)$

M-step: $\theta^{(t+1)} \in \operatorname{argmax}_\theta J'_{\text{EGPR}}(\theta, q^{(t+1)})$

The preceding section showed how to perform the E-step. To compute the M-step,

$$\begin{aligned} &\operatorname{argmax}_\theta J'_{\text{EGPR}}(\theta, q^{(t+1)}) \\ &\quad = \operatorname{argmax}_\theta E_{q^{(t+1)}(X_H)}\left[\log p_\theta(X_H, \bar{x}_O))\right] \end{aligned} \tag{52}$$

The M-step takes the same form as the EM algorithm presented in (Neal & Hinton, 1999). The update for θ depends on the particular factorization and parameterization properties of the model. Because the posterior distribution $q(X_H)$ obeys the same factorization properties as the unregularized model $p_\theta(X_H, X_O)$, the same closed-form updates for θ can be used.

Therefore, the upper bound on the EGPR objective, $J_{\text{EGPR-R2}}$, can be minimized using a three-way alternating optimization algorithm, which proceeds by alternating closed-form updates to r^M and r^M to convergence, alternating this whole update of r^M/s^M with closed-form updates to q until convergence, then finally alternating updates to q and θ until convergence. A schematic of the algorithm is shown in Figure 1. The full algorithm in pseudocode is shown in Algorithm 1.

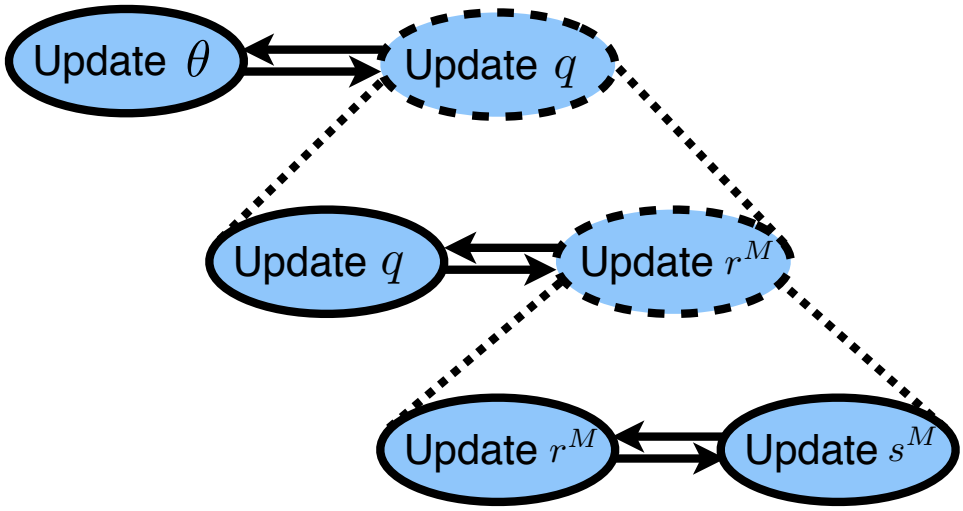


Figure 1: Illustration of optimization algorithm. Solid ovals denote closed-form update steps. Dashed ovals with dotted expansion lines denote updates that are implemented by alternating optimization. Pairs of opposing arrows indicate alternating optimization implemented by iterating each update to convergence. See the extended version (Libbrecht et al., 2015) for the full algorithm.

Theorem 2.5. *The modified EM algorithm monotonically increases the relaxed EGPR objective:*

$$J_{\text{EGPR-R2}}(\theta^{(t)}) \leq J_{\text{EGPR-R2}}(\theta^{(t+1)}). \tag{53}$$

Proof. Function $q^*(\cdot)$ of Algorithm 1 implements coordinate descent on q, r^M and s^M. $D(p\|q)$ and $D(p\|p)$ are jointly strictly convex in p and q and bounded below by 0. Thus, $J'_{\text{EGPR-R2}}$ is bounded below and jointly strictly convex in q, r^M and s^M. Convergence to the global optimum of $J'_{\text{EGPR-R2}}$ in q, r^M and s^M follows from its strict convexity (**?**).

$$\begin{aligned} J_{\text{EGPR-R2}}(\theta^{(t)}) &= J'_{\text{EGPR-R2}}(\theta^{(t)}, q^{(t+1)}, r^{M\,(t+1)}, s^{M\,(t+1)}) \\ &\leq J'_{\text{EGPR-R2}}(\theta^{(t+1)}, q^{(t+1)}, r^{M\,(t+1)}, s^{M\,(t+1)}) \\ &\leq J_{\text{EGPR-R2}}(\theta^{(t+1)}) \end{aligned}$$

The first equality follows from the global optimality of $q^{(t+1)}$, $r^{M\,(t+1)}$ and $s^{M\,(t+1)}$. The second inequality follows from the fact that $\theta^{(t+1)}$ is chosen to maximize $J'_{\text{EGPR-R2}}$. The third inequality follows from the fact that $J'_{\text{EGPR-R2}}(\theta, q, r^M, s^M)$ is a lower bound on $J_{\text{EGPR-R2}}(\theta)$. □

Algorithm 1 Efficient and scalable algorithm to optimize $J'_{\text{EGPR-R2}}$

function $r^{M*}(q)$
  for $h \in H$ **do**
    $q_h^M(x_h) \leftarrow \sum_{x_{H \neq h}} q(x_H)$ (belief propagation)
  end for
  Initialize $r^{M(0)}$, $s^{M(0)}$ arbitrarily.
  $t_1 \leftarrow 1$
  while not converged **do**
    for $v \in H$ **do**
      $r_v^{M(t_1)}(x_v) \leftarrow \frac{\lambda_{\text{R1}} q_v^M(x_v) + \lambda_G \sum_{(u,v) \in E'_{\text{EGPR}}} w'(u,v) s_u^{M(t_1-1)}(x_v)}{\lambda_{\text{R1}} + \lambda_G \sum_{(u,v) \in E'_{\text{EGPR}}} w'(u,v)}$
    end for
    for $u \in H$ **do**
      $s_u^{M(t_1)}(x_u) \leftarrow \frac{\exp \frac{\sum_{(u,v) \in E'_{\text{EGPR}}} w'(u,v) \log r_v^{M(t_1-1)}(x_u)}{\sum_{(u,v) \in E'_{\text{EGPR}}} w'(u,v)}}{\sum_{x'_u} \exp \frac{\sum_{(u,v) \in E'_{\text{EGPR}}} w'(u,v) \log r_v^{M(t_1-1)}(x'_u)}{\sum_{(u,v) \in E'_{\text{EGPR}}} w'(u,v)}}$
    end for
    $t_1 \leftarrow t_1 + 1$
  end while
  return $r^{M(t_1)}$
end function

function $q^*(\theta)$
  $t_2 \leftarrow 1$
  Initialize $r^{M(0)}$ arbitrarily.
  while not converged **do**
    $q^{(t_2)}(x_H) \leftarrow \frac{p_\theta(x_H, \bar{x}_O)^{1/(1+\lambda_{\text{R1}})} \prod_{h \in H} r_h^{M(t_2-1)}(x_h)^{\lambda_{\text{R1}}/(1+\lambda_{\text{R1}})}}{\sum_{x'_H} p_\theta(x'_H, \bar{x}_O)^{1/(1+\lambda_{\text{R1}})} \prod_{h \in H} r_h^{M(t_2-1)}(x'_h)^{\lambda_{\text{R1}}/(1+\lambda_{\text{R1}})}}$ (belief propagation)
    $r^{M(t_2)} \leftarrow r^{M*}(q^{(t_2)})$
    $t_2 \leftarrow t_2 + 1$
  end while
end function

Initialize $\theta^{(0)}$ arbitrarily.
$t_3 \leftarrow 1$
while not converged **do**
  $q^{(t_3)} \leftarrow q^*(\theta^{(t_3-1)})$
  $\theta^{(t_3)} \leftarrow \text{argmax}_\theta \, E_{q^{(t_3)}(X_H)} [\log p_\theta(X_H, \bar{x}_O))]$ (EM update)
end while
Output $\theta^{(t_3)}$

3 EGPR for Inference

In addition to its use for regularized learning, EGPR can be used directly as an inference algorithm. To do this, we compute $q^* \leftarrow \text{argmax}_q \, J'_{\text{EGPR-R2}}(\theta, q)$ and use this distribution as the posterior. As discussed above, q^* obeys the same factorization properties as p_θ, so algorithms such as Viterbi and belief propagation can be used to compute the MAP solution and marginal distributions of q respectively. Using EGPR as an inference algorithm results in posteriors which are smooth with respect to the graph G_R.

4 Related work

The most straightforward way to express similarity information in an unsupervised model is to encode it in the graphical model. For example, one might add a factor between X_u and X_v as in $\phi(x_u, x_v) = \lambda \mathbf{1}(x_u = x_v)$. This form of interaction is quite different from EGPR, because adding factors changes the "implementation" of the model, while EGPR regularizes what the model *does*. Moreover, there are two problems with this approach. First, it is not always clear what form of interaction is

most appropriate. In particular, although KL-based penalties have been very successful in graph-based semi-supervised learning, they cannot be converted to an equivalent set of probability factors. Second, adding similarity edges to the probabilistic model results in a model that does not, in general, have low tree-width, so efficient exact inference algorithms such as belief propagation cannot be used, and one must resort to approximate inference. As we show in Section 5.2, EGPR performs better than the loopy belief propagation (LBP) approximate inference algorithm on an augmented graph.

Three methods take a similar approach to ours, by augmenting a probabilistic model with a graph regularizer. First, Altun et al. (2005) describe a graph regularization for max-margin models applied to pitch-accent prediction and optical character recognition. However, this method involves a matrix inversion step, and thus it cannot scale to large models. Second, Subramanya et al. (2010) combine a temporal conditional random field with a regularizer that expresses pairwise squared-error penalties derived from unlabeled data. They apply this method to the part-of-speech tagging task (Subramanya et al., 2010) and later to related problems in natural language (Das & Petrov, 2011; Das & Smith, 2011). That work, however, resorts to a purely heuristic update step and lacks any optimality guarantees.

Third, He et al. (2013) present an approach based on an exponentiated gradient descent algorithm. Like our approach, He's approach exhibits monotone convergence. Although He's work has many similarities with our approach, He's work differs from ours in three important ways. First, He's method uses a squared-error penalty, which, as argued above, is less appropriate for probability distributions than Kullback-Leibler divergence (Bishop, 1995, p. 226). As shown in Section 5.2, using squared error also results in worse performance in practice. Second, the exponentiated gradient descent method is applied to semi-supervised handwriting recognition and part-of-speech tagging, while we apply EGPR to an unsupervised genome annotation problem. Third, He et al. (He et al., 2013) use an exponentiated gradient descent strategy, while we use alternating optimization.

Our alternating optimization approach has several benefits over the exponentiated gradient descent method of He et al. (He et al., 2013). First, the He et al. algorithm involves gradient calculations over each clique in the conditional dependence graph, with order $O(k^{|C|})$ for a variable of dimension k involved in cliques of size $|C|$. Our alternating minimization algorithm, by contrast, has closed-form updates of order $O(k)$ for q, r^M and s^M. (Updates for θ can still involve $O(k^{|C|})$ calculations, but these are generally very fast). Therefore, the alternating optimization algorithm is more appropriate for extremely large models or models with large cliques or high-order factors. Second, empirical studies of KL-based graph smoothness objectives have shown that alternating optimization algorithms perform better than gradient-based methods for these objectives (Subramanya & Bilmes, 2011; **?**). Finally, the alternating optimization strategy is parallelizable (**?**) and extremely simple to implement because the graph regularization step has simple, closed-form updates with no learning rate hyperparameters, and the posterior calculation can be performed using any probabilistic inference method on a model with the same conditional dependence graph as the unregularized model.

5 Simulations

All implementations in the below use the GMTK, the graphical models toolkit (**?**), and its use of virtual evidence factors, for HMM and dynamic graphical model inference

5.1 Learning a Gaussian Mixture on Poorly-Separated Data

First, we consider a simple example that demonstrates the utility of EGPR (Figure 2). Suppose we wish to learn a mixture of four Gaussians on data lying in a circle in 2D, using EM training to find cluster centers. Clearly, the training problem is underspecified in this case, because any set of centers at 90 degrees from one another relative to the circle's center represents an optimum of the model likelihood. However, suppose we additionally have pairwise information that certain slices of the circle should form clusters. To represent this additional information, we form a regularization graph that connects all pairs of positions within the same cluster and run EM with EGPR using this graph. EGPR finds the true cluster centers and recovers the true labels with 100% accuracy compared to 74% accuracy with EM alone. This example demonstrates how pairwise information can be integrated in the training process to produce a trained model that implicitly incorporates this information.

5.2 Comparison with Related Inference Methods

To evaluate the efficacy of EGPR, we compared EGPR to two related methods: 1) approximate inference on a graphical model with the same dependence structure, and 2) GPR using squared-error penalties. We compared to the approximate inference method loopy belief propagation (LBP) because it is one of the most widely used approximate inference methods. While we would have preferred to perform this comparison using our genomics data sets (see Section 6), due to the large size of the models and dense connectivity of the regularization graphs, it appeared that even our implementations of these methods

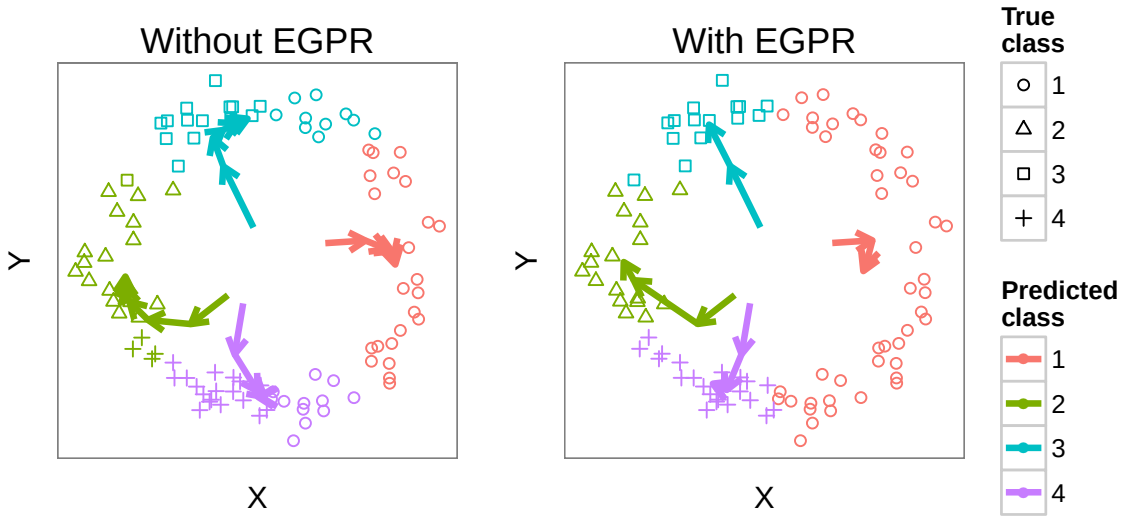


Figure 2: Using EGPR to learn ambiguous clusters. Shape denotes true class, color denotes predicted class, and colored arrows denote cluster means as they evolve between iterations of EM.

would take months to converge. Therefore, we instead performed this comparison using synthetic data. We generated a chain of length $n = 200$, with $(X_H, X_O) = (Z_{1:200}, Y_{1:200})$, where $Z_{1:200} \in \{0,1\}^n$ and $Y_{1:200} \in \mathbb{R}^n$. We defined an HMM over this chain with transition probabilities $\Pr(Z_i = Z_{i+1}) = 0.9$ and emission probabilities $Y_i \sim N(Z_i, \sigma)$, where we vary σ to control the difficulty of the problem—higher σ results in more challenging inference. We generated a graph $W \in \mathbb{R}^{n \times n}$ over the vertices of the chain by setting $w_{ij} = 1$ with probability 0.4 if $Z_i = Z_j$, $w_{ij} = 1$ with probability 0.1 if $Z_i \neq Z_j$, and $w_{ij} = 0$ otherwise. This model is meant to simulate the task of labeling a chain (such as a genomic sequence) where we have noisy information about which pairs of positions have the same label.

We compared five methods of inference: 1) inference on each position independently, with no chain model; 2) inference on the chain alone, without using W; 3) LBP on the chain plus extra factors of $\Pr(X_i = X_j) = \text{sigmoid}(\lambda w_{ij})$, where λ controls the strength of these factors; 4) GPR using the regularization graph W and a squared-error penalties as described in (He et al., 2013) (SQGPR); and 5) EGPR using the regularization graph W. We chose hyperparameters for each model (λ_G, λ_{R1} and λ_{R2} for GPR and λ for LBP) using a training set of 200 simulations. We evaluated results according to the average accuracy over 200 simulations of MAP inference on the model in question.

EGPR significantly outperforms all other models for all experiments (Figure 3, providing nearly as much improvement in accuracy as does the chain model itself. The pattern of accuracy is instructive in understanding the properties of each model. LBP performs very well when there is little noise, but becomes easily stuck in local optima on harder problems. GPR with squared error provides a modest improvement over the chain model, but has poor performance relative to KL penalties, consistent with previous work on semi-supervised methods (Subramanya & Bilmes, 2011; **?**).

6 Application: Genome Annotation Using Physical Interaction Information

Recently, many methods have been described that partition and label the human genome on the basis of a number of genome-wide real-valued signal tracks, generally employing temporal models such as HMMs (Day et al., 2007; Hoffman et al., 2012a; Ernst & Kellis, 2010; Filion et al., 2010; Thurman et al., 2007; Lian et al., 2008). Formally, these methods aim to learn a labeling $X_{H\,1:n} \in \{1..L\}^n$ which associates each position in the genome with one of L integer labels, such that positions that receive the same label exhibit similar patterns in the signal data. The input is comprised of a feature vector $X_{O\,i} \in \mathbb{R}^F$ at each position that represents the output of biological experiments that measure local properties of the DNA, including its interaction with binding proteins, its local structure, and various types of chemical modifications. The process is "semi-automated" because a human assigns a semantic interpretation of the integer labels subsequent to the unsupervised learning phase.

However, existing genome annotation methods cannot incorporate the genome's 3D conformation. The 3D arrangement of the genome in the nucleus plays a central role in gene regulation, chromatin state and replication timing (Misteli, 2007; Dekker et al., 2002; Ryba et al., 2010; Dixon et al., 2012). Genome conformation can be investigated using chromatin conformation capture experiments such as Hi-C (Lieberman-Aiden et al., 2009). A Hi-C experiment outputs a matrix of contact counts, where the number of contact counts of a pair of genomic positions is inversely proportional to the positions'

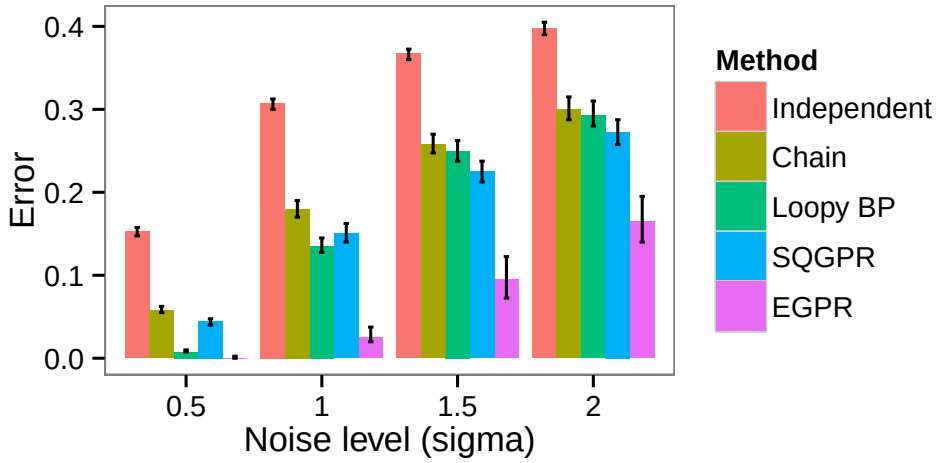
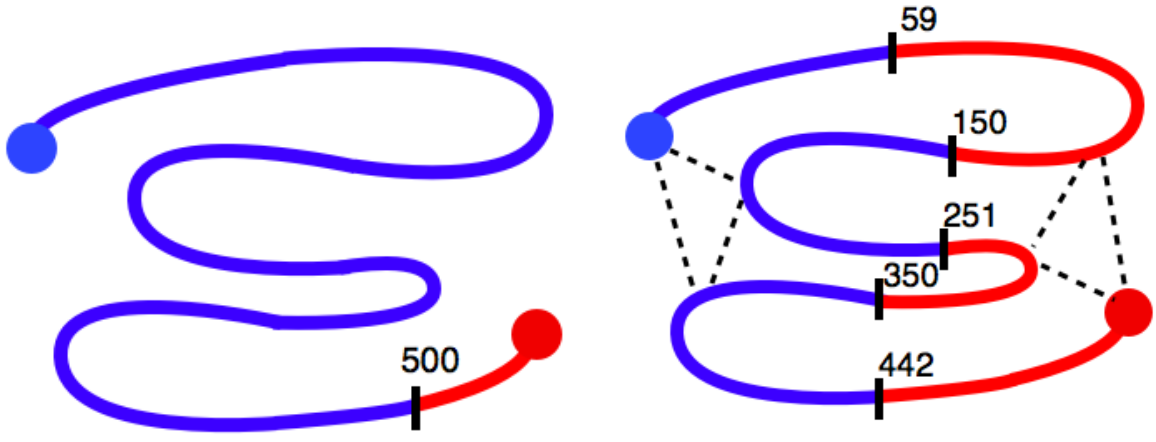


Figure 3: Comparison of EGPR with related inference methods. The X axis shows σ, a hyperparameter controlling the difficulty of inference. The Y axis shows the average accuracy over 200 simulations of MAP inference on the model in question (95% Wilcoxon test confidence intervals).

3D distance in the nucleus (Lieberman-Aiden et al., 2009; Ay et al., 2014b). Existing genome annotation methods can incorporate any data set that can be represented as a vector defined linearly across the genome, but they cannot incorporate inherently pairwise Hi-C data without resorting to simplifying transformations such as principle component analysis.

We therefore present a novel strategy for integrating 3D conformation information using EGPR in which we encourage pairs of positions which interact in 3D to receive the same label in the annotation by connecting these positions with edges in an EGPR graph (Figure 5(a)). While the assumption that positions close in 3D space have similar regulatory state is not necessarily true at a small scale ($\sim$1 thousand base pair elements), it does generally hold at a large scale ($\sim$1 million base pair elements) (Lieberman-Aiden et al., 2009).

6.1 Synthetic Example



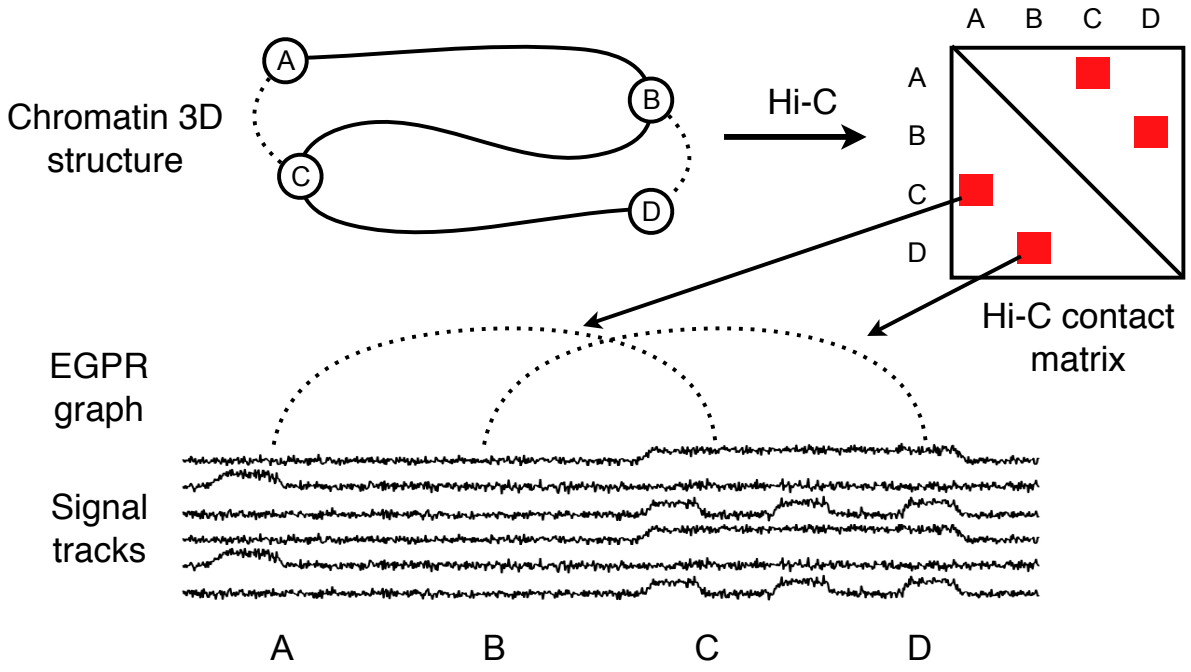
(a) Without EGPR (b) With EGPR

Figure 4: Synthetic model of genome spatial interactions. Color and labeled division lines indicate learned labels along the hypothesized 501 bp genome. Large filled circles indicate observed positions. Dotted lines indicate EGPR edges.

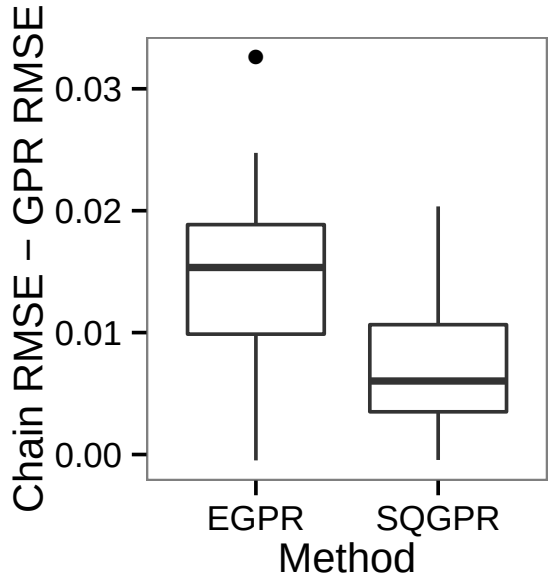
To motivate this approach, we first consider a simple synthetic model with 501 nodes and six EGPR edges. Let $Y_i \in \{0, 1\}$ for $i = 0 \ldots 500$. We assign $Y_1 = 0$ and $Y_{500} = 1$. Let $P(Y_{0:500}) = \prod_{i=0}^{499} 0.9^{\mathbf{1}(Y_i = Y_{i+1})} 0.1^{\mathbf{1}(Y_i \neq Y_{i+1})}$. In other words, the model places higher probability on neighboring positions taking the same label but provides no other

information. We construct an EGPR graph with $w_{0,200} = w_{0,400} = w_{200,400} = w_{100,300} = w_{100,500} = w_{300,500} = 1$, corresponding to a hypothetical 2D arrangement of the chain. Without EGPR, the model learns a trivial labeling of the chain, whereas with EGPR, the model learns a labeling corresponding to the 2D arrangement (Figure 4).

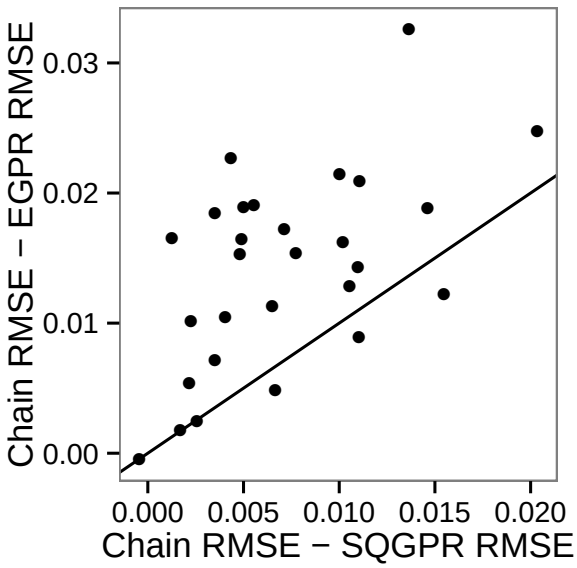
6.2 Real data



(a)



(b)



(c)

Figure 5: (a) Strategy for utilizing physical interaction information. (b) Improvement in RMSE over chain model for EGPR and SQGPR for 29 experiments. (c) Relative improvement in RMSE between EGPR and SQGPR for each of 29 experiments.

Next, to evaluate the efficacy of our approach, we performed genome annotation of the human fibroblast cell line IMR90.

We compared three models: (1) the chain model described in (Hoffman et al., 2012a), without 3D structure data, (2) the chain model augmented with 3D structure data expressed with squared-error GPR (SQGPR), and (3) the chain model augmented with 3D structure data expressed with EGPR. We would have liked to compare against loopy belief propagation as well, but as mentioned previously, it appeared that our fastest implementation of this method would take months to converge on this large data set. In order to evaluate our performance in a variety of conditions, we ran each model separately once for each of the 29 available data sets in IMR90.

For SQGPR and EGPR, we used a GBR graph based on 3D structure data. To generate the GBR graph representation of 3D structure used by EGPR and SQGPR, we used the Hi-C data set of (Dixon et al., 2012) and processed the Hi-C data into a matrix of pairwise p-values using the Fit-Hi-C method (Ay et al., 2014a). To remove noise and decrease the degree of the graph, we removed all contacts with uncorrected p-value $p > 10^{-6}$ and multiplied the remaining p-values by 10^6, similar to a Bonferroni correction. We generated the GPR graph by setting the weight between positions i and j, $w(i,j)$, to $w(i,j) \triangleq \max(0, -\log_e(p(i,j)/10^6))$, where $p(i,j)$ is the p-value of interaction between positions i and j. All annotations used four labels and binned the genome at 10,000 base pair resolution, comparable to the resolution of 40,000 bp used in (Dixon et al., 2012). We chose the best-performing hyperparameters for each GPR model using a validation set.

To evaluate these annotations, we used the time during the cell cycle at which the DNA is replicated as a gold standard (Woodfine et al., 2004). Replication time is highly correlated with gene expression and chromatin state and therefore is a good proxy for domain type (Lieberman-Aiden et al., 2009). To evaluate the accuracy with which an annotation predicts replication time, we compute a prediction as follows. Let $a_i \in \{1 \dots 4\}$ and x_i be the annotation label and replication time at position i, respectively. We compute the replication time mean over the positions assigned a given label ℓ as

$$\mu_\ell \triangleq \frac{\sum_{i=1}^n \mathbf{1}(a_i = \ell) x_i}{\sum_{i=1}^n \mathbf{1}(a_i = \ell)} \quad \text{for } \ell \in \{1 \dots 4\}. \tag{54}$$

We define a predicted replication time vector $x_i^p = \mu_{a_i}$ and compute the root mean squared error (RMSE) of this prediction as RMSE $= \sqrt{\sum_i (x_i - x_i^p)^2}$. EGPR consistently outperforms both the chain model alone and SQGPR (Figure 5).

7 Discussion

We have defined entropic graph-based posterior regularization (EGPR), a method to encourage a model's posteriors to be smooth according to a regularization graph. This method is motivated by graph-based methods for semi-supervised learning, which have had great success in that setting but have not been studied thoroughly in an unsupervised setting. We showed that EGPR greatly outperforms both the approximate inference method loopy belief propagation and previous methods for graph-based regularization. We used EGPR to incorporate 3D structure data for semi-automated genome annotation and showed that EGPR greatly improved the quality of the resulting annotations. This method will thereby enable these methods to distill complicated pairwise contact matrices into human-interpretable genome annotations. Moreover, because EGPR can use any graphical model and similarity graph, it will likely have diverse applications in other fields such as natural language and time-series analysis.

8 Frequently Asked Questions

This section collects questions we have received when presenting this work.

1. **Why is KL divergence better than squared error for probability distributions?** Squared error is based on a Gaussian error model, which is not appropriate for probability values, and it under-penalizes differences between small probability values. p-norms are defined over all real numbers, while posteriors must lie within the range $[0, 1]$ (and live in a simplex). A more justified way to measure divergence between probability distributions is to employ the KL divergence. The KL divergence measures the difference of exponents in the probability and so evaluates differences between small and large probabilities more uniformly. Also, Pinsker's inequality (Csiszár & Tusnády, 1984) combined with the relationship of ℓ-norms implies that $D(p\|q) \geq \frac{1}{2}\|p - q\|_1^2 \geq \frac{1}{2}\|p - q\|_\ell^2$, for all $\ell \geq 1$, where $\|\cdot\|_\ell$ is the ℓ-norm. Hence, minimizing KL divergence minimizes an upper bound on all ℓ-norms.

   As a concrete example, consider two pairs of probability distributions over two possible events: $[0.55, 0.45]$ vs. $[0.45, 0.55]$, and $[0.1, 0.9]$ vs. $[10^{-10}, 1 - 10^{-10}]$. The first pair of distributions are very similar, with both events being roughly equally likely. The second pair is quite dissimilar, with the first event being reasonably likely in the first case and astronomically unlikely in the second. Squared error actually regards the first pair as more dissimilar, while

KL divergence correctly identifies the second pair as much more dissimilar. Despite the advantages of KL divergence, although all posterior regularization objectives include a KL term binding q to be similar to p_θ, to our knowledge, no existing methods define the posterior regularizer **itself** using KL. This is also discussed in Section 1.

In practice, although a regularizer based on squared error shows good results for some problems, we demonstrate the better performance of KL divergence empirically in Sections 5 and 6.

2. **KL divergence is asymmetric in its two arguments. How does that affect the results?** As stated in Section 1, the KL divergence in Equation 5 is symmetrized because G_R is undirected—that is $w(u, v) = w(v, u)$—so $D(q_u \| q_v)$ and $D(q_v \| q_u)$ appear with the same weight in the regularizer. That is, $\sum_{u,v} w(u, v) D(q_u \| q_v) = \sum_{u \leq v} 2w(u, v)(D(q_u \| q_v) + D(q_v \| q_u))$.

3. **How does EGPR compare to other methods?** This topic is discussed in detail in Section 4.

   (a) **How does EGPR compare to approximate inference?** Posterior regularization is so named because it regularizes the model parameters indirectly via a penalty defined using the posterior distribution. It is therefore fundamentally different from approximate (or any) inference methods, which are used to perform probabilistic inference. Although complementary, inference methods are quite a separate machine-learning concept from parameter regularization strategies.

   In addition, we demonstrate empirically in Section 5 that EGPR outperforms the approximate inference algorithm loopy belief propagation (LBP) on a synthetic data set. (Note that LBP can be understood either in the context of belief propagation or as a variational method—see (**?**).)

   (b) **How does EGPR compare to the first posterior regularization method of (Ganchev et al., 2010)?** The method of (Ganchev et al., 2010) requires each posterior regularization term to involve variables in just one clique in the graphical model. Applying this method to a graph-based posterior regularization objective would result in a high tree-width model and intractable inference.

   (c) **How does EGPR compare to the graph-based posterior regularization method of (He et al., 2013)?** He et al. (2013) present an approach based on an exponentiated gradient descent algorithm. Like our approach, He's approach exhibits monotone convergence. Although He's work has many similarities with our approach, He's work differs from ours in three important ways. First, He's method uses a squared-error penalty, which, as argued above, is less appropriate for probability distributions than Kullback-Leibler divergence (Bishop, 1995, p. 226). As shown in Section 5.2, using squared error also results in worse performance in practice. Second, the exponentiated gradient descent method is applied to semi-supervised handwriting recognition and part-of-speech tagging, while we apply EGPR to an unsupervised genome annotation problem. Third, He et al. (He et al., 2013) use an exponentiated gradient descent strategy, while we use alternating optimization.

   Our alternating optimization approach has several benefits over the exponentiated gradient descent method of He et al. (He et al., 2013). First, the He et al. algorithm involves gradient calculations over each clique in the conditional dependence graph, with order $O(k^{|C|})$ for a variable of dimension k involved in cliques of size $|C|$. Our alternating minimization algorithm, by contrast, has closed-form updates of order $O(k)$ for q, r^M and s^M. (Updates for θ can still involve $O(k^{|C|})$ calculations, but these are generally very fast). Therefore, the alternating optimization algorithm is more appropriate for extremely large models or models with large cliques or high-order factors. Second, empirical studies of KL-based graph smoothness objectives have shown that alternating optimization algorithms perform better than gradient-based methods for these objectives (Subramanya & Bilmes, 2011). Finally, the alternating optimization strategy is parallelizable and extremely simple to implement because the graph regularization step has simple, closed-form updates with no learning rate hyperparameters, and the posterior calculation can be performed using any probabilistic inference method on a model with the same conditional dependence graph as the unregularized model.

9 Acknowledgments

This work was supported by NIH awards U41HG007000, R01ES024917, and R01GM103544 and by the Princess Margaret Cancer Foundation.

References

Altun, Yasemin, Belkin, Mikhail, and Mcallester, David A. Maximum margin semi-supervised learning for structured variables. In *NIPS*, pp. 33–40, 2005.

Ay, F., Bailey, T. L., and Noble, W. S. Statistical confidence estimation for Hi-C data reveals regulatory chromatin contacts. *Genome Research*, 24:999–1011, 2014a.

Ay, F., Bunnik, E. M., Varoquaux, N., Bol, S. M., Prudhomme, J., Vert, J.-P., Noble, W. S., and Le Roch, K. G. Three-dimensional modeling of the *P. falciparum* genome during the erythrocytic cycle reveals a strong connection between genome architecture and gene expression. *Genome Research*, 24:974–988, 2014b.

Bishop, C. *Neural Networks for Pattern Recognition*. Oxford UP, Oxford, UK, 1995.

Chapelle, O., Zien, A., and Schölkopf, B. (eds.). *Semi-supervised learning*. MIT Press, 2006.

Csiszár, I. and Tusnády, G. Information geometry and alternating minimization procedures. *Statistics and decisions*, pp. 205, 1984.

Das, D. and Petrov, S. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *NAACL*, pp. 600–609, 2011.

Das, D. and Smith, N.A. Semi-supervised framesemantic parsing for unknown predicates. In *Proc. of ACL*, 2011.

Day, N., Hemmaplardh, A., Thurman, R. E., Stamatoyannopoulos, J. A., and Noble, W. S. Unsupervised segmentation of continuous genomic data. *Bioinformatics*, 23(11):1424–1426, 2007.

Dekker, J., Rippe, K., Dekker, M., and Kleckner, N. Capturing chromosome conformation. *Science*, 295(5558):1306–1311, 2002.

Dixon, Jesse R, Selvaraj, Siddarth, Yue, Feng, Kim, Audrey, Li, Yan, Shen, Yin, Hu, Ming, Liu, Jun S, and Ren, Bing. Topological domains in mammalian genomes identified by analysis of chromatin interactions. *Nature*, 485(7398):376–380, 2012.

ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. *Nature*, 489:57–74, 2012.

Ernst, J. and Kellis, M. Discovery and characterization of chromatin states for systematic annotation of the human genome. *Nature Biotechnology*, 28(8):817–825, 2010.

Filion, G. J. et al. Systematic protein location mapping reveals five principal chromatin types in drosophila cells. *Cell*, 143 (2):212–224, 2010.

Ganchev, K., Graça, J., Gillenwater, J., and Taskar, B. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research*, 11:2001–2049, 2010.

He, Luheng, Gillenwater, Jennifer, and Taskar, Ben. Graph-based posterior regularization for semi-supervised structured prediction. In *CoNLL*, 2013.

Hoffman, M. M., Buske, O. J., Wang, J., Weng, Z., Bilmes, J. A., and Noble, W. S. Unsupervised pattern discovery in human chromatin structure through genomic segmentation. *Nature Methods*, 9(5):473–476, 2012a.

Hoffman, M. M. et al. Integrative annotation of chromatin elements from ENCODE data. *Nucleic Acids Research*, 41(2): 827–41, 2012b.

Joachims, T. Transductive inference for text classification using support vector machines. In *ICML*, pp. 200–209, 1999.

Lian, H., Thompson, W., Thurman, R. E., Stamatoyannopoulos, J. A., Noble, W. S., and Lawrence, C. Automated mapping of large-scale chromatin structure in encode. *Bioinformatics*, 24(17):1911–1916, 2008. PMC2519158.

Libbrecht, Maxwell W, Hoffman, Michael M, Bilmes, Jeffrey A, and Noble, William S. Entropic graph-based posterior regularization: Extended version. *Proceedings of the International Conference on Machine Learning*, 2015.

Lieberman-Aiden, Erez, van Berkum, Nynke L, Williams, Louise, Imakaev, Maxim, Ragoczy, Tobias, Telling, Agnes, Amit, Ido, Lajoie, Bryan R, Sabo, Peter J, Dorschner, Michael O, et al. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science*, 326(5950):289–293, 2009.

Misteli, T. Beyond the sequence: Cellular organization of genome function. *Cell*, 128(4):787–800, 2007.

Neal, R.M. and Hinton, G.E. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pp. 355–368. MIT Press, 1999.

Ryba, Tyrone, Hiratani, Ichiro, Lu, Junjie, Itoh, Mari, Kulik, Michael, Zhang, Jinfeng, Schulz, Thomas C, Robins, Allan J, Dalton, Stephen, and Gilbert, David M. Evolutionarily conserved replication timing profiles predict long-range chromatin interactions and distinguish closely related cell types. *Genome research*, 20(6):761–770, 2010.

Subramanya, A. and Bilmes, J. Semi-supervised learning with measure propagation. *Journal of Machine Learning Research*, 12(10):33113370, November 2011.

Subramanya, A., Petrov, S., and Pereira, F. Efficient graph-based semi-supervised learning of structured tagging models. In *Proc. of EMLNP 2010*, pp. 167–176. Assoc. for Comp. Linguistics, 2010.

Thurman, Robert E, Day, Nathan, Noble, William S, and Stamatoyannopoulos, John A. Identification of higher-order functional domains in the human encode regions. *Genome Research*, 17:917–927, 2007.

Wang, Jun, Jebara, Tony, and Chang, Shih-Fu. Graph transduction via alternating minimization. In *Proceedings of the 25th international conference on Machine learning*, pp. 1144–1151. ACM, 2008.

Woodfine, K., Fiegler, H., Beare, D. M., Collins, J. E., McCann, O. T., Young, B. D., Debernardi, S., Mott, R., Dunham, I., and Carter, N. P. Replication timing of the human genome. *Human molecular genetics*, 13(2):191–202, 2004.

Zhu, Xiaojin and Ghahramani, Zoubin. Learning from labeled and unlabeled data with label propagation. Technical report, 2002.

Zhu, Xiaojin, Kandola, Jaz S, Ghahramani, Zoubin, and Lafferty, John D. Nonparametric transforms of graph kernels for semi-supervised learning. In *NIPS*, volume 17, pp. 1641–1648, 2004.