Randomized Optimum Models for Structured Prediction

Daniel Tarlow
University of Toronto
Dept. of Computer Science

Ryan P. Adams
School of Engineering & Applied Sciences
Harvard University

Richard S. Zemel
University of Toronto
Dept. of Computer Science

Abstract

One approach to modeling structured discrete data is to describe the probability of states via an energy function and Gibbs distribution. A recurring difficulty in these models is the computation of the partition function, which may require an intractable sum. However, in many such models, the mode can be found efficiently even when the partition function is unavailable. Recent work on Perturb-and-MAP (PM) models (Papandreou and Yuille, 2011) has exploited this discrepancy to approximate the Gibbs distribution for Markov random fields (MRFs). Here, we explore a broader class of models, called Randomized Optimum models (RandOMs), which include PM as a special case. This new class of models encompasses not only MRFs, but also other models that have intractable partition functions yet permit efficient mode-finding, such as those based on bipartite matchings, shortest paths, or connected components in a graph. We develop likelihood-based learning algorithms for RandOMs, which, empirical results indicate, can produce better models than PM.

1 Introduction

One of the fundamental frustrations when modeling high-dimensional phenomena is that it is often easy to *score* configurations of data (e.g., using an energy function), but normalizing the distribution that this score implies may require enumerating a very large space (e.g., computing a partition function). When the normalizing constant is unknown, it becomes difficult to perform learning and inference, as one cannot assess which scoring functions give higher probability to the data. This can be somewhat alleviated when exact data can be generated from the model (Murray et al., 2006; Adams et al., 2009), but such situations are rare.

A curious computational phenomenon is that there are important and useful classes of energy-based models in which summing over all states is difficult, but *optimizing* is easy. That is, there are models in which the Gibbs-based probabilistic interpretation yields an unnormalizable model, but the configuration that minimizes the energy function (the mode, or maximum a posteriori (MAP) configuration) can be found exactly and efficiently. Two common examples of this are the graph cut algorithm for finding the mode of an associative (submodular) binary Markov random field (Kolmogorov and Zabih, 2004), and the bipartite matching problem, in which solutions can be found using the Hungarian algorithm (Munkres, 1957) in $\mathcal{O}(n^3)$ time, but computing the normalizing constant is $\#P$-complete (Valiant, 1979).

The complexity contrast between optimizing and normalizing has led to two different styles of learning in models of high-dimensional data. The first relies on optimization, but dispenses with a probabilistic interpretation, instead seeking to ensure that the ground truth has the optimal score. The energy-based models championed by LeCun et al. (2006), and structured SVMs trained via maximum margin methods (Tsochantaridis et al., 2005) both fall into this category. The second approach retains the probabilistic formulation, but utilizes approximate inference, such as loopy belief propagation, pseudo-likelihood, or sampling, to approximate the enumeration that is required for computing the probability of the data.

Our aim is to bridge this gap, developing probabilistic models that avoid sampling over complex discrete spaces, and the approximations required by enumeration. Recent work by Papandreou and Yuille (2011) has taken a step in this direction, in the particular case of Markov random fields. These authors reformulate a Gibbsian probabilistic model as injecting local noise into the energy function followed by mode-finding under the perturbed objective. They call this approach *Perturb-and-MAP*, and consider factorized noise injec-

Appearing in Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume 22 of JMLR: W&CP 22. Copyright 2012 by the authors.

tion based on the Gumbel distribution, which aims to approximate the Gibbs distribution implied by the original Markov random field.

In this paper, we develop a broader class of *Randomized Optimum models* (RandOMs), which are probabilistic models that exploit efficient optimization. RandOMs generalize previous work along three different axes. First, this new approach can be used to develop probabilistic models for complex structured spaces that are not typically modeled probabilistically, but in which global optima can be computed efficiently. We prove representational limitations to standard random field models and show that RandOMs can be used in a case where the standard graphical modeling representation fails. Second, we disconnect from attempting to approximate the Gibbs distribution and explore other ways of moving from an energy function to a probability distribution over structured outputs. Third, we develop new maximum likelihood-based learning schemes with different tradeoffs between efficiency and representation of uncertainty.

In total, we continue further in the direction of developing models that allow a better balance of probabilistic rigor with computational tractability. We see three main benefits to the approach: (a) it allows proper probabilistic modeling of structured domains: support is only given to configurations that satisfy domain-specific constraints; (b) an exact sample can be drawn from the models in polynomial time, without the need for a Markov chain; and (c) we believe our novel learning approaches to be a promising direction forward for learning with discrete output spaces that are more complex than are typically addressed by structured output learning. A key point related to the tractability of our approach is that we will map the complex discrete space into a well-behaved continuous latent space, then perform the difficult computations in the continuous space, where we can leverage powerful computational tools and concepts.

2 Randomized Optimum models

The goal of the Randomized Optimum model is to learn families of conditional distributions over highly structured spaces. The centerpiece of this approach is to specify these distributions in terms of the minima of randomized objective functions.

Let $\mathcal{Y}$ define a structured space of interest, such as the space of bipartite matchings, paths on a graph, or segmentations of a graph. An element $\boldsymbol{y} \in \mathcal{Y}$ is represented as the joint assignment of D categorical variables, $\boldsymbol{y} = \{y_1, \ldots, y_D\}$, where $y_d \in L_d = \{1, \ldots, K_d\}$, and K_d is the cardinality of variable d. In the supervised setting that we explore here, we wish to construct a distribution on this space by conditioning upon a

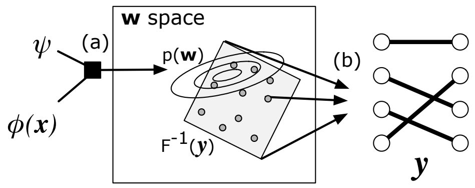


Figure 1: Overview of model. Starting from right to left: an observation of structured output $\boldsymbol{y}$ determines the inverse mapping set $\mathrm{F}^{-1}(\boldsymbol{y})$, which is denoted by the shaded region. We then have a region in $\boldsymbol{w}$ space that leads to $\boldsymbol{y}$ after a discrete optimization algorithm is run. $\boldsymbol{\psi}$ and $\boldsymbol{\phi}(\boldsymbol{x})$ define a distribution over $\boldsymbol{y}$ via $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$.

feature space $\mathcal{X}$. The approach we take is to mediate the interaction between $\mathcal{X}$ and $\mathcal{Y}$ via real-valued latent variables $\boldsymbol{w} \in \mathcal{W} = \mathbb{R}^P$, which we will discuss in more detail below. As an example, for paths on a graph, features may be properties of the edges and vertices.

An overview of the model is shown in Fig. 1. There are two main components of the model: (a) the mapping from features to a distribution in latent variable space, $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$; (b) the mapping from latent variable space to observations $\boldsymbol{y}$. In a standard CRF formulation, the (a) is deterministic while (b) is random. In the RandOM formulation, these are reversed: (a) is random, while (b) is deterministic. We discuss the relationship to standard CRF models more thoroughly in the Appendix, showing specifically how to map standard CRF notation into our formulation.

We will assume that there exists some matrix function $\boldsymbol{\phi} : \mathcal{X} \to \mathbb{R}^{F \times P}$ such that model parameters $\boldsymbol{\psi} \in \mathbb{R}^F$ can be used in conjunction with $\boldsymbol{\phi}(\boldsymbol{x})$ to define a distribution over $\boldsymbol{w}$, which will then be used to define a distribution over $\boldsymbol{y}$. Our objective is to find parameters $\boldsymbol{\psi}$ that most effectively specify a family of distributions $p(\boldsymbol{y} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x})) = \int_{\boldsymbol{w}} p(\boldsymbol{y} \mid \boldsymbol{w}) p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x})) d\boldsymbol{w}$, where $\boldsymbol{y} \in \mathcal{Y}$, given labeled data $\{\boldsymbol{x}_n, \boldsymbol{y}_n\}_{n=1}^N$.

In Randomized Optimum models, we define these distributions via a family of objective functions $f_{\boldsymbol{w}} : \mathcal{Y} \to \mathbb{R}$ which are indexed by the latent variables $\boldsymbol{w}$. Then, $\boldsymbol{y}$ comes not from a distribution given $\boldsymbol{w}$ but is the deterministic optimum of $f_{\boldsymbol{w}}(\boldsymbol{y})$. We assume that there is a unique minimum for $f_{\boldsymbol{w}}(\boldsymbol{y})$ for each $\boldsymbol{w}$ in $\mathcal{W}$, yielding a simple generative model:

$$\boldsymbol{w} \sim p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x})) \qquad \boldsymbol{y} = \operatorname*{argmin}_{\boldsymbol{y}'} f_{\boldsymbol{w}}(\boldsymbol{y}').$$

Given the features $\boldsymbol{x}$ and the model parameters $\boldsymbol{\psi}$, a random objective function is chosen via $\boldsymbol{w}$. The observed data are then specified as the unique minimum of this objective. We will denote this minimization as an implicit function $F(\boldsymbol{w}) = \operatorname{argmin}_{\boldsymbol{y}'} f_{\boldsymbol{w}}(\boldsymbol{y}')$, which has domain $\mathcal{W}$ and codomain $\mathcal{Y}$. We also make heavy use of the inverse set $F^{-1}(\boldsymbol{y})$, where:

$$\boldsymbol{w} \in F^{-1}(\boldsymbol{y}) \Rightarrow F(\boldsymbol{w}) = \boldsymbol{y} \qquad (1)$$

The core *probabilistic* component of this model is the conditional distribution $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$. In the generative model, this is the distribution that determines which optimization problem will be solved to find the dependent variable $\boldsymbol{y}$. Many regression models could be used for RandOMs, e.g., Hannah et al. (2011); however, for concreteness, the reader may think of the model in terms of simple basis function regression with spherical Gaussian noise and the mean constrained to the unit hypercube by the logistic function:

$$\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}), \nu \sim \mathcal{N}(\sigma(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\phi}(\boldsymbol{x})), \nu\mathbb{I}) \qquad (2)$$

where $\sigma(u) = (1 + \exp\{-u\})^{-1}$. We will always further assume that $\boldsymbol{w}$ is restricted to the unit hypercube.

2.1 Optimization Component of Model

Having specified the rest of the general framework, we turn our attention to the specification of $f_{\boldsymbol{w}}(\boldsymbol{y})$. In this paper we will focus on a set of specific cases in which Randomized Optimum models offer interesting computational advantages over alternative ways to specify families of distributions over structured spaces. There are three main requirements when choosing the objective function $f_{\boldsymbol{w}}(\boldsymbol{y})$. First, $f_{\boldsymbol{w}}(\boldsymbol{y})$ should enforce global constraints. For example, in the bipartite matching domain, $f_{\boldsymbol{w}}$ should enforce the constraint that each point is matched to exactly one other point. Second, the parameterization $\boldsymbol{w}$ should give enough flexibility to model the domain of interest. By modifying $\boldsymbol{w}$, we should be able to prefer favorable structured objects over unfavorable structured objects. Third, the function $\arg\min_{\boldsymbol{y}} f_{\boldsymbol{w}}(\boldsymbol{y})$ must be efficient to compute.

There are many possible functions $f_{\boldsymbol{w}}(\boldsymbol{y})$ that satisfy these requirements for various structured $\mathcal{Y}$. We consider three classes of such functions, noting that not all function within these classes will be suitable:

A. Standard exponential family models, where

$$f_{\boldsymbol{w}}(\boldsymbol{y}) = \langle \boldsymbol{w}, \rho(\boldsymbol{y}) \rangle, \qquad (3)$$

where $\rho(\boldsymbol{y})$ is the standard sufficient statistic in the exponential family representation of a MRF. This is the example considered by Papandreou and Yuille (2011), specifically in the case of a ferromagnetic Ising model. This standard CRF model supports efficient optimization via graph cuts.

B. Exponential family models with combinatorial base measure:

$$f_{\boldsymbol{w}}(\boldsymbol{y}) = \langle \boldsymbol{w}, \rho(\boldsymbol{y}) \rangle + \eta(\boldsymbol{y}). \qquad (4)$$

The function $\eta(\boldsymbol{y})$ specifies the support constraints for the model. That is, it assigns infinite cost in the objective function to values of $\boldsymbol{y}$ outside the desired sample space. For example $\eta(\boldsymbol{y})$ might encode the requirement that valid optima for $f_{\boldsymbol{w}}(\boldsymbol{y})$ be one-to-one matches in the bipartite matching problem. In the exponential family view, this can be thought of as the negative log of the base measure, which would take value infinity for values of $\boldsymbol{y}$ which should have zero probability. Note that representing these constraints within a graphical model formulation often require high order potentials and can be difficult to work with.

C. Beyond exponential families. These are functions $f_{\boldsymbol{w}}(\boldsymbol{y})$ that provably cannot be expressed in a standard graphical model formulation, even when using high order potentials, yet still admit efficient optimization. We give an example, make the above claim precise, and provide a proof in the Appendix.

This formulation allows a wide range of Randomized Optimum models with useful properties on structured spaces. In the remainder of this section, we examine two such cases of interest. Two other examples are given in the Appendix.

2.1.1 Example: Bipartite Matching

In the bipartite matching problem, we have an undirected bipartite graph $G = (\mathcal{V}, \mathcal{E})$, with equal-sized partite sets $\boldsymbol{A} \subseteq \mathcal{V}$ and $\boldsymbol{B} \subseteq \mathcal{V}$ i.e., $\boldsymbol{A} \cap \boldsymbol{B} = \emptyset$, $\boldsymbol{A} \cup \boldsymbol{B} = \mathcal{V}$, and $J = |\boldsymbol{A}| = |\boldsymbol{B}|$. Associated with each pair (or edge) $(a_i \in \boldsymbol{A}, b_j \in \boldsymbol{B})$ is a real value w_{ij}, which gives a cost incurred for matching a_i to b_j. The objective of bipartite matching is to choose a minimum cost set of pairs $\{(a_{i_1}, b_{j_1}), \ldots, (a_{i_{|\boldsymbol{A}|}}, b_{j_{|\boldsymbol{B}|}})\}$ such that each a_i and each b_j is paired with exactly one element of the opposing partite set. When (a_i, b_j) is included in the matching, we say that a_i is *matched* to b_j.

Bipartite matching enforces the concept of 1-to-1 correspondence, so it is not surprising that this structure arises often in real world problems in a variety of domains. Some notable examples include the problem of modeling correspondences in computer vision (Torresani et al., 2008); the assignment of workers to jobs in operations research; or computing word alignment in natural language processing (Taskar et al., 2005).

To express the bipartite matching problem in RandOM terms, we let $\boldsymbol{y} = \{y_{11}, \ldots, y_{JJ}\}$ be a length-J^2 boolean vector, where y_{ij} denotes the event that a_i is matched to b_j. In this case, the sufficient statistic is simply the identity: $\rho(\boldsymbol{y}) = \boldsymbol{y}$. Then $\boldsymbol{w} = \{w_{11}, \ldots, w_{JJ}\}$ is a length-J^2 real-valued vector, where w_{ij} gives the cost for matching a_i to b_j. Finally, $\eta(\boldsymbol{y})$ enforces the one-to-one matching constraint by assigning infinite cost to any $\boldsymbol{y}$ that is not a valid matching. It should then be clear that

$$f_{\boldsymbol{w}}(\boldsymbol{y}) = \langle \boldsymbol{w}, \rho(\boldsymbol{y}) \rangle + \eta(\boldsymbol{y}) = \sum_{ij} \boldsymbol{y}_{ij} \boldsymbol{w}_{ij} + \eta(\boldsymbol{y}), \quad (5)$$

and so minimizing $f_{\boldsymbol{w}}$ is equivalent to the minimum cost bipartite matching problem.

2.1.2 Example: Connected Components

There are many scenarios where observations may come in the form of a graph-structured clustering of points. For example, image segmentation can be thought of as a clustering of pixels over a lattice graph. One possible model of these outputs is as connected components, as in (Turaga et al., 2010). A natural goal is to define a proper probabilistic model of outputs generated by such a connected components procedure. This leads to an example of the third general type — that which cannot be expressed as a standard graphical model, even with high order terms. Given a graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$ with non-negative edge weights $\boldsymbol{w} = \{w_{ij} : (i,j) \in \mathcal{E}\}$ and a fixed threshold τ, we wish the minimum cost solution to be to label variables according to the connected component that they belong to, where an edge is deemed to exist if and only if its corresponding weight is above the threshold i.e. $w_{ij} > \tau$. To represent this labeling, we define variables $\boldsymbol{y} = \{y_1, \ldots, y_{|\mathcal{V}|}\} \in \{1, \ldots, |\mathcal{V}|\}^{|\mathcal{V}|}$, and force the label of all variables within a component to be the minimum index of any variable within that component. Under this definition, conditional on a given $\mathcal{G}$ and $\boldsymbol{w}$, there is a single desired labeling of variables, and all other labelings are equally undesirable.

More formally, we can describe $f_{\boldsymbol{w}}(\boldsymbol{y})$ using the following definitions:

$$\mathcal{P}(i,j) \text{ is the set of all paths from } i \text{ to } j \text{ in } \mathcal{G} \quad (6)$$

$$b_{ij}(\boldsymbol{w}) = \max_{p \in \mathcal{P}(i,j)} \min_{(l,m) \in p} w_{lm} \quad (7)$$

$$\mathcal{C}_i(\boldsymbol{w}) = \{j : i \neq j, b_{ij}(\boldsymbol{w}) > \tau\} \quad (8)$$

$$f_{\boldsymbol{w}}(\boldsymbol{y}) = \sum_{(i,j)\in\mathcal{E}} \mathbf{1}_{\{y_i = y_j\}} \max(0, b_{ij}(\boldsymbol{w}) - \tau) \quad (9)$$
$$+ \sum_{(i,j)\in\mathcal{E}} \mathbf{1}_{\{y_i \neq y_j\}} \max(0, \tau - b_{ij}(\boldsymbol{w})) + \eta(\boldsymbol{y})$$

$$\eta(\boldsymbol{y}) = \lim_{a\to\infty} a \cdot \left(1 - \prod_{i\in\mathcal{V}} [y_i = \min_{j\in\mathcal{C}_i(\boldsymbol{w})} j]\right). \quad (10)$$

It should be clear that by assigning $\boldsymbol{y}$ as the result of a connected components algorithm, $f_{\boldsymbol{w}}(\boldsymbol{y}) = 0$. All other assignments have $f_{\boldsymbol{w}}(\boldsymbol{y}) > 0$, so we can exactly and efficiently compute the arg-minimum of this $f_{\boldsymbol{w}}$. An interesting question is whether there exists an exponential family representation (i.e., in the form of Eq. 4) of this model. We will prove that this is impossible via the geometry of the inverse set $F^{-1}(\boldsymbol{y})$.

3 Inverse Set Geometry

We believe a productive means of thinking about RandOM models is via the geometry of the inverse mapping set $F^{-1}(\boldsymbol{y})$. It is this geometry that determines the tractability of a RandOM as $f_{\boldsymbol{w}}$ becomes more complex. Due to space constraints, discussion of important geometric properties, along with proofs, will be discussed in the Appendix. Here, we simply state the important results:

Proposition 1. *When $f_{\boldsymbol{w}}(\boldsymbol{y})$ is defined as the exponential family (possibly with combinatorial base measure) as in Eq. 4, $F^{-1}(\boldsymbol{y})$ is a convex set.*

Proposition 2. *The inverse set $F^{-1}(\boldsymbol{y})$ for the connected components problem from Section 2.1.2 is a star-convex set.*

Lemma 1. *Let $f_{\boldsymbol{w}}(\boldsymbol{y})$ be defined as the connected components objective in Eq. 9. There is no equivalent expression of $f_{\boldsymbol{w}}(\boldsymbol{y})$ in the exponential family form Eq. 4.*

4 Learning RandOMs

The maximum likelihood learning objective is to maximize the amount of probability mass assigned to the ground truth labeling by the model. In the RandOM formulation, because the relationship between latent variables $\boldsymbol{w}$ and observations $\boldsymbol{y}$ is deterministic, this amounts to placing as much mass of $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$ as possible inside the $F^{-1}(\boldsymbol{y})$ sets. In this section, we design learning algorithms for RandOMs that take advantage of the structure of $F^{-1}(\boldsymbol{y})$.

The principal challenge we need to address is how to handle the unknown latent variable $\boldsymbol{w}$ for each training case. For any given observed configuration $\boldsymbol{y}$, there is a set of $\boldsymbol{w}$ corresponding to objective functions for which $F(\boldsymbol{w}) = \boldsymbol{y}$. Recall that this set is the inverse mapping set $F^{-1}(\boldsymbol{y})$. The likelihood of $\boldsymbol{\psi}$ requires integrating over this set:

$$p(\{\boldsymbol{y}_n\}_{n=1}^N \mid \boldsymbol{\psi}, \{\boldsymbol{x}_n\}_{n=1}^N) = \prod_{n=1}^N \int_{F^{-1}(\boldsymbol{y}_n)} p(\boldsymbol{w}_n \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}_n))\, \mathrm{d}\boldsymbol{w}_n.$$

Because the optimization procedure is a many-to-one mapping, and since exactly integrating over $F^{-1}(\boldsymbol{y})$ is intractable (even in the case where $F^{-1}(\boldsymbol{y})$ is a convex set and $p(\boldsymbol{w}_n \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}_n))$ is Gaussian), there are essentially two practical approaches: approximately integrate over $\boldsymbol{w}$ using Markov chain Monte Carlo (MCMC); or find a point estimate for $\boldsymbol{w}$ by solving a constrained optimization problem. In this section, we will discuss formulations of both of these approaches where efficient combinatorial algorithms can be used to make learning more efficient. An orthogonal issue that we will not discuss further is the treatment of $\boldsymbol{\psi}$. Here, we will find a point estimate, but it would be straightforward to integrate over it via sampling.

4.1 Monte Carlo Expectation Maximization

Although we cannot generally analytically integrate over the pre-image set, we can nevertheless instantiate the latent variables $\{\boldsymbol{w}_n\}_{n=1}^N$ explicitly in a Markov chain, drawing samples using a specialized MCMC procedure. This general approach has the property that it allows for maximal flexibility in the choice of $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$, enabling any option that would be available in the case where the $\boldsymbol{w}$ themselves were the observations, such as nonparametric mixture models. We emphasize that sampling over $F^{-1}(\boldsymbol{y})$ is very different from the typical sampling over a complex discrete space.

More specifically, we write the joint distribution $p(\boldsymbol{\psi}, \{\boldsymbol{w}_n, \boldsymbol{y}_n, \boldsymbol{x}_n\}_{n=1}^N)$ as,

$$p(\boldsymbol{\psi}) \prod_{n=1}^N p(\boldsymbol{w}_n \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}_n))\, \mathbf{1}_{\{\boldsymbol{w}_n \in F^{-1}(\boldsymbol{y}_n)\}} \qquad (11)$$

Note that $p(\boldsymbol{\psi}, \{\boldsymbol{w}_n, \boldsymbol{y}_n, \boldsymbol{x}_n\}_{n=1}^N)$ has two parts. The first part, $p(\boldsymbol{\psi}) \prod_{n=1}^N p(\boldsymbol{w}_n \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}_n))$, is any standard regression model for $\boldsymbol{w}_n$ given inputs $\boldsymbol{x}_n$, using parameters $\boldsymbol{\psi}$. The non-standard part of this formulation comes from the second part — the indicator function $\mathbf{1}_{\{\boldsymbol{w}_n \in F^{-1}(\boldsymbol{y}_n)\}}$, which constrains each $\boldsymbol{w}_n$ to lie in the inverse mapping set $F^{-1}(\boldsymbol{y}_n)$ by assigning zero probability to any $\boldsymbol{w}_n$ that lies outside the set. One of the main operations that we will need to repeatedly perform during learning is to check whether a given $\boldsymbol{w}_n$ lies within $F^{-1}(\boldsymbol{y}_n)$. This "set membership" query could be done simply by running the optimization algorithm associated with the RandOM. We will discuss a much more efficient strategy, and its tailoring specifically to slice sampling, in Section 4.1.1.

Monte Carlo expectation maximization (MCEM) (Wei and Tanner, 1990) alternates between computing a Monte Carlo estimate of the expected complete data log likelihood (the E-step) and maximizing $\boldsymbol{\psi}$ given this approximation (the M-step). In MCEM iteration t, we use the procedure described in Section 4.1.1 to generate K samples of the latent variables $\boldsymbol{w}_n$ using the current parameter estimate $\boldsymbol{\psi}^{(t)}$: $\boldsymbol{w}_{n,k}^{(t)} \sim p(\boldsymbol{w}_n \mid \boldsymbol{\psi}^{(t)}, \boldsymbol{x}_n) \cdot \mathbf{1}_{\{\boldsymbol{w}_n \in F^{-1}(\boldsymbol{y}_n)\}}$. These samples are then used to estimate the complete data log likelihood:

$$\mathbb{E}_{\boldsymbol{w}^{(t)}} \left[\ln p(\{\boldsymbol{y}_n, \boldsymbol{w}_n\}_{n=1}^N \mid \boldsymbol{\psi}, \{\boldsymbol{x}_n\}_{n=1}^N) \right] \approx \sum_{n=1}^N \ln \left\{ \frac{1}{K} \sum_{k=1}^K p(\boldsymbol{w}_{n,k}^{(t)} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}_n)) \right\}. \quad (12)$$

This Monte Carlo approximation to the E-step is then used in the M-step to find the new parameter settings:

$$\boldsymbol{\psi}^{(t+1)} = \operatorname*{argmax}_{\boldsymbol{\psi}} \sum_{n=1}^N \ln \left\{ \frac{1}{K} \sum_{k=1}^K p(\boldsymbol{w}_{n,k}^{(t)} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}_n)) \right\}.$$

In each E-step, the Markov chain for each latent variable is initialized to its ending value from the previous step. As the inverse mapping does not change during updates to $\boldsymbol{\psi}$, the new starting location for $\boldsymbol{w}_n$ is guaranteed to lie within $F^{-1}(\boldsymbol{y}_n)$. In the experimental section, we explore the effect of varying the number of samples to draw in each E-step, considering the case of using as few as 1 sample per instance per iteration.

4.1.1 Specialized Slice Sampler

Critical to MCEM (and fully-Bayesian inference) is the task of sampling the latent $\boldsymbol{w}$. In this section, we will discuss why this problem is particularly well suited to slice sampling (Neal, 2003), and we will show how dynamic combinatorial algorithms and the inner workings of slice sampling can be made to work synergistically together. For a detailed review of slice sampling, we recommend Neal (2003). We assume familiarity of basic slice sampling, and give a detailed description of our implementation of slice sampling, specialized for the case of bipartite matching models, in Algorithm 2 (see Appendix). The same principles apply for other RandOMs. Here we discuss the non-standard properties of our specialized slice sampler.

First, we maintain algorithmic state $\mathcal{A}_{\boldsymbol{y}}$ throughout the execution of the sampling algorithm. Whenever we make a set membership query (In-Inverse-Set in Alg. 2), we warm-start from the solution of the previous call, using dynamic combinatorial optimization e.g. for bipartite matching problems, we use the dynamic Hungarian algorithm of Korsah et al. (2007).

Second, we re-sample blocks of $\boldsymbol{w}$ so as to be compatible with the dynamic combinatorial algorithm. In the bipartite matching case, we resample a subset of dimensions of $\boldsymbol{w}$ at a time, where the subset corresponds to the terms that define the costs for a single row of the bipartite matching cost matrix. With this choice, the dynamic Hungarian algorithm can check inverse set membership in worst case $\mathcal{O}(J^2)$ time rather than the worst case $\mathcal{O}(J^3)$ time that would be required if all entries of $\boldsymbol{w}$ changed. More importantly, empirically, the membership checks after a single row update often behave as $\mathcal{O}(1)$ or $\mathcal{O}(J)$, depending on how many values of $F(\boldsymbol{w})$ change as a result of the modification. If we modify $\boldsymbol{w}$ without leaving the $F^{-1}(\boldsymbol{y})$ set, then by definition, $F(\boldsymbol{w})$ has not changed, making the dynamic algorithm very fast. When the sampler does step outside $F^{-1}(\boldsymbol{y})$, it is typically by a small amount, leading to a new $F(\boldsymbol{w})$ that is not far from $\boldsymbol{y}$. One elaboration that we leave to future work is whether the call to the dynamic combinatorial algorithm can be halted early as soon as the solution is provably different from $\boldsymbol{y}$.

Third, many set membership queries are avoided altogether. This can be done whenever the intersection of

$F^{-1}(\boldsymbol{y})$ with the subspace in which we are resampling is a convex set. This is clearly the case when $F^{-1}(\boldsymbol{y})$ is itself a convex set. Even in cases where $F^{-1}(\boldsymbol{y})$ is not a convex set, it may be that the intersection with the resampling subspace is convex. We can avoid certain set membership calls by the following reasoning: we are guaranteed inductively that the initial point $\boldsymbol{w}$ lies within $F^{-1}(\boldsymbol{w})$. The alternative points that we encounter during slice sampling all lie on the line $\boldsymbol{w}+\alpha\cdot\boldsymbol{s}$ for some $\alpha \in \mathbb{R}$. Thus, if we observe that $\boldsymbol{w}+\alpha\cdot\boldsymbol{s}$ also lies within $F^{-1}(\boldsymbol{w})$, then we can infer that $\boldsymbol{w}+\alpha'\cdot\boldsymbol{s}$ lies within $F^{-1}(\boldsymbol{w})$ for all values $\alpha' \in [\alpha, 0]$ (if $\alpha < 0$) or $\alpha' \in [0, \alpha]$ (if $\alpha > 0$). When we encounter a point that can be proven to lie within $F^{-1}(\boldsymbol{w})$, there is clearly no reason to issue a set membership query.

When $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$ is log concave and $F^{-1}(\boldsymbol{w})$ is a convex set, then the restriction of $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$ to $F^{-1}(\boldsymbol{w})$ will also be log concave. Practically, this means that the slice sampler will never have to traverse difficult energy barriers to reach any part of the space. From a theoretical point of view, it guarantees that that the slice sampler is "perfect" in that each step is drawing a uniform sample from the slice and is not simply leaving the uniform distribution invariant.

4.2 Hard EM

Rather than integrating $\int p(\boldsymbol{w} \mid \boldsymbol{y}, \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))\mathrm{d}\boldsymbol{w}$ in the E-step, we can instead maximize over $\boldsymbol{w}$. This yields a "hard EM" algorithm, in which the E-step seeks the $\boldsymbol{w}$ in $F^{-1}(\boldsymbol{y})$ with maximum density $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$:

$$\underset{\boldsymbol{w}}{\operatorname{argmax}}\, p(\boldsymbol{w} \mid \boldsymbol{y}, \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x})) = \underset{\boldsymbol{w} \in F^{-1}(\boldsymbol{y})}{\operatorname{argmax}}\, p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x})).$$

The methods presented in this section assume that $f_{\boldsymbol{w}}$ is defined as the exponential family form of Eq. 4, so $F^{-1}(\boldsymbol{y})$ is a convex polytope. If $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$ is log-concave in $\boldsymbol{w}$, the E-step is a convex optimization problem, constrained to a convex polytope. In the case that $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$ is Gaussian distributed, the optimization problem is a quadratic program (QP):

$$\textbf{min.}_{\boldsymbol{w}} \quad (\boldsymbol{w} - \hat{\boldsymbol{w}})^T(\boldsymbol{w} - \hat{\boldsymbol{w}}) \tag{13}$$

$$\textbf{s.t.} \quad f_{\boldsymbol{w}}(\boldsymbol{y}) \le f_{\boldsymbol{w}}(\boldsymbol{y}') \qquad \forall \boldsymbol{y}' \ne \boldsymbol{y}, \tag{14}$$

where $\hat{\boldsymbol{w}}$ is the mode of $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$.

The difficulty is that the number of constraints needed to define $F^{-1}(\boldsymbol{y})$ (and thus the QP) is often exponential in $|\boldsymbol{y}|$. For all but small problems, it is intractable to explicitly instantiate the QP. However, as an efficient separation algorithm exists, the problem can be optimized by a cutting planes approach.

From an algorithmic perspective, we have arrived at an algorithm that bears some resemblance to a standard structural SVM formulation (Tsochantaridis et al., 2005). The most obvious similarity is that both approaches involve formulating a QP that optimizes over model parameters, where the QP constraints are of the form that the ground truth $\boldsymbol{y}$ has better objective than all other assignments $\boldsymbol{y}'$. To find violated constraints in the QP formulation of structural SVMs (or to find subgradients in the empirical risk minimization formulation) the same inner loop is used as in this hard EM formulation. The main difference in the inner loop is that the objective of the hard EM QP encourages $\boldsymbol{w}'$ to be near $\hat{\boldsymbol{w}}$, the mode of $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$, versus the QP objective that encourages that $\boldsymbol{w}'$ be near $\mathbf{0}$.

Algorithm 1 Hard EM+Margin: E-step

```
$\boldsymbol{w} \leftarrow \hat{\boldsymbol{w}}$      $\mathcal{S} \leftarrow \emptyset$            // Active set of constraints.
while 1 do
    $\boldsymbol{y}' \leftarrow \arg\min_{\hat{\boldsymbol{y}} \ne \boldsymbol{y}} f_{\boldsymbol{w}}(\hat{\boldsymbol{y}})$
    if $\boldsymbol{y}' \in \mathcal{S}$ then
        break
    else
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{\boldsymbol{y}'\}$
    end if
    $\boldsymbol{w} \leftarrow \arg\min_{\boldsymbol{w}', \xi \ge 0} (\boldsymbol{w}' - \hat{\boldsymbol{w}})^T(\boldsymbol{w}' - \hat{\boldsymbol{w}}) - C\xi$
              s.t.  $f_{\boldsymbol{w}'}(\boldsymbol{y}) \le f_{\boldsymbol{w}'}(\boldsymbol{y}') - \xi$      $\forall \boldsymbol{y}' \in \mathcal{S}$
end while
```

The second major difference is that the QP variables in this hard EM formulation are latent variables $\boldsymbol{w}$, while the variables in the QP from the structural SVM formulation are more analogous to our $\boldsymbol{\psi}$ parameters. In fact, we could make the connection to the structural SVM formulation clearer by making the additional assumption that the relationship between $\boldsymbol{\psi}$ and $\boldsymbol{w}$ is deterministically set to $\boldsymbol{w}_n = \boldsymbol{\psi}^T \phi(\boldsymbol{x}_n)$. In this case, the QPs for all data cases n must be combined into one QP and solved simultaneously, like in a structural SVM algorithm. Mediating the interaction of $\boldsymbol{\psi}$ and $\boldsymbol{y}$ via random $\boldsymbol{w}$ allows for the probabilistic nature of the RandOM approach.

A practical problem with this hard EM algorithm is that the solutions $\boldsymbol{w}$ often lie on the boundary of the polytope $F^{-1}(\boldsymbol{y})$. This leads to undesirable behavior in the M-step, where significant mass is placed outside $F^{-1}(\boldsymbol{y})$. To counteract this bias, we modify the QP objective to encourage finding points that have high density under $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$ but that are deeper inside $F^{-1}(\boldsymbol{y})$. We do this by adding a margin term to the QP objective, leading to the modified Hard EM QP given in Alg. 1. There, C is a parameter that trades off the margin between $\boldsymbol{w}$ and the boundary of $F^{-1}(\boldsymbol{y})$ with maximizing $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$.

5 Experimental Evaluation

The main experimental questions we focus on concern the representational capabilities of Randomized Optimum models. We ran two sets of experiments. First, we experimented with synthetic data, and second, we applied a bipartite matching RandOM to a real-world lung CT scan registration task from medical imaging.

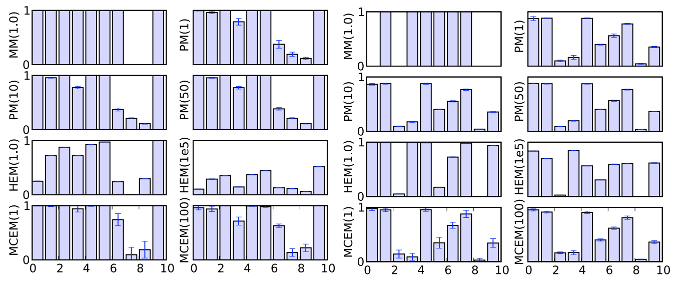
The goal is to learn to align landmarks in 3D lung volumes between two phases of the respiratory process. We show that RandOMs achieve higher accuracy than not only the Perturb-and-MAP models, but also a max-margin model trained structural SVM on the same features.

Evaluation: To evaluate the probability assigned to a data point $(\boldsymbol{x}, \boldsymbol{y})$ by the model, we draw samples $\boldsymbol{y}_k$ from $p(\boldsymbol{y} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$ using learned parameters ψ, and we count the fraction of samples that match the ground truth, i.e. $\frac{1}{K}\sum_k \mathbf{1}_{\{\boldsymbol{y}_k=\boldsymbol{y}_n\}}$. For our experiments here, we use K=10000.

Synthetic Matching Experiments: We generated synthetic data sets for bipartite matching models by sampling J landmark points (set $\boldsymbol{A}$) uniformly in the 2D box $[-1, 1]^2$. For each point, we also generated a real-valued size and 3D color vector in RGB space. We then generated J target points (set $\boldsymbol{B}$) by associating one point in $\boldsymbol{B}$ with one point in $\boldsymbol{A}$, then drawing the properties of $\boldsymbol{B}$ by perturbing the associated point in $\boldsymbol{A}$. In Dataset 1, we color the points in $\boldsymbol{B}$ to be uniformly random (so color becomes an irrelevant feature), while noisily drawing size and position from a distribution centered at the value for the associated point in $\boldsymbol{A}$. In Dataset 2, we use a multimodal generation process; with probability $\frac{1}{2}$, we follow a similar procedure as above, but only set position to be relevant, and with probability $\frac{1}{2}$ set only color to be relevant. The features $\{\phi_{fij}(\boldsymbol{x}) | f \in \{\text{position, color, size}\}\}$ are length-J^2 real vectors, where there is an entry for each $(i \in \boldsymbol{A}, j \in \boldsymbol{B})$ pair. We set the entry of $\phi_{fij}(\boldsymbol{x})$ corresponding to (i, j) to be the squared distance between a_i and b_j along dimension f.

Results for the first data set are shown in Fig. 2. PM denotes Perturb-and-MAP, HEM denotes Hard EM, MCEM denotes Monte Carlo EM, and MM denotes max-margin. In parenthesis are the parameter setting, denoting the margin parameter, or the number of samples per iteration per instance. Comparing the MCEM algorithms to PM, we see that MCEM slightly outperforms PM, but that both do a good job in assigning mass to all training points. This is interesting, because it shows that Randomized Optimum models that do not attempt to approximate the standard Gibbs distribution can still have good representational power.

Quantitative results for both synthetic data sets are shown in Fig. 3. We compare the probability assigned to the data by each algorithm with several parameter settings. If the probability assigned to an instance was small enough that no samples were drawn that exactly matched the ground truth, then we remove that instance's contribution from the log probability



(a) "Unimodal" data set. (b) "Multimodal" dataset.

Figure 2: Distribution of probability mass placed on the data by different learning algorithms. Along the x-axis are different training examples. The height of the bar shows the fraction of samples drawn from the model that matched the ground truth. $J = 5$, and $N = 10$.

(otherwise it would go to $-\infty$) and report this failure in the second column. Plots of how the log probability is assigned to data instances are shown in Fig. 2.

Amongst the Hard EM (HEM) algorithms, we see that including the term that rewards increasing the margin does improve performance. Increasing the term to be very large encourages finding the point that has largest margin, regardless of where mass of the distribution $p(\boldsymbol{w} \mid \boldsymbol{\psi}, \boldsymbol{\phi}(\boldsymbol{x}))$ lies. This causes the model to place nonzero mass on every instance, but it must increase its uncertainty, so the amount of mass placed on each instance decreases. We believe the reason why some data points are given negligible mass is because the E step is not guaranteed to find a point inside $F^{-1}(\boldsymbol{y})$ if the cutting planes algorithm is only run for a small number of iterations (as is done in these experiments).

As is particularly evident in the "multimodal" setting, the max-margin approach cannot always find a setting of $\boldsymbol{\psi}$ that perfectly separates the ground truths from all other assignments. In geometric terms, the subspaces spanned by the rows of $\phi(\boldsymbol{x}_n)$ do not intersect $F^{-1}(\boldsymbol{y}_n)$ for all n. While the probabilistic algorithms are able to place mass in all inverse mapping sets using distributions in $\boldsymbol{w}$ space, the MM approach is not.

In both experiments, PM and MCEM achieve the best performance. Interestingly, while increasing the number of samples per iteration seems to improve MCEM, PM does not seem to benefit. In both cases, the top performing algorithm is MCEM with 100 samples drawn per instance per iteration in the E step.

Lung CT Registration Task: We obtained volumetric CT scans of pairs of lungs at different phases of the respiratory process, from 10 subjects (Castillo et al., 2009).[1] Included with the data are a set of hand-labeled landmarks giving ground truth correspondences between points in the pairs of lung vol-

[1]Available at http://www.dir-lab.com/index.html

Alg	Avg. Log Prob	# Prob=0
Max-margin	0	2
PM(1)	-0.51	0
PM(10)	-0.50	0
PM(50)	-0.50	0
HEM(0.0)	-0.74	2
HEM(1.0)	-1.09	0
HEM(1e5)	-1.59	0
MCEM(1)	**-0.44**	**0**
MCEM(100)	**-0.44**	**0**

(a) "Unimodal" data

Alg	Avg. Log Prob	# Prob=0
Max-margin	0	4
PM(1)	-1.06	0
PM(10)	-1.06	0
PM(50)	-1.06	0
HEM(0.0)	-1.08	2
HEM(1.0)	-1.09	1
HEM(1e5)	-1.38	0
MCEM(1)	-1.07	0
MCEM(100)	**-0.97**	**0**

(b) "Multimodal" data

Figure 3: Quantitative results on synthetic data.

umes. We take 30 corresponding landmarks from each pair of lungs, and we extract several features for each landmark in each volume. We use a total of 20 features, including 3D position, mean intensity of varying sized patches around the landmark, standard deviation of intensity in varying sized patches around the landmark, and the Hessian-based "vesselness" filter of Frangi et al. (1998) at several scales. As in the synthetic data, for each landmark feature, we compute $\phi_{fij}(\boldsymbol{x})$ for a pair (a_i, b_j) as the squared difference of the features for a_i and b_j along dimension f.

We split the 10 instances into 10 different training sets, where we train on 9 instances and evaluate based on predictive performance on the held out 10th instance. The results are averaged over the 10 splits and are reported in Fig. 4. The columns show the fraction of predicted pairs that are correct (the "score"), along with average probability assigned to the test data, and average log probability assigned to the test data. Parentheses in the last column give the number of cases where the model assigned low enough probability to the test data such that we drew no samples that exactly matched the true test labeling.

We compare MCEM to PM and max-margin baselines. Here, though MCEM displays a small amount of overconfidence, it outperforms both baselines in overall score; that is, the Randomized Optimum models produce the most accurate predictions on this task.

6 Discussion and Related Work

To our knowledge, the line of work that includes Papandreou and Yuille (2011) and this work is unique in the technique of defining generative probabilistic models that include include an explicit optimization procedure. However, the motivating ideas are present in several works, which build probabilistic models around specialized tractable computational structures.

Alg	**Avg. Score**	**Avg. Prob**	**Avg. Log Prob**
Max-margin	91.0%	.400	0 (6)
PM(1)	95.0%	.538	-1.52 (0)
PM(10)	94.9%	.543	-1.54 (0)
MCEM(1)	98.8%	.837	-1.67 (2)
MCEM(5)	96.8%	.568	-0.62 (1)
MCEM(10)	96.7%	.544	-0.56 (1)

Figure 4: Quantitative results on lung CT data.

For example, Determinental Point Processes (Kulesza and Taskar, 2011) define probability models around the computation of the determinant of a matrix. Domke (2011) defines probability models around a fixed number of iterations of belief propagation. Neither of these computations is a discrete optimization of the type we consider, but the spirits are similar.

There is also work on learning under the assumption that test-time inference will involve running a discrete optimization procedure. Structured output SVMs are perhaps the most popular, and they have been widely applied e.g., Taskar et al. (2005); Ratliff et al. (2006); Szummer et al. (2008); Joachims et al. (2009).

In this paper we have presented a framework for defining probabilistic models that exploit efficient methods for optimizing the scoring function. The models derive from a simple generative procedure that allows a rich class of probabilistic models. While we have shown how a previous model, Perturb-and-MAP, can be applied beyond settings typically modeled as conditional random fields, RandOMs are more general than even the extended version of Perturb-and-MAP. The learning procedures we provide show how to directly formulate a maximum likelihood learning objective. The experimental results show this learning to be both tractable and successful in a real-world application.

RandOMs trade enumerating an exponential number of configurations for an integration of the inverse mapping polytope. We believe moving the difficult computations to the continuous latent space is desirable, because it allows the use of powerful continuous-variable tools to be used in the context of structured discrete output spaces.

Finally, we have only begun to explore the representational power of the framework. RandOMs can readily incorporate other generative distributions over the latent variables, such as Gaussian Processes, mixture models, or various nonparametric models. Exploring these alternatives is a focus of our future work.

Acknowledgements

We thank Juan Eugenio Iglesias for help designing features for the lungs model, along with help preparing the data.

References

R. P. Adams, I. Murray, and D. J. C. MacKay. The Gaussian process density sampler. In *NIPS 21*, 2009.

R. Castillo, E. Castillo, R. Guerra, V. Johnson, T. McPhail, A. Garg, and T. Guerrero. A framework for evaluation of deformable image registration spatial accuracy using large landmark point sets. *Phys Med Biol*, 54:1849–1870, 2009.

W. H. Cunningham. Minimum cuts, modular functions, and matroid polyhedra. *Networks*, 15(2):205–215, 1985.

J. Domke. Parameter learning with truncated message-passing. In *CVPR*, 2011.

A. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever. Multiscale vessel enhancement filtering. In *MICCAI*, 1998.

L. A. Hannah, D. M. Blei, and W. B. Powell. Dirichlet process mixtures of generalized linear models. *JMLR*, 12:1923–1953, 2011.

T. Joachims, T. Finley, and C.-N. Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77 (1):27–59, 2009.

V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE PAMI*, 26 (2):147–159, 2004.

G. Korsah, A. Stentz, and M. Dias. The dynamic Hungarian algorithm for the assignment problem with changing costs. Technical Report CMU-RI-TR-07-27, Robotics Institute, Pittsburgh, PA, July 2007.

A. Kulesza and B. Taskar. Structured determinantal point processes. In *NIPS 23*, 2011.

Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press, 2006.

J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, 1957.

I. Murray, Z. Ghahramani, and D. J. C. MacKay. MCMC for doubly-intractable distributions. In *UAI*, pages 359–366, 2006.

R. M. Neal. Slice sampling. *Annals of Statistics*, 31 (3):705–767, 2003.

V. Ng. Graph-cut-based anaphoricity determination for coreference resolution. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2009.

G. Papandreou and A. Yuille. Perturb-and-MAP random fields: Using discrete optimization to learn and sample from energy models. In *ICCV*, 2011.

N. Ratliff, J. A. Bagnell, and M. Zinkevich. Maximum margin planning. In *ICML*, 2006.

C. Rother, V. Kolmogorov, and A. Blake. Grab-Cut: interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.

C. Smith. A characterization of star-shaped sets. *American Mathematical Monthly*, 75(4), 1968.

M. Szummer, P. Kohli, and D. Hoiem. Learning CRFs using graph cuts. In *ECCV*, 2008.

B. Taskar, S. Lacoste-Julien, and D. Klein. A discriminative matching approach to word alignment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 73–80, 2005.

L. Torresani, V. Kolmogorov, and C. Rother. Feature correspondence via graph matching: Models and global optimization. In *ECCV*. 2008.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, 2005.

S. C. Turaga, K. L. Briggman, M. Helmstaedter, W. Denk, and H. S. Seung. Maximin affinity learning of image segmentation. In *NIPS*, 2010.

L. G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.

Wainwright and Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 2008.

H. Wang. *Proteins, Interactions, and Complexes: A Computational Approach*. PhD thesis, Department of Computer Science, Stanford University, 2008.

G. C. G. Wei and M. A. Tanner. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):pp. 699–704, 1990.

Appendix

7 More Example RandOM Constructions

7.1 Example: Shortest Paths

In the *st*-shortest path problem, we are given a directed graph $G=(\mathcal{V},\vec{\mathcal{E}})$, along with two special nodes, a source s, and a destination t. Each directed edge (i,j) has a cost w_{ij}. The goal of the *st*-shortest path problem is to find a path from s to t such that the sum of edge costs along the path is minimized.

Just as matchings represent a certain type of fundamental structure, so do shortest paths. For example, consider observations of people walking through their neighborhood from home to work. A natural model of these observations is that people have a cost function for traversing sections of road or sidewalk that depend on features such as length, scenery, crowdedness, or safety. To get between two points, we might suppose that a person chooses the path that has lowest cost under their (to us, unobserved) cost function.

In RandOM terms, we let $\boldsymbol{w}$ be a vector of length $|\vec{\mathcal{E}}|$, where there is a cost w_{ij} for each directed edge (i,j). A shortest path is uniquely defined by the set of edges that it traverses, so we can represent the path using binary variables $\boldsymbol{y}=\{y_{ij}|(i,j)\in\vec{\mathcal{E}}\}$, where y_{ij} indicates that the path traversed the edge from i to j. As before, we let the sufficient statistic be the identity $\rho(\boldsymbol{y})=\boldsymbol{y}$, and we let $\eta(\boldsymbol{y})$ enforce the constraint that $\boldsymbol{y}$ corresponds to a valid *st*-path.[2] It is then clear that minimizing $f_{\boldsymbol{w}}$ is equivalent to finding the shortest *st*-path.

7.1.1 Example: Minimum Cuts

Another example is finding the minimum cut in a graph. Given an undirected graph $G=(\mathcal{V},\mathcal{E})$ with nonnegative edge weights $\{\boldsymbol{w}_{ij}:(i,j)\in\mathcal{E}\}$, the minimum cut problem is to split the graph into two subsets $\boldsymbol{S}$ and $\bar{\boldsymbol{S}}$ such that the sum of the weights of the "cut" edges is minimized. An edge is considered cut if one endpoint lies in $\boldsymbol{S}$ and the other lies in $\bar{\boldsymbol{S}}$.

There exist efficient algorithms for finding the minimum cut in graphs with arbitrary topology. This is particularly useful because of the connection between finding a minimum cut in a graph and minimizing graph-structured submodular functions over binary variables. That is, these submodular minimization problems can be solved exactly via reduction to min-cut (Cunningham, 1985), which has led to widespread practical application. One example is the so-called "graph cuts" algorithm, which is a workhorse of computer vision. The most common use in computer vision is binary image segmentation, where the goal is to label each pixel in an image as belonging to one of two semantic classes. For example, separating foreground from background or object (e.g., airplane) from not-object. This task is fundamental in many vision applications, can be a subtask for non-binary image segmentation, and has applications on its own to image editing (Rother et al., 2004). The effectiveness of the available optimization procedures has also led to the problem being used to model protein-protein interactions in computation biology (Wang, 2008), and in natural language processing (Ng, 2009).

To express the minimization in RandOM terms, we use the standard connection with graph-structured submodular energy functions over binary variables. Given a graph $G=(\mathcal{V},\mathcal{E})$, we associate a binary variable y_i with each vertex i, so $\boldsymbol{y}=\{y_1,\ldots,y_{|\mathcal{V}|}\}$. In the image labeling setting, for example, there would be one binary variable per pixel. In this case, the sufficient statistic $\rho(\boldsymbol{y})$ is no longer the identity. Instead, it is

$$\rho(\boldsymbol{y}) = (\{\mathbf{1}_{\{\boldsymbol{y}_i=k\}}|i\in\mathcal{V},k\in\{0,1\}\}, \tag{15}$$

$$\{\mathbf{1}_{\{\boldsymbol{y}_i\neq\boldsymbol{y}_j\}}|(i,j)\in\mathcal{E}\}). \tag{16}$$

There is an indicator for each y_i, and indicators denoting whether variables that share an edge in G have different values (i.e., the edge is cut). The vector $\boldsymbol{w}$ associates a cost with each variable taking on each value, and for cutting each edge. The constraints $\eta(\boldsymbol{y})$ are unused, because all binary labelings are allowed.

Assuming the $\boldsymbol{w}$ values associated with edges are nonnegative, the resulting $f_{\boldsymbol{w}}(\boldsymbol{y})=\langle\boldsymbol{w},\rho(\boldsymbol{y})\rangle$ will be submodular and can thus be minimized efficiently. Conversely, it is the case that any graph-structured submodular function over binary variables can be expressed in this form (Kolmogorov and Zabih, 2004).

8 Relation to Conditional Random Field Representation

In Papandreou and Yuille (2011), it is discussed how the Gibbs distribution arising from a typical CRF formulation can be recovered in the PM framework by adding independent Gumbel noise to the energy of each joint assignment. While the same argument applies to RandOMs, since it is not practically implementable, and since it has been discussed in Papandreou and Yuille (2011), we do not discuss it further here. Instead, we focus on how the *energy function* of

[2] One way to define a shortest *st*-path as a set of constraints: constrain the out-degree of s to be 1, the in-degree of t to be 1, and for all other vertices, force the in-degree to equal the out-degree, which must be less than or equal to 1. Note these are linear constraints in $\boldsymbol{y}$.

a standard CRF formulation can be recovered using our notation and the RandOM formulation.

The latent variables $\boldsymbol{w}$ are defined such that the energy function used by a standard exponential family form of CRF could be recovered as $f_{\boldsymbol{w}}(\boldsymbol{y})$. In our notation, we could achieve this by deterministically setting $\boldsymbol{w} = \boldsymbol{\psi}^T \boldsymbol{\phi}(\boldsymbol{x})$. To illustrate this representational choice concretely, consider a pairwise graphical model over graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with unary feature set $\mathcal{U}$ and pairwise feature set $\mathcal{P}$, and thus energy function

$$E(\boldsymbol{y}) = \sum_{d \in \mathcal{V}} \sum_{u \in \mathcal{U}} \psi_u \bar{\phi}_{ud}(y_d; \boldsymbol{x}) \tag{17}$$

$$+ \sum_{d,d' \in \mathcal{E}} \sum_{p \in \mathcal{P}} \psi_p \bar{\phi}_{pdd'}(y_d, y_{d'}; \boldsymbol{x}), \tag{18}$$

where $\bar{\phi}_{ud}$ are unary feature functions that give a feature response for feature u at location d for each possible setting of y_d. When $\boldsymbol{y}$ are discrete, the feature functions $\bar{\phi}$ decompose as

$$\bar{\phi}_{ud}(y_d; \boldsymbol{x}) = \sum_{k \in L_d} \mathbf{1}_{\{y_d = k\}} \phi_{udk}(\boldsymbol{x}) \tag{19}$$

$$\bar{\phi}_{pdd'}(y_d, y_{d'}; \boldsymbol{x}) = \sum_{k,k' \in L_d \times L_{d'}} \mathbf{1}_{\{y_d = k \wedge y_{d'} = k'\}} \phi_{pdd'kk'}(\boldsymbol{x}) \tag{20}$$

We can then flatten the above ϕ functions into a single vector, $\boldsymbol{\phi}$, such that $E(\boldsymbol{y}) = \left\langle \boldsymbol{\psi}^T \boldsymbol{\phi}(\boldsymbol{x}), \rho(\boldsymbol{y}) \right\rangle$, where $\rho(\boldsymbol{y})$ is the set of standard exponential family sufficient statistics for the canonical overcomplete representation: $\rho(\boldsymbol{y}) = \left(\mathbf{1}_{\{\boldsymbol{y}_\alpha = k_\alpha\}}\right)_{\alpha \in \mathcal{V} \cup \mathcal{E}, k_\alpha \in \times_{d \in \alpha} L_d}$, where k_α is either the label space for a single variable, or the cross product of label spaces for variables that share an edge (Wainwright and Jordan, 2008). From here, it is straightforward to see that if we set $\boldsymbol{w}$ to be deterministically defined as $\boldsymbol{w} = \boldsymbol{\psi}^T \boldsymbol{\phi}(\boldsymbol{x})$, then $f_{\boldsymbol{w}}(\boldsymbol{y}) = E(\boldsymbol{y})$. In the RandOM formulation, $\boldsymbol{w}$ is instead a random function of $\boldsymbol{\psi}^T \boldsymbol{\phi}(\boldsymbol{x})$.

9 Geometry of Inverse Mapping Sets

A fundamental challenge when using real-valued parameters to define cost functions over discrete spaces is that there are many settings of the parameters that lead to the same minimum cost assignment. We refer to this set of parameter settings as the "inverse mapping set," and — as mentioned before — we denote it $F^{-1}(\boldsymbol{y})$. One of the central themes of this work is that optimization algorithms can be productively thought of in terms of the geometry of the inverse mapping set $F^{-1}(\boldsymbol{y})$, and leveraging structure in $F^{-1}(\boldsymbol{y})$ can lead to efficiencies in learning algorithms.

In this section, we develop some intuitions about these important sets and prove properties of $F^{-1}(\boldsymbol{y})$ for different optimization problems defined by $f_{\boldsymbol{w}}$. In the next section, we will discuss learning algorithms that make use of these characterizations.

Proposition 3. *When $f_{\boldsymbol{w}}(\boldsymbol{y})$ is defined as the exponential family (possibly with combinatorial base measure) as in Eq. 4, $F^{-1}(\boldsymbol{y})$ is a convex set.*

Proof. First, consider the case that $\eta(\boldsymbol{y}) = 0$ for all $\boldsymbol{y}$. Then, as noted in Papandreou and Yuille (2011), $F^{-1}(\boldsymbol{y})$ is defined by the conjunction of constraints $\langle \boldsymbol{w}, \rho(\boldsymbol{y}) \rangle < \langle \boldsymbol{w}, \rho(\boldsymbol{y'}) \rangle$ for all $\boldsymbol{y'} \neq \boldsymbol{y}$. This is a set of linear constraints in $\boldsymbol{w}$. Half-spaces are convex sets, and the intersection of a set of convex sets is a convex set, which proves the $\eta(\boldsymbol{y}) = 0$ case.

For the more general case where $\eta(\boldsymbol{y})$ may define nontrivial support, $F^{-1}(\boldsymbol{y})$ is defined by the conjunction of constraints $\langle \boldsymbol{w}, \rho(\boldsymbol{y}) \rangle < \langle \boldsymbol{w}, \rho(\boldsymbol{y'}) \rangle$ for all $\boldsymbol{y'} \neq \boldsymbol{y}$, *where $\boldsymbol{y'}$ is allowed by η*. Again, though, this is an intersection of half-spaces, which completes the proof. □

Proposition 4. *When $f_{\boldsymbol{w}}$ is defined as the connected components objective in Eq. 9, $F^{-1}(\boldsymbol{y})$ may be non-convex.*

Proof. We prove this by example, constructing $\boldsymbol{w}^A \in F^{-1}(\boldsymbol{y})$, $\boldsymbol{w}^B \in F^{-1}(\boldsymbol{y})$ and $\boldsymbol{w}^c = \lambda \boldsymbol{w}^A + (1 - \lambda)\boldsymbol{w}^B$ such that $\lambda \in [0, 1]$ and $\boldsymbol{w}^C \notin F^{-1}(\boldsymbol{y})$.

Suppose we have $G = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V} = \{v_1, v_2, v_3, v_4\}$ and edges $\mathcal{E} = \{(1, 2), (2, 3), (3, 4), (4, 1)\}$. Let $\tau > .5$. It is clear that if we set $\boldsymbol{w}^A = (w^A_{12}, w^A_{23}, w^A_{34}, w^A_{41}) = (1, 1, 1, 0)$, then there is a single connected component, so $F(\boldsymbol{w}^A) = (1, 1, 1, 1)$. Similarly, if we set $\boldsymbol{w}^B = (w^B_{12}, w^B_{23}, w^B_{34}, w^B_{41}) = (1, 0, 1, 1)$, then there is a single connected component, so also $F(\boldsymbol{w}^B) = (1, 1, 1, 1)$. Thus for $\boldsymbol{y} = (1, 1, 1, 1)$, $\boldsymbol{w}^A \in F^{-1}(\boldsymbol{y})$ and $\boldsymbol{w}^B \in F^{-1}(\boldsymbol{y})$.

Now let $\lambda = .5$, so $\boldsymbol{w}^c = \lambda \boldsymbol{w}^A + (1 - \lambda)\boldsymbol{w}^B = (1, .5, 1, .5)$. Now, however, both of the edges $(2, 3)$ and $(4, 1)$ have weight below the threshold i.e. $w_{ij} \leq \tau$, leaving us with two separate connected components. So $F(\boldsymbol{w}^C) \neq \boldsymbol{y}$ and thus $\boldsymbol{w}^C \notin F^{-1}(\boldsymbol{y})$, which completes the proof. □

Lemma 2. *Let $f_{\boldsymbol{w}}(\boldsymbol{y})$ be defined as the connected components objective in Eq. 9. There is no equivalent expression of $f_{\boldsymbol{w}}(\boldsymbol{y})$ in the exponential family form Eq. 4.*

Proof. This follows simply from a proof by contradiction that uses Propositions 3 and 4.

Suppose for the sake of contradiction that there is some $\tilde{f}_{\boldsymbol{w}}(\boldsymbol{y})$ such that $\tilde{f}_{\boldsymbol{w}}(\boldsymbol{y})$ is expressible in exponential family form, and that $\tilde{f}_{\boldsymbol{w}}(\boldsymbol{y}) = f_{\boldsymbol{w}}(\boldsymbol{y})$ for all $\boldsymbol{y}$. Then $F^{-1}(\boldsymbol{y}) = \tilde{F}^{-1}(\boldsymbol{y})$, and by Proposition 3, $F^{-1}(\boldsymbol{y})$ is a convex set. However, this contradicts Proposition 4. □

Note that this result is specific to the choice of parameterization for a problem. For example, there are functions $f_{\tilde{\boldsymbol{w}}}$ of higher dimensional $\tilde{\boldsymbol{w}}$ that assign the same cost as $f_{\boldsymbol{w}}$ to all $\boldsymbol{y}$, where the inverse mapping in the higher dimensional space could very well be onto a convex set.

9.1 Star Convexity

Our final result in this section is to show that there are properties of $F^{-1}(\boldsymbol{y})$ beyond convexity that will still be useful in certain later learning formulations. To illustrate this, we take as example the inverse set $F^{-1}(\boldsymbol{y})$ for the connected components problem from Section 2.1.2. While not necessarily a convex set, $F^{-1}(\boldsymbol{y})$ for this problem still has particular tractable structure that will allow us to learn a RandOM using some of the techniques in Section 4.

We begin by recalling the definition of *star convexity* (Smith, 1968).

Definition 1. *A set $\mathcal{S}$ is* star convex *if there exists a point $\boldsymbol{t} \in \mathcal{S}$ such that $\forall \boldsymbol{s} \in \mathcal{S}$, $\lambda \boldsymbol{s} + (1-\lambda)\boldsymbol{t} \in \mathcal{S}$ for all $\lambda \in [0,1]$. We call $\boldsymbol{t}$ a center point.*

Proposition 5. *The inverse set $F^{-1}(\boldsymbol{y})$ for the connected components problem from Section 2.1.2 is a star-convex set.*

Proof. The main idea is that for any $\boldsymbol{s} \in F^{-1}(\boldsymbol{y})$ and all pairs of variables i, j such that $y_i = y_j$, there must be at least one critical path between i and j using edges (k,l) such that $s_{kl} > \tau$. We define a center point $\boldsymbol{t}$ and show that moving from $\boldsymbol{s}$ towards $\boldsymbol{t}$ does not alter the critical path structure.

First, given $\boldsymbol{y}$, we give a center point $\boldsymbol{t}$. For each $(i,j) \in \mathcal{E}$, let $t_{ij} = 1$ if $y_i = y_j$ and $t_{ij} = 0$ if $y_i \neq y_j$. We then claim that $\boldsymbol{t}$ is a center point. To verify that $\boldsymbol{t}$ is a center point and thus that $F^{-1}(\boldsymbol{y})$ is star-convex, we need to show that for an arbitrary point $\boldsymbol{s} \in F^{-1}(\boldsymbol{y})$, it holds that $\lambda \boldsymbol{s} + (1-\lambda)\boldsymbol{t} = \boldsymbol{u} \in F^{-1}(\boldsymbol{y})$ for all $\lambda \in [0,1]$.

Next, we show that for any edge (i,j), we can independently change the value of u_{ij} to any value between s_{ij} and t_{ij} while maintaining that $F(\boldsymbol{u}) = \boldsymbol{y}^{\boldsymbol{u}}$ is equal to $\boldsymbol{y}$, which ensures $\boldsymbol{u} \in F^{-1}(\boldsymbol{y})$. A subtle point is that we are proving a slightly stronger condition than is required. Whereas star-convexity requires only that the line segment connecting $\boldsymbol{s}$ and $\boldsymbol{t}$ lies fully within $F^{-1}(\boldsymbol{y})$, we show that the largest axis-aligned hyperrectangle that has $\boldsymbol{s}$ and $\boldsymbol{t}$ as corners lies fully within $F^{-1}(\boldsymbol{y})$.

Let $r(\boldsymbol{w})$ be the set of edges $(i,j) \in \mathcal{E}$ such that $w_{ij} > \tau$. A consequence of this definition of r and the definition of $\boldsymbol{t}$ is that $y_i = y_j$ if and only if $(i,j) \in r(\boldsymbol{t})$. The next claim is that $r(\boldsymbol{t}) \supseteq r(\boldsymbol{s})$. Suppose this were false. Then $s_{ij} > \tau$ for some (i,j) where $y_i \neq y_j$, which contradicts $\boldsymbol{s} \in F^{-1}(\boldsymbol{y})$.

Consider a pair (i,j) such that $y_i = y_j$. Although it might be that $(i,j) \notin r(\boldsymbol{s})$ or even that $(i,j) \notin \mathcal{E}$, we know due to $\boldsymbol{s} \in F^{-1}(\boldsymbol{y})$ that there is some other path between i and j via edges in $r(\boldsymbol{s})$. We call such a path $p(i,j)$ a *critical path* between i and j. Note that all edges (k,l) on $p(i,j)$ have $t_{kl} = 1$, so letting $u_{kl} = \lambda_{kl} s_{kl} + (1-\lambda_{kl}) t_{kl}$ for $\lambda_{kl} \in [0,1]$ ensures that all critical paths in $r(\boldsymbol{s})$ are also in $r(\boldsymbol{u})$. This implies that after setting $u_{kl} = \lambda_{kl} s_{kl} + (1-\lambda_{kl}) t_{kl}$, if $y_k = y_l$, then $y_k^{\boldsymbol{u}} = y_l^{\boldsymbol{u}}$.

The final step is to show that by setting $u_{kl} = \lambda s_{kl} + (1-\lambda_{kl}) t_{kl}$, we never induce $y_i^{\boldsymbol{u}} = y_j^{\boldsymbol{u}}$ when $y_i \neq y_j$. We make use of the fact that $y_i^{\boldsymbol{u}} = y_j^{\boldsymbol{u}}$ when $y_i \neq y_j$ for some i and j if and only if there is at least one edge $(k,l) \in \mathcal{E}$ such that $(k,l) \in r(\boldsymbol{u})$ while $(k,l) \notin r(\boldsymbol{t})$. Consider a $(k,l) \in \mathcal{E}$ such that $y_k \neq y_l$. From here, it follows that $t_{kl} = 0$ and $s_{kl} \leq \tau$. Letting $u_{kl} = \lambda_{kl} s_{kl} + (1-\lambda_{kl}) t_{kl}$ for $\lambda_{kl} \in [0,1]$, it is clear that $u_{kl} \leq \tau$. This implies that $(k,l) \notin r(\boldsymbol{t}) \implies (k,l) \notin r(\boldsymbol{u})$, which completes the proof. □

10 Slice Sampling Algorithm

Algorithm 2 Inner Slice Sampling Loop for Bipartite Matching RandOM

```
Input: $\mathcal{A}_{\boldsymbol{y}}$    Current state of dynamic Hungarian algorithm, which stores $\boldsymbol{y}$ internally
Input: $\boldsymbol{w} \in \mathbb{R}^{J^2}$,   which lies within $F^{-1}(\boldsymbol{y})$
Input: $\alpha \in \mathbb{R}$    Slice sampling step out parameter
$\boldsymbol{W} \leftarrow \textsc{Reshape}(\boldsymbol{w}, (J, J))$ { Treat $\boldsymbol{w}$ as a $J \times J$ matrix}
for $j = 1$ to $J$ do
    $u' \leftarrow \log(\textsc{Random-Uniform}(0, 1)) + \log p(\boldsymbol{W} \mid \boldsymbol{\psi}, \boldsymbol{x})$
    $\boldsymbol{s} \leftarrow \textsc{Random-Normal}(\boldsymbol{0}, \mathbb{I}_J)$
    {Step out}
    $inInvSetL, inInvSetR, inInvSet \leftarrow 0, 0, 0$ { Used to cache set membership calls}
    $\boldsymbol{W}_{j:} \leftarrow \boldsymbol{W}_{j:} - \alpha \boldsymbol{s}$ { Subtract from row $j$ of $\boldsymbol{W}$}
    $b_l, b_r \leftarrow -\alpha, \alpha$ {Keep track of how far we step out left and right, respectively}
    while $\log p(\boldsymbol{W} \mid \boldsymbol{\psi}, \boldsymbol{x}) > u' \wedge (inInvSetL \leftarrow \textsc{In-Inverse-Set}(\boldsymbol{W}, j; \mathcal{A}_{\boldsymbol{y}}))$ do
        $\boldsymbol{W}_{j:} \leftarrow \boldsymbol{W}j : -\alpha \boldsymbol{s}$
        $b_l \leftarrow b_l - \alpha$
    end while
    $\boldsymbol{W}_{j:} \leftarrow \boldsymbol{W}_{j:} - b_l \boldsymbol{s} + \alpha \boldsymbol{s}$ { Return to starting point, and step right}
    while $\log p(\boldsymbol{W} \mid \boldsymbol{\psi}, \boldsymbol{x}) > u' \wedge (inInvSetR \leftarrow \textsc{In-Inverse-Set}(\boldsymbol{W}, j; \mathcal{A}_{\boldsymbol{y}}))$ do
        $\boldsymbol{W}_{j:} \leftarrow \boldsymbol{W}j : +\alpha \boldsymbol{s}$
        $b_r \leftarrow b_r + \alpha$
    end while
    {Step in}
    $b \leftarrow b_r$ { Current state of $\boldsymbol{W}$ is where we ended step out right}
    while $!(inInvSet \wedge \log p(\boldsymbol{W} \mid \boldsymbol{\psi}, \boldsymbol{x}) > u')$ do
        $b' \leftarrow \textsc{Random-Uniform}(b_l, b_r)$
        $\boldsymbol{W}_{j:} \leftarrow \boldsymbol{W}_{j:} + (b' - b)\boldsymbol{s}$
        if $(b' < 0 \wedge inInvSetL) \vee (b' > 0 \wedge inInvSetR)$ then
            $inInvSet \leftarrow 1$
        else
            $inInvSet \leftarrow \textsc{In-Inverse-Set}(\boldsymbol{W}, j; \mathcal{A}_{\boldsymbol{y}})$
        end if
        if $b' < 0$ then
            $b_l \leftarrow b'$
            $inInvSetL \leftarrow inInvSet$
        else if $b' > 0$ then
            $b_r \leftarrow b'$
            $inInvSetR \leftarrow inInvSet$
        end if
    end while
end for
Return $\textsc{Reshape}(\boldsymbol{W}, (J^2))$
```