Randomized Optimum Models for Structured Prediction — 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

[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 $\boldsymbol{y}$.

here. Instead, we focus on how the *energy function* of 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 nonconvex.*

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 hyper-rectangle 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
$\boldsymbol{W} \leftarrow$ RESHAPE$(\boldsymbol{w}, (J, J))$ { Treat $\boldsymbol{w}$ as a $J \times J$ matrix}
for $j = 1$ to $J$ do
  $u' \leftarrow \log(\text{RANDOM-UNIFORM}(0,1)) + \log p(\boldsymbol{W} \mid \boldsymbol{\psi}, \boldsymbol{x})$
  $\boldsymbol{s} \leftarrow \text{RANDOM-NORMAL}(\boldsymbol{0}, \mathbb{I}_J)$
  {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 $\log p(\boldsymbol{W} \mid \boldsymbol{\psi}, \boldsymbol{x}) > u' \wedge (inInvSetL \leftarrow \text{IN-INVERSE-SET}(\boldsymbol{W}, j; \mathcal{A}_{\boldsymbol{y}}))$ 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 $\log p(\boldsymbol{W} \mid \boldsymbol{\psi}, \boldsymbol{x}) > u' \wedge (inInvSetR \leftarrow \text{IN-INVERSE-SET}(\boldsymbol{W}, j; \mathcal{A}_{\boldsymbol{y}}))$ 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
    $b' \leftarrow \text{RANDOM-UNIFORM}(b_l, b_r)$
    $\boldsymbol{W}_{j:} \leftarrow \boldsymbol{W}_{j:} + (b' - b)\boldsymbol{s}$
    if $(b' < 0 \wedge inInvSetL) \vee (b' > 0 \wedge inInvSetR)$ then
      $inInvSet \leftarrow 1$
    else
      $inInvSet \leftarrow \text{IN-INVERSE-SET}(\boldsymbol{W}, j; \mathcal{A}_{\boldsymbol{y}})$
    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 RESHAPE$(\boldsymbol{W}, (J^2))$
```