XOR-CD: Linearly Convergent Constrained Structure Generation

Fan Ding [1] Jianzhu Ma [2] Jinbo Xu [3] Yexiang Xue [1]

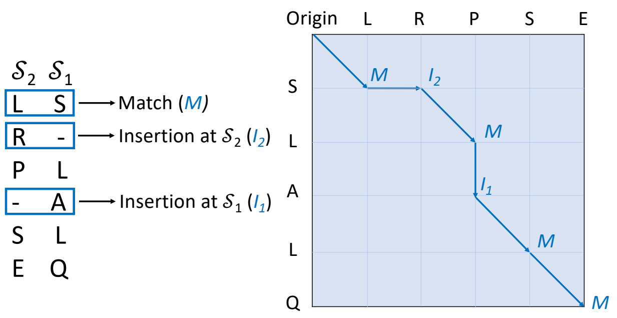
Abstract

We propose XOR-Contrastive Divergence learning (XOR-CD), a provable approach for constrained structure generation, which remains difficult for state-of-the-art neural network and constraint reasoning approaches. XOR-CD harnesses XOR-Sampling to generate samples from the model distribution in CD learning and is guaranteed to generate valid structures. In addition, XOR-CD has a linear convergence rate towards the global maximum of the likelihood function within a vanishing constant in learning exponential family models. Constraint satisfaction enabled by XOR-CD also boosts its learning performance. Our real-world experiments on data-driven experimental design, dispatching route generation, and sequence-based protein homology detection demonstrate the superior performance of XOR-CD compared to baseline approaches in generating valid structures as well as capturing the inductive bias in the training set.

1. Introduction

Generative modeling has received tremendous success in recent years. Notable examples include image synthesis (Goodfellow et al., 2014; Radford et al., 2015; Isola et al., 2017; Brock et al., 2018), music composition (Briot et al., 2017; Engel et al., 2017; Prenger et al., 2019), molecule synthesis, drug discovery (Liu et al., 2018; Kusner et al., 2017; Jin et al., 2018) and more.

Nevertheless, learning generative models over a constrained space still remains a major research challenge. Take the example of protein homology detection, an important biological application considered in this paper, to align two protein sequences, where each amino acid in one sequence can be aligned either to another amino acid in the other



$\mathcal{S}_2$ $\mathcal{S}_1$
L S
R -
P L
- A
S L
E Q
Match (M)
Insertion at $\mathcal{S}_2$ (I_2)
Insertion at $\mathcal{S}_1$ (I_1)
Origin
L
R
P
S
E
S
L
A
L
Q
M
I_2
M
I_1
M
M

Figure 1. An illustration of sequence-based protein homology detection problem. **(Left)** The problem is to predict the alignment two amino acids sequences $\mathcal{S}_1$ and $\mathcal{S}_2$, where one amino acid from one sequence can be aligned to either one amino acid from the other sequence (match), or to a gap (insertion, marked by $-$). **(Right)** Biological constraint: such an alignment must form a path from the top-left to the bottom-right corner in the alignment matrix, where a diagonal transition represents a match, a horizontal or a vertical transition represents an insertion.

sequence or a gap. **Biological constraints** require such an alignment forms a continuous path from the top-left corner to the bottom-right corner in the alignment matrix (see Figure 1). Given two protein sequences, the **learning problem** is to predict which alignment is more likely based on samples from the training set. Previous constrained satisfaction approaches (Rychlewski et al., 2000) identify an alignment satisfying all constraints, yet are optimal only regarding a rigid expert-defined objective, which may fail to include essential biological factors. Learning-based approaches (Söding, 2005; Ma et al., 2014) harness highly flexible neural models for alignment prediction. However, the predictions often violate constraints (i.e., they do not form paths). Similar challenges are present in many real-world problems: it is difficult to generate structures which simultaneously (i) satisfy constraints, and (ii) effectively capture the inductive bias present in the training set.

We present XOR-CD, a constrained generative model based on contrastive divergence and XOR-Sampling, which converges to the **global optimum** of the likelihood function of an exponential family model within a vanishing constant in linear number of steps. Rather than using Markov Chain Monte Carlo (MCMC) to sample from the current model distribution as in the classical case of contrastive divergence, XOR-CD generates samples using XOR-Sampling, a recent

[1]Department of Computer Science, Purdue University, West Lafayette, USA [2]Institute for Artificial Intelligence, Peking University, Beijing, China [3]Toyota Technological Institute at Chicago, Illinois, USA. Correspondence to: Fan Ding <ding274@purdue.edu>, Yexiang Xue <yexiang@purdue.edu>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

approach with provable guarantees. XOR-Sampling reduces the sampling problem into queries to NP oracles subject to randomized XOR constraints. The empirical probability of getting one sample can be bounded within a constant multiplicative factor of the true probability. Our main contribution is to embed XOR sampling into contrastive divergence learning, which yields a learning approach with provable guarantees. XOR-CD advances the state-of-the-art in constrained structure generation in the following way:

1. **Constraint Satisfaction**: Because XOR-CD reduces the sample generation problem into constraint satisfaction subject to randomized constraints, the structures generated from XOR-CD are guaranteed to satisfy constraints, addressing a key limitation of many neural network based structure generation approaches.
2. **Linear Convergence Speed to the Global Optimum**: We are able to prove that XOR-CD has a linear convergence rate towards the *global* maximum of the likelihood function within a vanishing constant when learning exponential family models.
3. **Constraint Satisfaction Improves Learning Performance**: We observe empirically that XOR-CD learns faster and better than state-of-the-art neural-based approaches in constrained generation domains. We hypothesize that the improvement in learning performance is due to better constraint satisfaction offered by XOR-CD. Because the samples generated from the model distribution always satisfy constraints, XOR-CD can focus on learning the *structural differences* of the samples generated from the model distribution and from data. Baseline approaches, contrarily, spend most of their time struggling in generating valid structures.

We demonstrate the power of XOR-CD on **three real-world applications**. Aside from the protein homology detection problem, our second application is on the optimal experiment design, which tests n crops in a n-by-n field. Agriculturalists require very crop to be planted exactly once in every row and column, forming a so-called Latin square. Yet, other implicit criteria can only be learned from a dataset of good designs, therefore making it a learning problem. Our third application is on dispatching route generation, in which we suggest routes for delivery drivers. The routes have to form a Hamilton cycle, visiting each requested location once and only once. Aside from this hard requirement, they also need to be similar to historical routes, satisfying drivers' implicit preferences. In all 3 applications, our method generates structures that not only have higher likelihood than competing approaches, but also 100% satisfy constraints, while the validity rate of competing approaches are less than 20%. In addition, the distributions of the valid structures generated by XOR-CD closely resemble those in the training set, demonstrating that XOR-CD can successfully capture the inductive bias in the training set. Furthermore, the learned XOR-CD model can be used to complete partially filled structures. These completed structures 100% satisfy constraints and are close to those in the training set.

2. Preliminaries

2.1. Exponential Family Models

We consider discrete exponential family models over random variables $X \in \mathcal{X} \subseteq \{0,1\}^n$ with parameters $\theta \in \mathbb{R}^d$:

$$P_\theta(X) = c(X)e^{\theta^T\phi(X)-\Lambda(\theta)}, \tag{1}$$

where $c(X)$ is the carrier measure, $\phi : \mathcal{X} \to \mathbb{R}^d$ is the sufficient statistics and $\Lambda(\theta)$ is the log partition function:

$$\Lambda(\theta) = \log \sum_{x\in\mathcal{X}} c(x)e^{\theta^T\phi(x)}. \tag{2}$$

Notice that $\Lambda(\theta)$ contains a discrete integral over a constrained structure space $\mathcal{X}$, which makes the entire problem computationally intractable. For example, in the protein homology detection application, $\mathcal{X}$ represents the space of all alignments that form valid paths. Given $x_1, x_2, \ldots, x_N$ from the training set, the **learning problem** is to find the optimal parameters θ which maximize the log likelihood $l(\theta) = \frac{1}{N}\sum_{i=1}^N \log P_\theta(x_i)$:

$$\max_\theta \; l(\theta) = \frac{1}{N}\sum_{i=1}^N \theta^T\phi(x_i) - \Lambda(\theta) + \text{constant}. \tag{3}$$

Given the learned model, the **inference problem** is to complete a partial structure to maximize the joint probability:

$$(x_p, x^*) = \arg\max_{(x_p,x)\in\mathcal{X}} P_\theta(x_p, x).$$

Here, x_p represents a partially filled structure, and x^* are the assignment to the remaining variables. Back to the protein homology detection example, the learning problem is to identify which types of alignments are more likely in the training dataset, while the inference problem is to predict an alignment given the amino acid sequences of two proteins. $\Lambda(\theta)$ is a convex function of θ. Denote $\nabla\Lambda(\theta)$ the gradient vector of $\Lambda(\theta)$. We can prove $\nabla\Lambda(\theta)$ is the expectation of the sufficient statistics $\phi(x)$ under P_θ. That is,

$$\nabla\Lambda(\theta) = \mathbb{E}_{P_\theta}[\phi(x)] = \sum_{x\in\mathcal{X}} \phi(x)P_\theta(x). \tag{4}$$

2.2. Contrastive Divergence Learning

Contrastive Divergence (CD) (Hinton, 2002b) applies stochastic gradient ascent to maximize the log likelihood of an exponential family model. From Equations 3 and 4, the gradient of the log likelihood can be written as:

$$\nabla l(\theta) = \mathbb{E}_D[\phi(X)] - \mathbb{E}_{P_\theta}[\phi(X)]. \quad (5)$$

Here, $\mathbb{E}_D$ denotes the expectation over the data distribution, and $\mathbb{E}_{P_\theta}$ denotes the expectation over the current model distribution P_θ. Let $\{x_1, \dots, x_M\}$ be a mini-batch of training data, CD uses the sample mean $\frac{1}{M}\sum_{i=1}^M \phi(x_i)$ to approximate $\mathbb{E}_D[\phi(X)]$. Denote $x_i^{(k)}$ as the sample after taking k Markov Chain Monte-Carlo (MCMC) steps following the current model distribution $P_\theta(x)$ starting from data x_i. CD uses $\frac{1}{M}\sum_{i=1}^M \phi(x_i^{(k)})$ to approximate $\mathbb{E}_{P_\theta}[\phi(X)]$. Overall, the gradient of the log likelihood is approximated by

$$g_{cd}(\theta) \approx \frac{1}{M}\sum_{i=1}^M \phi(x_i) - \frac{1}{M}\sum_{i=1}^M \phi(x_i^{(k)}). \quad (6)$$

CD hence iterates the following update $\theta_{t+1} = \theta_t + \eta g_{cd}(\theta_t)$ until convergence, where η is the learning rate.

2.3. XOR Sampling

Our method leverages recent advancements in XOR sampling (Ermon et al., 2013b), which reduces the sampling problem into queries to NP oracles subject to XOR constraints. XOR sampling guarantees that the probability of drawing a sample is sandwiched between a multiplicative constant of the true probability. We only present the general idea of XOR-Sampling on unweighted functions here and refer the readers to the paper (Ermon et al., 2013b) for the weighted case. For the unweighted case, assuming $w(x)$ takes binary values, we need to draw samples from the set $\mathcal{W} = \{x : w(x) = 1\}$ uniformly at random; i.e., suppose $|\mathcal{W}| = 2^l$, then each member in $\mathcal{W}$ should have 2^{-l} probability to be sampled. XOR proceeds by adding k randomized XOR constraints $XOR_k(x) = 1$ to the original problem and returns an element uniformly at random from the constrained set $\mathcal{W}_k = \{x : w(x) = 1, XOR_k(x) = 1\}$ when $|\mathcal{W}_k|$ is small enough and can be sampled by an exact sampler. k is increased from 1 until $|\mathcal{W}_k|$ becomes small. Because the k-th XOR constraint removes at random half of the elements from the previous set $\mathcal{W}_{k-1}$, one can prove a constant bound on the probabilities of getting one sample from XOR sampling (Gomes et al., 2007a;b).

For the weighted case, one needs to draw samples from an unnormalized function $w(x)$, i.e., the probability of getting a sample x_0, $P(x_0)$ is proportional to $w(x)$. The idea is to discretize $w(x)$ and transform the weighted problem into an unweighted one with additional variables. Our paper uses the constant approximation bounds of XOR sampling on weighted functions through the following theorem. The details on the discretization scheme and the choice of the parameters of the original algorithm to reach the bound in Theorem 1 are in the supplementary materials.

Theorem 1. *(Ermon et al., 2013b) Let $1 < \delta \leq \sqrt{2}$, $0 < \gamma < 1$, $w : \{0,1\}^n \to \mathbb{R}^+$ be an unnormalized probability density function. $P(x) \propto w(x)$ is the normalized distribution. Then, with probability at least $1-\gamma$, XOR-Sampling(w, δ, γ) succeeds and outputs a sample x_0 by querying $O(n \ln(\frac{n}{\gamma}))$ NP oracles. Upon success, each x_0 is produced with probability $P'(x_0)$. We must have*

$$1/\delta P(x_0) \leq P'(x_0) \leq \delta P(x_0).$$

Moreover, let $\phi : \{0,1\}^n \to \mathbb{R}^+$ be one non-negative function, then the expectation of one sampled $\phi(x)$ satisfies,

$$\frac{1}{\delta}\mathbb{E}_{P(x)}[\phi(x)] \leq \mathbb{E}_{P'(x)}[\phi(x)] \leq \delta \mathbb{E}_{P(x)}[\phi(x)]. \quad (7)$$

3. XOR-Contrastive Divergence

We propose XOR-CD, a new contrastive divergence method for constrained structure generation on exponential family models, which is guaranteed to converge to the global maximum of the likelihood function within a vanishing constant in linear number of CD iterations. XOR-CD breaks down the gradient of the log likelihood function into the divergence of the expectations of the sufficient statistics over the training data and over the current model distribution, following the CD framework. However, XOR-CD leverages XOR-sampling to generate samples in estimating $\mathbb{E}_{P_\theta}[\phi(X)]$.

The detailed procedure of XOR-CD is shown in Algorithm 1. XOR-CD takes the exponential family model $P_\theta(X)$ with sufficient statistics $\phi(X)$, carrier measure $c(X)$, training data $\{x_i\}_{i=1}^N$, initial model parameter θ_0, the learning rate η, the number of CD iterations T, XOR-Sampling parameters (δ, γ), and batch sizes M, K as input, and outputs the learned parameter $\overline{\theta_T}$. To approximate $\mathbb{E}_{P_{\theta_t}}[\phi(X)]$ at step t, XOR-CD draws K samples $x'_1, \dots, x'_K$ from $P_{\theta_t}(X)$ using XOR-Sampling, where K is a user-determined sample size. Because XOR-Sampling has a failure rate, XOR-CD repeatedly call XOR-Sampling until all K samples are obtained successfully (line 2 – 6). Then, XOR-CD also draws M samples from the training set $\{x_i\}_{i=1}^N$ uniformly at random to approximate $\mathbb{E}_D[\phi(X)]$. Once all the samples are obtained, XOR-CD uses $\overline{g_t} = \frac{1}{M}\sum_{j=1}^M \phi(x_j) - \frac{1}{K}\sum_{j=1}^K \phi(x'_j)$ as an approximation for the gradient of the log likelihood. θ is updated following the rule $\theta_{t+1} = \theta_t + \eta\overline{g_t}$ for T steps, and η is the learning rate. Finally, the average $\overline{\theta_T} = \frac{1}{T}\sum_{t=1}^T \theta_t$ is the final output of the algorithm.

3.1. Linear Convergence to the Global Optimum

We can show that XOR-CD converges to the global optimum of the log likelihood function in addition to a vanishing term. Moreover, the speed of the convergence is linear with respect to the number of contrastive divergence steps.

Denote $Var_D(\phi(x)) = \mathbb{E}_D[||\phi(x)||_2^2] - ||\mathbb{E}_D[\phi(x)]||_2^2$ and $Var_{P_\theta}(\phi(x)) = \mathbb{E}_{P_\theta}[||\phi(x)||_2^2] - ||\mathbb{E}_{P_\theta}[\phi(x)]||_2^2$ as the total variations of $\phi(x)$ w.r.t. the data distribution P_D and model distribution P_θ. The precise mathematical theorem states:

Theorem 2. *(main) Let $P_\theta(X) : \mathcal{X} \to \mathbb{R}^+$ be the exponential family model denoted in Equation 1. Given data points $\{x_i\}_{i=1}^N$, the log likelihood $l(\theta) = \frac{1}{N}\sum_{i=1}^N \log P_\theta(x_i)$. Denote $OPT = \max_\theta l(\theta)$. Let $Var_D(\phi(x)) \leq \sigma_1^2$, $\max_\theta Var_{P_\theta}(\phi(x)) \leq \sigma_2^2$ and $||\mathbb{E}_{P_\theta}[\phi(x)]||_2^2 \leq \varepsilon^2$. Suppose $1 \leq \delta \leq \sqrt{2}$ is used in XOR-sampling, the learning rate $\eta \leq \frac{2-\delta^2}{\sigma_2^2\delta}$, and $\overline{\theta_T}$ is the output of XOR-CD. We have:*

$$OPT - \mathbb{E}[l(\overline{\theta_T})] \leq \frac{\delta||\theta_0 - \theta^*||_2^2}{2\eta T} + \frac{\eta(\sigma_2^2 + \varepsilon^2)}{K} + \frac{\eta\sigma_1^2}{\delta M}.$$

XOR-CD is the first provable algorithm which converges to the global maximum of the likelihood function and a tail term for exponential family models. Moreover, the rate of the convergence is linear in the number of SGD iterations T. Previous approaches do not have such tight bounds. Variational inference approaches such as the Variational Auto-encoders (VAEs) (Kingma & Welling, 2013) optimize the Evidence Lower Bound (ELBO). However, the gap between the lower bound and the true likelihood can become arbitrarily large. Expectation Propagation (EP) methods (Minka, 2013; Dehaene & Barthelmé, 2015) computes an upper bound of the likelihood, which can be arbitrarily loose as well. Various Generative Adversarial Nets (GANs) (Goodfellow et al., 2014; Radford et al., 2015; Isola et al., 2017) and flow models (Kingma & Dhariwal, 2018; Prenger et al., 2019) do not have theoretic bounds.

The main challenge to prove Theorem 2 lies in the fact that we cannot ensure the unbiasedness of the gradient. Because the partition function $\Lambda(\theta)$ is convex with respect to θ in exponential family models, a gradient descent algorithm can be proven to be linearly convergent towards the maximum of the likelihood function, if the expectation of the estimated gradient is unbiased, ie, $\mathbb{E}[\overline{g_t}] = \nabla l(\theta_t)$. However, even though we apply XOR-sampling, we still cannot guarantee the unbiasedness of $\overline{g_t}$. Instead, using Theorem 1, our bound for $\overline{g_t}$ is in the following form:

$$\frac{1}{\delta}[\nabla l(\theta_t)]^+ \leq \mathbb{E}[\overline{g_t}^+] \leq \delta[\nabla l(\theta_t)]^+, \tag{8}$$

$$\delta[\nabla l(\theta_t)]^- \leq \mathbb{E}[\overline{g_t}^-] \leq \frac{1}{\delta}[\nabla l(\theta_t)]^-. \tag{9}$$

Here, $[f]^+$ means the positive part of f, ie, $[f]^+ = \max\{f, \mathbf{0}\}$, and $[f]^-$ means the negative part of f, ie, $[f]^- = \min\{f, \mathbf{0}\}$. The bound in Equation 8 and 9 can be proven following the fact that $\nabla l(\theta) = \mathbb{E}_D[\phi(X)] - \mathbb{E}_{P_\theta}[\phi(X)]$ and applying Equation 7. The proof of Theorem 2 relies on our following new result (Theorem 3) on Stochastic Gradient Descent (SGD) algorithms which only have access to constant approximate gradient vectors. As far as we know, previous SGD convergence analysis largely requires the unbiasedness of the gradient. We are the first to extend SGD convergence bounds to biased cases. Theorem 3 requires function f to be L-smooth. $f(\theta)$ is L-smooth if and only if $||f(\theta_1) - f(\theta_2)||_2 \leq L||\theta_1 - \theta_2||_2$. Notice that the conditions of Theorem 2 automatically guarantee the L-smoothness of the log likelihood.

Algorithm 1 XOR-CD

```
Input: $\theta_0, c(X), \phi(X), T, \eta, \delta, \gamma, M, K, \{x_i\}_{i=1}^N$.
1 for $t = 0$ to $T$ do
2     $j \leftarrow 1$
      while $j \leq K$ do
3         $x' \leftarrow$ XOR-Sampling$\left(c(X)e^{\theta_t{}^T\phi(X)}, \delta, \gamma\right)$
          if $x' \neq Failure$ then
4             $x'_j \leftarrow x'; j \leftarrow j+1$
5         end
6     end
7     Sample $\{x_j\}_{j=1}^M$ uniformly from $\{x_i\}_{i=1}^N$.
      $\overline{g_t} = \frac{1}{M}\sum_{j=1}^M \phi(x_j) - \frac{1}{K}\sum_{j=1}^K \phi(x'_j)$
      $\theta_{t+1} = \theta_t + \eta\overline{g_t}$
8 end
9 return $\overline{\theta_T} = \frac{1}{T}\sum_{t=1}^T \theta_t$.
```

Theorem 3. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a L-smooth convex function and $\theta^* = argmin_\theta f(\theta)$. In iteration t of SGD, g_t is the estimated gradient, i.e., $\theta_{t+1} = \theta_t - \eta g_t$. If $Var(g_t) \leq \sigma^2$, and there exists $1 \leq c \leq \sqrt{2}$ s.t. $\frac{1}{c}[\nabla f(\theta_t)]^+ \leq \mathbb{E}[g_t^+] \leq c[\nabla f(\theta_t)]^+$ and $c[\nabla f(\theta_t)]^- \leq \mathbb{E}[g_t^-] \leq \frac{1}{c}[\nabla f(\theta_t)]^-$, then for any $T > 1$ and step size $\eta \leq \frac{2-c^2}{Lc}$, let $\overline{\theta_T} = \frac{1}{T}\sum_{t=1}^T \theta_t$, we have*

$$\mathbb{E}[f(\overline{\theta_T})] - f(\theta^*) \leq \frac{c||\theta_0 - \theta^*||_2^2}{2\eta T} + \frac{\eta\sigma^2}{c}. \tag{10}$$

The proofs of Theorems 2 and 3 are left to the supplementary materials. Here we outline the sketch to prove Theorem 3.

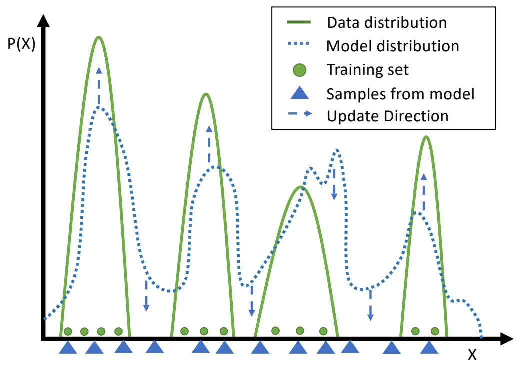
Proof. (sketch for Theorem 3) One can show under the conditions of Theorem 3, we must have (via Lemma 2, stated and proved in supplementary materials):

$$\frac{1}{c}||\mathbb{E}[g_t]||_2^2 \leq \langle\nabla f(\theta_t), \mathbb{E}[g_t]\rangle \leq c||\mathbb{E}[g_t]||_2^2.$$

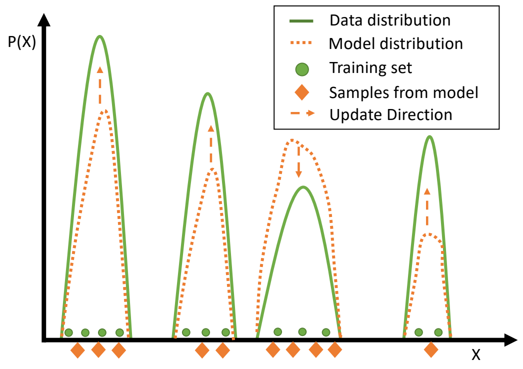
$$\frac{1}{c}\langle\mathbb{E}[g_t], \theta_t - \theta^*\rangle \leq \langle\nabla f(\theta_t), \theta_t - \theta^*\rangle \leq c\langle\mathbb{E}[g_t], \theta_t - \theta^*\rangle.$$

By the L-smoothness of f, for the t-th iteration,

$$f(\theta_{t+1}) \leq f(\theta_t) + \langle\nabla f(\theta_t), \theta_{t+1} - \theta_t\rangle + \frac{L}{2}||\theta_{t+1} - \theta_t||_2^2,$$



(a) Updating steps of traditional CD



(b) Updating steps of XOR-CD

Figure 2. An intuitive explanation of why constraint satisfaction enabled by XOR-CD improves the overall learning performance. In training data, valid structures are scattered across several isolated regions due to combinatorial constraints (green curves in both plots denote the training data distribution and green dots denote the training samples). Many negative samples generated by traditional CD do not satisfy constraints (blue triangles in the left plot). Therefore, traditional CD spends many iterations minimizing the likelihood of invalid structures (blue arrows in the left plot). XOR-CD converges faster to the ground-truth data distribution because all its updates are used to match the data distribution within the regions that satisfy constraints (orange arrows in the right plot).

$$= f(\theta_t) - \eta\langle\nabla f(\theta_t), g_t\rangle + \frac{Lt^2}{2}||g_t||^2. \quad (11)$$

By the convexity of f, we have

$$f(\theta_t) \leq f(\theta^*) + \langle\nabla f(\theta_t), \theta_t - \theta^*\rangle. \quad (12)$$

The following inequalities can be shown, combining Lemma 2, Equation 11 and 12:

$$\mathbb{E}[f(\theta_{t+1})] \leq f(\theta^*) + c\mathbb{E}\left[\langle g_t, \theta_t - \theta^*\rangle - \frac{\eta}{2}||g_t||_2^2\right] + \frac{\eta}{c}\sigma^2.$$

which implies $\mathbb{E}[f(\theta_{t+1})] - f(\theta^*) \leq \frac{c}{2\eta}\mathbb{E}[(||\theta_t - \theta^*||_2^2 - ||\theta_{t+1} - \theta^*||_2^2)] + \frac{\eta}{c}\sigma^2$. Sum this inequality for $t = 0, \ldots, T-1$, we get

$$\sum_{t=0}^{T-1}\mathbb{E}[f(\theta_{t+1}) - f(\theta^*)]$$
$$\leq \frac{c}{2\eta}(||\theta_0 - \theta^*||_2^2 - \mathbb{E}[||\theta_T - \theta^*||_2^2]) + \frac{T\eta}{c}\sigma^2$$
$$\leq \frac{c||\theta_0 - \theta^*||_2^2}{2\eta} + \frac{T\eta}{c}\sigma^2.$$

Finally, by Jensen's inequality, $tf(\overline{\theta_T}) \leq \sum_{t=1}^{T} f(\theta_t)$,

$$\sum_{t=0}^{T-1}\mathbb{E}[f(\theta_{t+1}) - f(\theta^*)] = \mathbb{E}[\sum_{t=1}^{T} f(\theta_t)] - Tf(\theta^*)$$
$$\geq T\mathbb{E}[f(\overline{\theta_T})] - Tf(\theta^*).$$

Combining the above equations we get

$$\mathbb{E}[f(\overline{\theta_T})] \leq f(\theta^*) + \frac{c||\theta_0 - \theta^*||_2^2}{2\eta T} + \frac{\eta}{c}\sigma^2.$$

This completes the proof. □

The proof of Theorem 2 is to apply Theorem 3 on the log likelihood function and noticing that $l(\theta)$ is L-smooth when the total variation $Var(\phi(x))$ is bounded (proved by a separate lemma). The lemmas and the proofs are left to section B.4 in supplementary materials. Theorem 2 states that in expectation, the difference between the output of XOR-CD algorithm $\overline{\theta_T}$ and the true optimum OPT is bounded by a term that is inversely proportional to the number of iterations T and a tail term $\frac{\eta(\sigma_2^2+\varepsilon^2)}{K} + \frac{\eta\sigma_1^2}{\delta M}$. To reduce the tail term with fixed steps η, we can generate more samples at each iteration to reduce the variance (increase M and K). In addition, to quantify the computational complexity of XOR-CD, we prove the following theorem in the supplementary materials detailing the number of queries to NP oracles needed for XOR-CD.

Theorem 4. *XOR-CD in Algorithm 1 uses $O(Tn\ln\frac{n}{\gamma} + TK)$ queries to NP oracles.*

3.2. Constraint Satisfaction Improves Learning

In the previous section we prove the theoretic convergence of XOR-CD towards the global optimum of the likelihood function. Despite XOR-CD has to query NP oracles, which are significantly more expensive than e.g., MCMC sampling, we notice that XOR-CD converges to the global optimum faster than classical CD approaches in real-world experiments, especially on constrained structure generation problems (see the experiment section). Notice that our observation is different from that of (Hinton, 2002a), where they observe CD works reasonably well even if the number of the MCMC steps k is kept far less than that required for well mixing. We attribute the observational difference to the types of problems we consider, which are mainly *con-*

strained structure generation problems. The capability to generate negative samples that satisfy constraints becomes important for likelihood learning in this setting.

We use Figure 2 as an intuitive explanation of why the constraint satisfaction enabled by XOR-CD leads to improvement in the learning performance. Figure 2(a) depicts one iteration of a traditional CD process. Here, CD tries to match the current model distribution (shown in the blue dashed line) with the data distribution (shown in the green line), by increasing the likelihood of the training samples and decreasing the likelihood of the negative samples generated typically by MCMC (denoted by the pulling of blue arrows). Because MCMC does not guarantee the constraint satisfaction of the negative samples, traditional CD spends much time pulling down the model likelihood in regions which violate constraints. On the contrary, Figure 2(b) depicts one iteration of XOR-CD. Because the negative samples are generated provably from XOR-sampling, they satisfy constraints. This allows XOR-CD to focus on matching the likelihood within the region that satisfy constraints; hence leading towards faster matching to the data distribution.

4. Related Work

There is a fruitful line of work for generative machine learning. Energy-based models (Hinton & Salakhutdinov, 2006; Bengio & Delalleau, 2009; Carreira-Perpinan & Hinton, 2005) take advantage of either exponential families (Hinton, 2002b; Jiang et al., 2018; Durkan et al., 2020) or neural networks (Belanger & McCallum, 2016; Belanger et al., 2017) for structure modelling. Qiu et al. (2019) leverages coupling of Markov chains to get unbiased samples in Contrastive Divergence framework, which however is hard to reach in practice. Score matching based methods (Bao et al., 2020; Song & Ermon, 2020; Pang et al., 2020) try to estimate the score function in order to get rid of the intractable partition function. Deep generative models like graph neural networks (Grover et al., 2019; Zhou et al., 2018) and normalizing flow models (Kingma & Dhariwal, 2018; Prenger et al., 2019) are widely used recently. Generative Adversarial Networks (GANs) (Goodfellow et al., 2014; Radford et al., 2015; Isola et al., 2017) learn the structure in a likelihood-free manner. While learning the evidence lower bound, soft constraints were embedded to Variational Auto-Encoder (VAE) (Kingma & Welling, 2013) for molecule design (Kusner et al., 2017; Jin et al., 2018). However, these deep generative methods can hardly deal with hard combinatorial constraints.

Previous approaches embed machine learning models into the optimization by, e.g., integrating neural networks and decision trees with constraint programming (Lallouet & Legtchenko, 2007), or introducing a *Neuron* global constraint that represents a pre-trained neural network (Lombardi & Gualandi, 2016; Lombardi et al., 2017). Machine learning approaches have also been used to solve constraint reasoning and optimization problems (Galassi et al., 2018; Vinyals et al., 2015; Khalil et al., 2017). Graves et al. (2016) employs neural networks for discrete structure generation, while Wang et al. (2019); Amos & Kolter (2017); Agrawal et al. (2019) and de Avila Belbute-Peres et al. (2020) integrate logical reasoning and differentiable optimization problems within deep learning architectures. Parity constraints are proposed for both sampling (Gomes et al., 2007a; Ermon et al., 2013b) and counting problems (Ermon et al., 2013a; Chakraborty et al., 2014; Achlioptas & Theodoropoulos, 2017; Achlioptas et al., 2018; Ding et al., 2019) in probabilistic inference. These approaches provide constant approximation guarantees on either the probability of the samples or the estimated values of discrete integration.

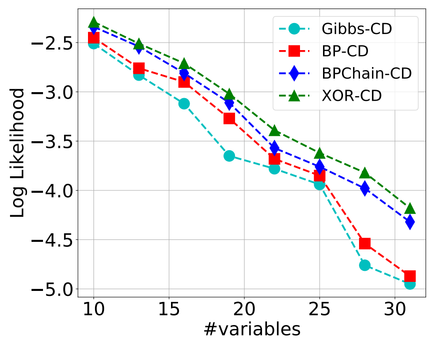


Figure 3. Averaged log likelihood of 100 structures generated by different learning algorithms on a discrete exponential family model varying the number of variables. The structures generated by XOR-CD have the highest average log-likelihoods.

5. Experiments

In this section we show the superior performance of XOR-CD on 4 structure generation experiments, one on synthetic data generated by a known model and the other three are dispatching route generation, optimal experimental design, and sequence-based protein homology detection. One baseline is Contrastive Divergence CD_{100}, denoted as Gibbs-CD, which uses Gibbs Sampling (Carreira-Perpinan & Hinton, 2005) of 100 steps to obtain the samples from the model distribution. We also compare with Generative Adversarial Networks (GAN) (Goodfellow et al., 2014), belief propagation-based CD approaches, BP-CD, and the recent BPChain-CD (Fan & Xue, 2020). Various experiment settings are left to the supplementary materials. In Figure 3, we show XOR-CD learns the highest log likelihood on a synthetic dataset (details in the supplementary materials).

5.1. Dispatching Route Generation

We consider the problem of generating dispatching routes for delivery drivers. The delivery routes need to form Hamiltonian cycles, where each location is visited once and ex-

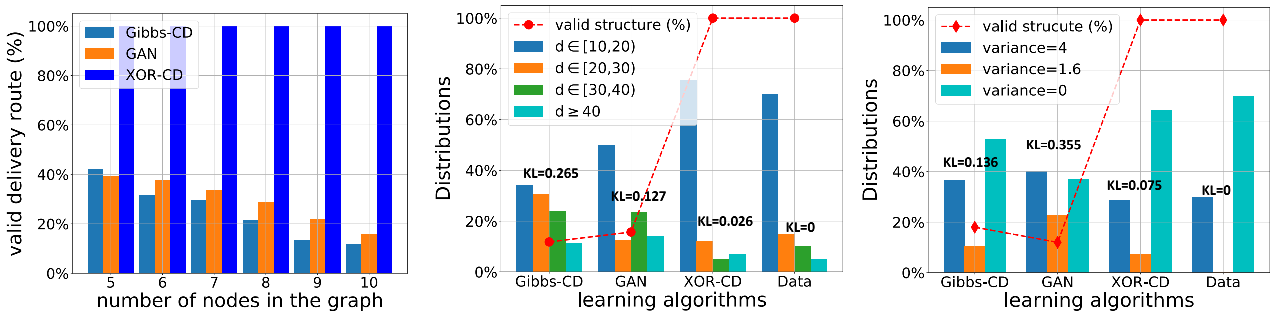


Figure 4. XOR-CD outperforms competing approaches by producing 100% valid delivery routes and experiment designs while capturing the inductive bias in training data. (**Left**) The percentage of valid routes generated by different algorithms, varying the number of locations. (**Middle**) The dashed line shows the percentage of valid routes generated from different algorithms when the number of locations is 10. The bars show the distributions of these valid routes grouped by the traveling distances d. The distribution of XOR-CD closely matches that of the training data with the smallest KL divergence 0.026. (**Right**) The dashed line shows the percentage of valid experiment designs generated from different algorithms on 5×5 grids. The bars show the distributions of these valid structures grouped by variance. XOR-CD generates 100% valid designs, and has the minimal KL divergence 0.075 towards the training data distribution.

actly once. For this experiment, we assume the number of delivery locations n is fixed, although it is not difficult to extend our approach to the cases where n varies. In the learning phase, we are given a dataset of historical trips, where each trip is represented with a permutation of n locations, denoting the order of in which these locations are visited. We learn an exponential family model to capture the likelihood of different permutations. Specifically, denote $x_{i,j}$ as a binary indication variable, which is 1 if and only if the i-th location to visit is location j. The exponential family model is $Pr(x) \propto \exp(\theta_0 + \sum_{i,j} \theta_{i,j} x_{i,j} + \sum_{i,j,k} \theta_{i,j,k} x_{i,j} x_{i+1,k})$ and the θ's are the parameters to learn. Learning rate is fixed as 0.1 and total number of epochs T is 500. There is also a timeout of 10 hours for all algorithms. We also set both M and K to be 100, and parameters for XOR-Sampling were kept the same as in (Ermon et al., 2013b). We leave the data generation process, and how we add the Hamilton cycle constraint into XOR-CD to the supplementary materials. In solving the inference problem during testing, we have a series of additional constraints detailing the requirement of a new day delivery. Such constraints include e.g., certain locations must be visited first, and one location must be visited after another location, etc. Therefore, the inference problem is to find variable assignments to all $x_{i,j}$ variables, which maximizes the likelihood, while satisfies the Hamilton cycle and the additional constraints.

We first examine the validity of the routes generated. The left figure in Figure 4 shows the percentage of valid Hamilton cycles generated from different algorithms, varying the numbers of delivery locations from 5 to 10. We can see that XOR-CD can generate 100% valid Hamilton structures while the competing methods at best generate 40% valid routes. The red dashed line in the middle figure shows the percentage of valid Hamilton cycles generated from different algorithms when the number of locations is 10. We then examine whether the generated routes resemble those in the training data. To validate this, we evaluate the distribution of the total lengths of the routes generated. Without imposing additional constraints, the lengths distribution of the generated routes should closely resembles that of the training set for a successful learning algorithm. The bars in Figure 4 (middle) demonstrate such distributions of the valid routes generated by each algorithm. The last column shows the distribution of the training data. We can see the distribution from XOR-CD closely matches the data distribution, with KL divergence of 0.026. Other approaches are worse. This indicates that XOR-CD is able to capture the inductive bias of the training set better than competing approaches.

5.2. Optimal Experiment Design

We further consider the optimal experiment design problem. Here, we generate an experiment design in the form of a Latin square, which is a n by n matrix and each entry can be planted with crop 1 to n. Each crop needs to be planted exactly once in each row and column. Let $x_{i,j,k}$ be an indicator variable which is 1 if and only if crop k is planted at the i,j-th entry. The exponential family model is: $Pr(x) \propto \exp(\theta_0 + \sum_{i,j,k} \theta_{i,j,k} x_{i,j,k} + \sum_{i,j,m,l} \theta^1_{i,j,m,l} \phi(x_{i,j,m}, x_{i+1,j,l}) + \theta^2_{i,j,m,l} \phi(x_{i,j,m}, x_{i,j+1,l}))$. The learning parameters are set the same as in the previous task. The learning problem is to identify the values of θ's. During testing, the inference problem is to generate experiment designs from partially filled matrices satisfying the Latin square requirement while closely resemble those in the training set.

We first examine the validity of experiment designs. Here we consider generating 5×5 Latin squares. The percentage of valid Latin squares are shown in the red dashed curve of Figure 4 (right). Again, XOR-CD generates 100% valid

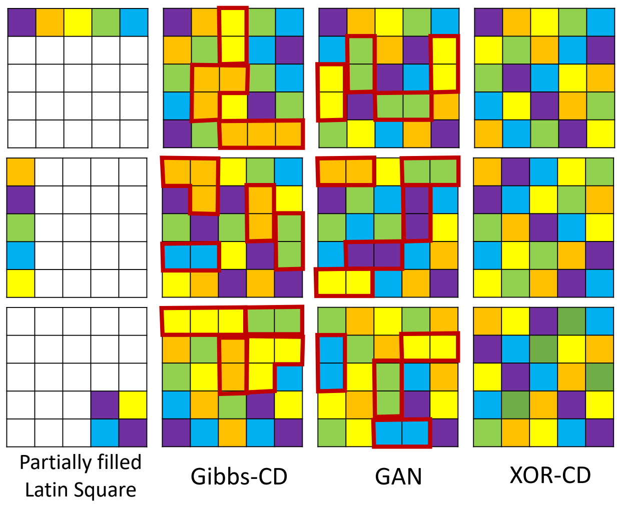


Figure 5. XOR-CD (column 4) generates valid Latin squares from partially filled structures (column 1), which all have variance 4, the most frequent variance in the training dataset. Gibbs-CD (column 2) and GAN (column 3) cannot generate valid Latin squares (constraints violation shown in red boxes).

experiment designs while competing approaches at best generate 20%. To judge how well the generated Latin squares resemble those in the training set, we evaluate the distribution of the spatial variance of the generated Latin squares. This metric was used as an additional criterion for good experiment designs (see, e.g., (Gomes et al., 2004; Smith et al.; Le Bras et al., 2012)). Without partially filled cells, the generated Latin squares should be close in spatial variance as those in the training set. The bars in Figure 4 shows that the distribution of the spatial variance of Latin squares generated by XOR-CD matches that of the training set most with KL divergence of 0.075. In addition, Figure 5 shows that XOR-CD is able to complete a partially-filled Latin square resembling those in the training set while Gibbs-CD and GAN cannot generate valid structures.

5.3. Sequence-based Protein Homology Detection

We also consider a real-world task, sequence-based protein homology detection. Our approach is based upon comparing protein sequence profiles, which are derived from multiple sequence alignment (MSA) of homologies in a protein family. Let $\mathcal{S}_1$ and $\mathcal{S}_2$ be two sequences of amino acids. Our goal is to align the two sequences. Our tasks are: (1) **(learning)** given a dataset of aligned pairs of amino acid sequences, learn the likelihoods of different alignments of the two sequences; (2) **(inference)** given a new pair of amino acid sequences, determine their most likely alignment.

The exponential family model for protein alignment is similar to the one used in (Ma et al., 2014), where we use flow constraints to guarantee the solution to form a valid path in the alignment matrix. The details are left to the supplementary materials. We constructed the training set from the

	Dynamic Program	Gibbs-CD	XOR-CD
valid align.	**100%**	0%	**100%**

	Precision		Recall	
	exact match	4-offset	exact match	4-offset
Dynamic Program	28.7%	39.5%	33.5%	41.2%
Gibbs-CD	39.6%	47.8%	37.9%	45.4%
XOR-CD	**48.8%**	**54.3%**	**45.3%**	**52.1%**

Table 1. **(Upper)** XOR-CD and dynamic programming generate 100% valid alignments for protein homology detection, while Gibbs-CD cannot. **(Lower)** Precision and recall of the alignments found by different approaches. XOR-CD outperforms two baselines by a large margin, even when both metrics are calculated taking into account both valid and invalid alignments. 4-offset is a relaxed measure.

PDB40 dataset (Wu & Xu, 2020). Following the practice of (Ma et al., 2014), the reference alignment (groundtruth) is generated by DeepAlign (Wang et al., 2013). Our experiment data include those whose groundtruth alignment has fewer than 50 gap positions (excluding the gap in the beginning and end) and the total length is up to 200. The test set is made up with 50 randomly sampled sequences from the PDB40 dataset separated from the training set. Following common practice, we use precision and recall as evaluation metrics. Precision is the fraction of correctly aligned amino acid pairs within all predicted ones, and recall is the fraction of correctly aligned pairs within all ground-truth ones. Notice these two metrics are local, and can be computed even when the global alignment is invalid (does not form a path in the alignment matrix). We compare XOR-CD with dynamic programming and Gibbs-CD. Dynamic programming uses an expert-defined objective (Rychlewski et al., 2000) with a few learned terms. It always produces valid alignments.

As shown in Table 1 (Upper), both dynamic programming and XOR-CD have the ability to generate 100% valid alignments, while Gibbs-CD cannot. XOR-CD outperforms both baselines in precision and recall in Table 1 (Lower). 4-position off is a relaxed metric that considers a alignment correct if it is off by at most 4 positions. Using this relaxed metric, XOR-CD still outperforms both baselines by 7% in precision and 14% in recall. Notice these metrics are computed taking into both valid and invalid alignments. In summary, XOR-CD outperforms baselines in all learning metrics while also generating 100% valid alignments.

6. Conclusion

We proposed XOR-CD, a novel algorithm for constrained structure generation. We showed theoretically that XOR-CD has a linear convergence rate to the global optimum for exponential family models. Empirically, we demonstrated the superior performance of XOR-CD on three real-world constrained structure generation tasks. In all tasks, XOR-CD generates 100% valid structures and these generated structures closely match those in the training set. Future work includes extending XOR-CD to deep generative models.

Acknowledgements

This research was supported by NSF grants IIS-1850243, CCF-1918327. We thank anonymous reviewers for their comments and suggestions.

References

Achlioptas, D. and Theodoropoulos, P. Probabilistic model counting with short xors. In *International Conference on Theory and Applications of Satisfiability Testing*, pp. 3–19. Springer, 2017.

Achlioptas, D., Hammoudeh, Z., and Theodoropoulos, P. Fast and flexible probabilistic model counting. In *International Conference on Theory and Applications of Satisfiability Testing*, 2018.

Agrawal, A., Amos, B., Barratt, S., Boyd, S., Diamond, S., and Kolter, J. Z. Differentiable convex optimization layers. In *Advances in neural information processing systems*, pp. 9562–9574, 2019.

Amos, B. and Kolter, J. Z. Optnet: Differentiable optimization as a layer in neural networks. *arXiv preprint arXiv:1703.00443*, 2017.

Bao, F., Li, C., Xu, T., Su, H., Zhu, J., and Zhang, B. Bilevel score matching for learning energy-based latent variable models. *Advances in Neural Information Processing Systems*, 33, 2020.

Barton, J. P., De Leonardis, E., Coucke, A., and Cocco, S. Ace: adaptive cluster expansion for maximum entropy graphical model inference. *Bioinformatics*, 32(20):3089–3097, 2016.

Belanger, D. and McCallum, A. Structured prediction energy networks. In *International Conference on Machine Learning*, pp. 983–992, 2016.

Belanger, D., Yang, B., and McCallum, A. End-to-end learning for structured prediction energy networks. *arXiv preprint arXiv:1703.05667*, 2017.

Bengio, Y. and Delalleau, O. Justifying and generalizing contrastive divergence. *Neural computation*, 21(6):1601–1621, 2009.

Briot, J.-P., Hadjeres, G., and Pachet, F.-D. Deep learning techniques for music generation–a survey. *arXiv preprint arXiv:1709.01620*, 2017.

Brock, A., Donahue, J., and Simonyan, K. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Carreira-Perpinan, M. A. and Hinton, G. E. On contrastive divergence learning. In *Aistats*, volume 10, pp. 33–40. Citeseer, 2005.

Chakraborty, S., Fremont, D. J., Meel, K. S., Seshia, S. A., and Vardi, M. Y. Distribution-aware sampling and weighted model counting for sat. In *AAAI*, 2014.

de Avila Belbute-Peres, F., Economon, T. D., and Kolter, J. Z. Combining differentiable pde solvers and graph neural networks for fluid flow prediction. 2020.

Dehaene, G. P. and Barthelmé, S. Bounding errors of expectation-propagation. In *Advances in Neural Information Processing Systems*, pp. 244–252, 2015.

Ding, F., Wang, H., Sabharwal, A., and Xue, Y. Towards efficient discrete integration via adaptive quantile queries. *arXiv preprint arXiv:1910.05811*, 2019.

Durkan, C., Murray, I., and Papamakarios, G. On contrastive learning for likelihood-free inference. *arXiv preprint arXiv:2002.03712*, 2020.

Engel, J., Resnick, C., Roberts, A., Dieleman, S., Norouzi, M., Eck, D., and Simonyan, K. Neural audio synthesis of musical notes with wavenet autoencoders. In *International Conference on Machine Learning*, pp. 1068–1077. PMLR, 2017.

Ermon, S., Gomes, C. P., Sabharwal, A., and Selman, B. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *Proceedings of the 30th International Conference on Machine Learning, ICML*, 2013a.

Ermon, S., Gomes, C. P., Sabharwal, A., and Selman, B. Embed and project: Discrete sampling with universal hashing. In *Advances in Neural Information Processing Systems (NIPS)*, 2013b.

Fan, D. and Xue, Y. Contrastive divergence learning with chained belief propagation. In *International Conference on Probabilistic Graphical Models*, 2020.

Galassi, A., Lombardi, M., Mello, P., and Milano, M. Model Agnostic Solution of CSPs via Deep Learning: A Preliminary Study. In *Proceedings of CPAIOR*, volume 10848 of *LNCS*, pp. 254–262. Springer, 2018.

Gomes, C., Sellmann, M., Van Es, C., and Van Es, H. The challenge of generating spatially balanced scientific experiment designs. In *International Conference on Integration of Artificial Intelligence (AI) and Operations*

Research (OR) Techniques in Constraint Programming, pp. 387–394. Springer, 2004.

Gomes, C. P., Sabharwal, A., and Selman, B. Near-uniform sampling of combinatorial spaces using xor constraints. In Schölkopf, B., Platt, J. C., and Hoffman, T. (eds.), *Advances in Neural Information Processing Systems 19*, pp. 481–488. MIT Press, 2007a.

Gomes, C. P., Van Hoeve, W.-J., Sabharwal, A., and Selman, B. Counting csp solutions using generalized xor constraints. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, 2007b.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwinska, A., Colmenarejo, S. G., Grefenstette, E., Ramalho, T., Agapiou, J., Badia, A. P., Hermann, K. M., Zwols, Y., Ostrovski, G., Cain, A., King, H., Summerfield, C., Blunsom, P., Kavukcuoglu, K., and Hassabis, D. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538 (7626):471–476, 2016.

Grover, A., Zweig, A., and Ermon, S. Graphite: Iterative generative modeling of graphs. In *International conference on machine learning*, pp. 2434–2444. PMLR, 2019.

Hinton, G. and Salakhutdinov, R. Reducing the dimensionality of data with neural networks. *Science*, 313(5786): 504 – 507, 2006.

Hinton, G. E. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, pp. 1771–1800, 2002a.

Hinton, G. E. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002b.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

Jiang, B., Wu, T.-Y., Jin, Y., Wong, W. H., et al. Convergence of contrastive divergence algorithm in exponential family. *The Annals of Statistics*, 46(6A):3067–3098, 2018.

Jin, W., Barzilay, R., and Jaakkola, T. S. Junction tree variational autoencoder for molecular graph generation. In *ICML*, 2018.

Khalil, E., Dai, H., Zhang, Y., Dilkina, B., and Song, L. Learning combinatorial optimization algorithms over graphs. In *Advances in Neural Information Processing Systems*, pp. 6348–6358. 2017.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in neural information processing systems*, pp. 10215–10224, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kusner, M. J., Paige, B., and Hernández-Lobato, J. M. Grammar variational autoencoder. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 1945–1954, 2017.

Lallouet, A. and Legtchenko, A. Building Consistencies for Partially Defined Constraints with Decision Trees and Neural Networks. *International Journal on Artificial Intelligence Tools*, 16(4):683–706, 2007.

Le Bras, R., Gomes, C., and Selman, B. From streamlined combinatorial search to efficient constructive procedures. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

Liu, Q., Allamanis, M., Brockschmidt, M., and Gaunt, A. Constrained graph variational autoencoders for molecule design. In *Advances in neural information processing systems*, pp. 7795–7804, 2018.

Lombardi, M. and Gualandi, S. A lagrangian propagator for artificial neural networks in constraint programming. *Constraints*, 21(4):435–462, 2016.

Lombardi, M., Milano, M., and Bartolini, A. Empirical decision model learning. *Artif. Intell.*, 244:343–367, 2017.

Ma, J., Wang, S., Wang, Z., and Xu, J. Mrfalign: protein homology detection through alignment of markov random fields. *PLoS Comput Biol*, 10(3):e1003500, 2014.

Minka, T. P. Expectation propagation for approximate bayesian inference. *arXiv preprint arXiv:1301.2294*, 2013.

Pang, T., Xu, K., Li, C., Song, Y., Ermon, S., and Zhu, J. Efficient learning of generative models via finite-difference score matching. *arXiv preprint arXiv:2007.03317*, 2020.

Ping, W. and Ihler, A. Belief propagation in conditional rbms for structured prediction. *arXiv preprint arXiv:1703.00986*, 2017.

Prenger, R., Valle, R., and Catanzaro, B. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3617–3621. IEEE, 2019.

Qiu, Y., Zhang, L., and Wang, X. Unbiased contrastive divergence algorithm for training energy-based latent variable models. In *International Conference on Learning Representations*, 2019.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Rychlewski, L., Li, W., Jaroszewski, L., and Godzik, A. Comparison of sequence profiles. strategies for structural predictions using sequence information. *Protein Science*, 9(2):232–241, 2000.

Smith, C., Gomes, C., and Fernandez, C. Streamlining local search for spatially balanced latin squares.

Söding, J. Protein homology detection by hmm–hmm comparison. *Bioinformatics*, 21(7):951–960, 2005.

Song, Y. and Ermon, S. Improved techniques for training score-based generative models. *Advances in Neural Information Processing Systems*, 33, 2020.

Vinyals, O., Fortunato, M., and Jaitly, N. Pointer networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, pp. 2692–2700. 2015.

Wang, P.-W., Donti, P. L., Wilder, B., and Kolter, Z. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. *arXiv preprint arXiv:1905.12149*, 2019.

Wang, S., Ma, J., Peng, J., and Xu, J. Protein structure alignment beyond spatial proximity. *Scientific reports*, 3: 1448, 2013.

Wu, F. and Xu, J. Deep template-based protein structure prediction. *bioRxiv*, 2020.

Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018.

Supplementary Materials

A. XOR-Sampling for the Weighted Case

The text here provides a synopsis for the approach in (Ermon et al., 2013b). We still encourage the readers to read the original text for a better explanation. Let $w(x)$ as defined before, $Z = \sum_{x\in\mathcal{X}} w(x)$ and $P(x) = w(x)/Z$, the high-level idea of XOR-Sampling is to first dicretize $w(x)$ to $w'(x)$ as in Definition 1, followed by embedding the weighted $w'(x)$ to the unweighted space Δ_w. Finally, XOR-sampling uses hashing and randomization to sample uniformly from Δ_w.

Definition 1. *Assume $w(x)$ has both upper and lower bound, namely, $M = \max_x w(x)$ and $m = \min_x w(x)$. Let $b \geq 1, \epsilon > 0, r = 2^b/(2^b - 1)$ and $l = \lceil \log_r(2^n/\epsilon)\rceil$. Partition the configurations into the following weight based disjoint buckets: $\mathcal{B}_i = \{x | w(x) \in (\frac{M}{r^{i+1}}, \frac{M}{r^i}]\}, i = 0, \dots, l-1$ and $\mathcal{B}_l = \{x | w(x) \in (0, \frac{M}{r^l}]\}$. The discretized weight function $w' : \{0,1\}^n \to \mathbb{R}^+$ is defined as follows: $w'(x) = \frac{M}{r^{i+1}}$ if $x \in \mathcal{B}_i, i = 0, \dots, l-1$ and $w'(x) = 0$ if $x \in \mathcal{B}_l$. This leads to the corresponding discretized probability distribution $p'(x) = w'(x)/Z'$ where Z' is the normalization constant of $w'(x)$.*

For the weighted case, the goal of XOR-sampling is to guarantee that the probability of sampling one x is proportional to the unnormalized density (up to a multiplicative constant). Using the discretization in Definition 1, we obtain a distribution $p'(x)$ which satisfying $\frac{1}{\rho}p(x) \leq p'(x) \leq \rho p(x)$ where $\rho = \frac{r^2}{1-\epsilon}$. Then, XOR-sampling implements a horizontal slice technique to transform a weighted problem into an unweighted one. For the easiness of illustration, we denote $M' = \max_x w'(x)$ and m' as the smallest nonzero value of $w'(x)$. Then consider a simple case where $b = 1$ and $r = 2$. In this case we have $M' = 2^{l-1}m'$. Let $y = (y_0, \dots, y_{l-2})^T \in \{0,1\}^{l-1}$ be a binary vector of length $l-1$, XOR-sampling samples (x, y) uniformly at random from the following set Δ_w using the unweighted version of XOR-sampling based on hashing and randomization:

$$\Delta_w = \{(x, y) : w'(x) \leq 2^{i+1}m' \Rightarrow y_i = 0\}. \quad (13)$$

Upon obtaining one sample (x, y) uniformly at random from Δ_w, we only return x. It can be proved that the probability of sampling x from $w'(x)$ is proportional to $m'2^{i-1}$ when $w'(x)$ is sandwiched between $m'2^{i-1}$ and $m'2^i$. Therefore, this technique leads to the constant approximation guarantee of XOR-Sampling, which states formally as Theorem 5:

Theorem 5. *(Ermon et al., 2013b) Let $\epsilon > 0, b > 1, P \geq 2$, $0 < \delta_0 < 1$, and $\gamma_0 = \log((P+2\sqrt{P+1}+2)/P)$. For any $\alpha \in \mathbb{Z}$, $\alpha > \gamma_0$, let $c(\alpha, P) = 1 - 2^{\gamma_0-\alpha}/(1 - \frac{1}{P} - 2^{\gamma_0-\alpha})^2$. Let $r = 2^b/(2^b-1), l = \lceil \log_r(2^n/\epsilon)\rceil, \rho = r^2/(1-\epsilon), \kappa = 1/c(\alpha, P)$ and bucket $\mathcal{B}_l$ as in Definition 1 in the supplementary materials. Denote $Pr'_s(x)$ as distribution of the samples generated by XOR-Sampling($w, l, b, \delta, P, \alpha$) and let $\phi : \{0,1\}^n \to \mathbb{R}^+$ be one non-negative function. Then, with probability at least $(1-\delta_0)c(\alpha, P)2^{-(\gamma_0+\alpha+1)}\frac{P}{P-1}$, XOR-Sampling succeeds and outputs a sample x_0. Upon success, each x_0 is output with probability $P'(x_0)$, which is within a constant factor of the true $P(x_0)$. Furthermore, the expectation of a non-negative function $\phi(x)$, $\mathbb{E}_{P(x)}[\phi(x)]$ can be bounded by:*

$$\begin{aligned}\frac{1}{\rho\kappa}\mathbb{E}_{P'(x)}[\phi(x)] - \epsilon\eta_\phi \leq \mathbb{E}_{P(\theta)}[\phi(x)] \\ \leq \rho\kappa\mathbb{E}_{P'(x)}[\phi(x)] + \epsilon\eta_\phi.\end{aligned} \quad (14)$$

Theorem 1 is a simplified representation of the previous Theorem. Our new theoretic results can be stated simpler and clearer building upon the statement in Theorem 1. Assuming all the parameters of Theorem 5, we set $\delta = \rho\kappa$, and $\gamma = 1 - (1-\delta_0)c(\alpha, P)2^{-(\gamma_0+\alpha+1)}\frac{P}{P-1}$ and can obtain the corresponding guarantee stated in Theorem 1.

B. Proofs

Theorem 2 states that the function value of the output of XOR-CD, in expectation converges to the true optimum within a small constant distance at a linear speed w.r.t. the number of iterations T. To prove Theorem 2, we first prove two lemmas.

Lemma 1. *If the total variation $\max_\theta Var_{P_\theta}(\phi(X)) \leq \sigma_2^2$, then $l(\theta)$ is σ_2^2-smooth w.r.t. θ.*

B.1. Proof of Lemma 1

Proof. Since $l(\theta) = -\frac{1}{N}\sum_{i=1}^N \log P_\theta(x_i)$, L-smoothness requires that

$$||\nabla l(\theta_1) - \nabla l(\theta_2)||_2 \leq L||\theta_1 - \theta_2||_2, \ \ \forall \theta_1, \theta_2 \in \ dom\ f,$$

where L is a constant. Because of the mean value theorem, there exists a point $\tilde{\theta} \in (\theta_1, \theta_2)$ such that

$$\nabla l(\theta_1) - \nabla l(\theta_2) = \nabla(\nabla l(\tilde{\theta}))(\theta_1 - \theta_2).$$

Taking the L_2 norm for both sides, we have

$$\begin{aligned}||\nabla l(\theta_1) - \nabla l(\theta_2)||_2 =& ||\nabla(\nabla l(\tilde{\theta}))(\theta_1 - \theta_2)||_2 \\ \leq& ||\nabla(\nabla l(\tilde{\theta}))||_2 \, ||\theta_1 - \theta_2||_2\end{aligned} \quad (15)$$

Then, the problem is to bound the matrix 2-norm $||\nabla(\nabla l(\tilde{\theta}))||_2$. Since we know the explicit form of $l(\theta)$, we know

$$\nabla l(\theta) = \nabla\Lambda(\theta) - \frac{1}{N}\sum_{i=1}^N \phi(x_i),$$

$$\nabla(\nabla l(\theta)) = \sum_{x\in\mathcal{X}} [\phi(x) - \nabla\Lambda(\theta)][\phi(x) - \nabla\Lambda(\theta)]^T P_\theta(x), \quad (16)$$

where $\nabla(\nabla l(\theta))$ is the co-variance matrix. Denote $\text{Cov}_\theta[\phi(X)] = \nabla(\nabla l(\theta))$, which is both symmetric and positive semi-definite. We have

$$||\nabla(\nabla l(\tilde{\theta}))||_2 = ||\text{Cov}_\theta[\phi(X)]||_2 = \lambda_{max},$$

where λ_{max} is the maximum eigenvalue of the matrix $\text{Cov}_\theta[\phi(X)]$. Then, because of the positive semi-definiteness of the co-variance matrix, all the eigenvalues are non-negative, and we can bound λ_{max} as

$$\lambda_{max} \leq \sum_i \lambda_i = Tr(\text{Cov}_\theta[\phi(X)]),$$

where $Tr(\text{Cov}_\theta[\phi(X)])$ is the trace of matrix $\text{Cov}_\theta[\phi(X)]$. Using the definition in Equation 16, $Tr(\text{Cov}_\theta[\phi(X)])$ can be further derived as:

$$Tr(\text{Cov}_\theta[\phi(X)]) = \mathbb{E}_{P_\theta}[||\phi(X)||_2^2] - ||\mathbb{E}_{P_\theta}[\phi(X)]||_2^2,$$

which is equal to the total variation $Var_{P_\theta}(\phi(X))$. Therefore, we have

$$||\nabla(\nabla l(\tilde{\theta}))||_2 \leq Var_{P_\theta}(\phi(X)) \leq \sigma_2^2.$$

Combining this with Equation 15, we know

$$||\nabla l(\theta_1) - \nabla l(\theta_2)||_2 \leq \sigma_2^2 \; ||\theta_1 - \theta_2||_2.$$

This completes the proof.

□

In addition, based on the constant approximation of $\mathbb{E}_{D,P'_\theta}[\overline{g_t}]$, we can bound another two important terms shown in Lemma 2.

Lemma 2. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex function and $\theta^* = argmin_\theta f(\theta)$. In iteration t, g_t is the estimated gradient. If there exists a constant $c \geq 1$ s.t. $\frac{1}{c}[\nabla f(\theta_t)]^+ \leq \mathbb{E}[g_t^+] \leq c[\nabla f(\theta_t)]^+$ and $c[\nabla f(\theta_t)]^- \leq \mathbb{E}[g_t^-] \leq \frac{1}{c}[\nabla f(\theta_t)]^-$, then we have*

$$\frac{1}{c}||\mathbb{E}[g_t]||_2^2 \leq \langle \nabla f(\theta_t), \mathbb{E}[g_t] \rangle \leq c||\mathbb{E}[g_t]||_2^2.$$

$$\frac{1}{c}\langle \mathbb{E}[g_t], \theta_t - \theta^* \rangle \leq \langle \nabla f(\theta_t), \theta_t - \theta^* \rangle \leq c\langle \mathbb{E}[g_t], \theta_t - \theta^* \rangle.$$

B.2. Proof of Lemma 2

Proof. (Lemma 2) Since we have the constant bound that

$$\frac{1}{c}\nabla f(\theta_t)^+ \leq \mathbb{E}[g_t^+] \leq c\nabla f(\theta_t)^+. \tag{17}$$

$$c\nabla f(\theta_t)^- \leq \mathbb{E}[g_t^-] \leq \frac{1}{c}\nabla f(\theta_t)^-. \tag{18}$$

and because of $g_t^+ \geq \mathbf{0}$ and $g_t^- \leq \mathbf{0}$ we can obtain

$$\frac{1}{c}||\mathbb{E}[g_t^+]||_2^2 = \frac{1}{c}\langle \mathbb{E}[g_t^+], \mathbb{E}[g_t^+] \rangle \leq \langle \nabla f(\theta_t)^+, \mathbb{E}[g_t^+] \rangle$$
$$\leq c\langle \mathbb{E}[g_t^+], \mathbb{E}[g_t^+] \rangle = c||\mathbb{E}[g_t^+]||_2^2.$$
$$\frac{1}{c}||\mathbb{E}[g_t^-]||_2^2 = \frac{1}{c}\langle \mathbb{E}[g_t^-], \mathbb{E}[g_t^-] \rangle \leq \langle \nabla f(\theta_t)^-, \mathbb{E}[g_t^-] \rangle$$
$$\leq c\langle \mathbb{E}[g_t^-], \mathbb{E}[g_t^-] \rangle = c||\mathbb{E}[g_t^-]||_2^2.$$

which exactly means

$$\frac{1}{c}||\mathbb{E}[g_t]||_2^2 \leq \langle \nabla f(\theta_t), \mathbb{E}[g_t] \rangle \leq c||\mathbb{E}[g_t]||_2^2.$$

To prove the second inequality, we need to take advantage of the convexity of f. Denote $[\theta_t - \theta^*]^+ = \max\{\theta_t - \theta^*, \mathbf{0}\}$ and $[\theta_t - \theta^*]^- = \min\{\theta_t - \theta^*, \mathbf{0}\}$, we know $\theta_t - \theta^* = [\theta_t - \theta^*]^+ + [\theta_t - \theta^*]^-$. In addition, because f is convex, the index set of non-zero entries of $[\theta_t - \theta^*]^+$ and $\nabla f(\theta_t)^+$ is the same. The index set of non-zero entries of $[\theta_t - \theta^*]^-$ and $\nabla f(\theta_t)^-$ is also the same. In addition, because of Equation 17 and 18, the index set of non-zero entries of $\mathbb{E}[g_t^+]$ ($\mathbb{E}[g_t^-]$) is the same with $\nabla f(\theta_t)^+$ ($\nabla f(\theta_t)^-$). Combining these facts with Equations 17 and 18, we have

$$\frac{1}{c}\langle \mathbb{E}[g_t^+], [\theta_t - \theta^*]^+ \rangle \leq \langle \nabla f(\theta_t)^+, [\theta_t - \theta^*]^+ \rangle$$
$$\leq c\langle \mathbb{E}[g_t^+], [\theta_t - \theta^*]^+ \rangle.$$
$$\frac{1}{c}\langle \mathbb{E}[g_t^-], [\theta_t - \theta^*]^- \rangle \leq \langle \nabla f(\theta_t)^-, [\theta_t - \theta^*]^- \rangle$$
$$\leq c\langle \mathbb{E}[g_t^-], [\theta_t - \theta^*]^- \rangle.$$

Combining these two equations, we have

$$\frac{1}{c}\langle \mathbb{E}[g_t], \theta_t - \theta^* \rangle \leq \langle \nabla f(\theta_t), \theta_t - \theta^* \rangle \leq c\langle \mathbb{E}[g_t], \theta_t - \theta^* \rangle.$$

This completes the proof. □

Lemma 2 gives the new bounds of two terms assuming the constant bound on the gradient, which are essential to the proof of convergence rate. Based on Lemma 2, we can prove Theorem 3, which bounds the error of Stochastic Gradient Descent (SGD) on a convex optimization problem when the estimated gradient g_t in the t-th step resides in a constant bound of $\nabla f(\theta_t)$.

B.3. Proof of Theorem 3

Theorem 3 indicates that even the expectation of gradients in each iteration only have a constant approximation, SGD is still able to converge to the optimal solution within a small constant gap at linear speed. The complete proof of Theorem 3 is as follows:

Proof. (Theorem 3) By L-smooth of f, for the t-th iteration,

$$f(\theta_{t+1}) \leq f(\theta_t) + \langle \nabla f(\theta_t), \theta_{t+1} - \theta_t \rangle + \frac{L}{2}||\theta_{t+1} - \theta_t||_2^2,$$
$$= f(\theta_t) - \eta\langle \nabla f(\theta_t), g_t \rangle + \frac{Lt^2}{2}||g_t||^2.$$

Because of the constant bound on gradient and $||\mathbb{E}[g_t]||_2^2 = \mathbb{E}[||g_t||_2^2] - Var(g_t)$, by taking expectation on both sides w.r.t g_t we get from Lemma 2 that

$$\begin{aligned}\mathbb{E}[f(\theta_{t+1})] &\leq f(\theta_t) - \frac{\eta}{c}||\mathbb{E}[g_t]||_2^2 + \frac{L\eta^2}{2}\mathbb{E}[||g_t||_2^2],\\ &= f(\theta_t) - \frac{\eta}{c}(\mathbb{E}[||g_t||_2^2] - Var(g_t)) + \frac{L\eta^2}{2}\mathbb{E}[||g_t||_2^2],\\ &\leq f(\theta_t) - \frac{\eta(2-L\eta c)}{2c}\mathbb{E}[||g_t||_2^2] + \frac{\eta}{c}\sigma^2,\\ &\leq f(\theta_t) - \frac{\eta c}{2}\mathbb{E}[||g_t||_2^2] + \frac{\eta}{c}\sigma^2,\end{aligned}$$

where the last inequality follows as $L\eta c \leq 2 - c^2$. Because f is convex, still from Lemma 2 we get

$$\begin{aligned}&\mathbb{E}[f(\theta_{t+1})]\\ &\leq f(\theta^*) + \langle\nabla f(\theta_t), \theta_t - \theta^*\rangle - \frac{\eta c}{2}\mathbb{E}[||g_t||_2^2] + \frac{\eta}{c}\sigma^2,\\ &\leq f(\theta^*) + c\langle\mathbb{E}[g_t], \theta_t - \theta^*\rangle - \frac{\eta c}{2}\mathbb{E}[||g_t||_2^2] + \frac{\eta}{c}\sigma^2,\\ &= f(\theta^*) + c\mathbb{E}[\langle g_t, \theta_t - \theta^*\rangle - \frac{\eta}{2}||g_t||_2^2] + \frac{\eta}{c}\sigma^2.\end{aligned}$$

We now repeat the calculations by completing the square for the middle two terms to get

$$\begin{aligned}&\mathbb{E}[f(\theta_{t+1})]\\ &\leq f(\theta^*) + \frac{c}{2\eta}\mathbb{E}[2\eta\langle g_t, \theta_t - \theta^*\rangle - \eta^2||g_t||_2^2] + \frac{\eta}{c}\sigma^2,\\ &\leq f(\theta^*) + \frac{c}{2\eta}\mathbb{E}[||\theta_t - \theta^*||_2^2 - ||\theta_t - \theta^* - \eta g_t||_2^2] + \frac{\eta}{c}\sigma^2,\\ &= f(\theta^*) + \frac{c}{2\eta}\mathbb{E}[(||\theta_t - \theta^*||_2^2 - ||\theta_{t+1} - \theta^*||_2^2)] + \frac{\eta}{c}\sigma^2.\end{aligned}$$

Summing the above equations for $t = 0, \ldots, T-1$, we get

$$\begin{aligned}&\sum_{t=0}^{T-1}\mathbb{E}[f(\theta_{t+1}) - f(\theta^*)]\\ &\leq \frac{c}{2\eta}(||\theta_0 - \theta^*||_2^2 - \mathbb{E}[||\theta_T - \theta^*||_2^2]) + \frac{T\eta}{c}\sigma^2\\ &\leq \frac{c||\theta_0 - \theta^*||_2^2}{2\eta} + \frac{T\eta}{c}\sigma^2.\end{aligned}$$

Finally, by Jensen's inequality, $tf(\overline{\theta_T}) \leq \sum_{t=1}^T f(\theta_t)$,

$$\begin{aligned}\sum_{t=0}^{T-1}\mathbb{E}[f(\theta_{t+1}) - f(\theta^*)] &= \mathbb{E}[\sum_{t=1}^{T} f(\theta_t)] - Tf(\theta^*)\\ &\geq T\mathbb{E}[f(\overline{\theta_T})] - Tf(\theta^*).\end{aligned}$$

Combining the above equations we get

$$\mathbb{E}[f(\overline{\theta_T})] \leq f(\theta^*) + \frac{c||\theta_0 - \theta^*||_2^2}{2\eta T} + \frac{\eta}{c}\sigma^2.$$

This completes the proof. □

B.4. Proof of Theorem 2

Finally, we give the full proof of Theorem 2 as follows:

Proof. (Theorem 2) Since we use M samples from the training set $\{x_i\}_{i=1}^N$ and K samples $x'_1, \ldots, x'_K$ from $P_{\theta_t}(X)$ using XOR-Sampling at each iteration, we have

$$\overline{g_t} = \frac{1}{M}\sum_{j=1}^{M}\phi(x_j) - \frac{1}{K}\sum_{i=1}^{K}\phi(x'_i).$$

Denote $g_t^i = \frac{1}{M}\sum_{j=1}^M \phi(x_j) - \phi(x'_i)$, we have the expectation of $\overline{g_t}$ as

$$\mathbb{E}_{D,P'_\theta}[\overline{g_t}] = \mathbb{E}_{P'_\theta}[\mathbb{E}_D[\phi(x)] - \phi(x')] = \mathbb{E}_{D,P'_\theta}[g_t^i].$$

In each iteration t we can adjust the parameters in XOR-Sampling to make the tail $\epsilon\eta_\phi$ zero, then for each g_t^i we can obtain from Theorem 1 that

$$\frac{1}{\delta}[g(\theta_t)]^+ \leq \mathbb{E}_{D,P'_\theta}[g_t^{i+}] \leq \delta[g(\theta_t)]^+. \tag{19}$$

$$\delta[g(\theta_t)]^- \leq \mathbb{E}_{D,P'_\theta}[g_t^{i-}] \leq \frac{1}{\delta}[g(\theta_t)]^-. \tag{20}$$

where $g(\theta_t)$ is the true gradient at t-th iteration. Denote $\overline{g_t}^+ = \max\{\overline{g_t}, \mathbf{0}\}$ and $\overline{g_t}^- = \min\{\overline{g_t}, \mathbf{0}\}$. Clearly, $g_t^{i+} \geq 0$ and $g_t^{i-} \leq 0$. Moreover, for a given dimension, either $g_t^{i+} = 0$ for that dimension or $g_t^{i-} = 0$. Evaluating $\overline{g_t}$ dimension by dimension, we can see that $\overline{g_t}^+ = \frac{1}{K}\sum_{i=1}^K g_t^{i+}$ and $\overline{g_t}^- = \frac{1}{K}\sum_{i=1}^K g_t^{i-}$. Combined with Equation 19 and 20, we know

$$\frac{1}{\delta}[g(\theta_t)]^+ \leq \mathbb{E}_{D,P'_\theta}[\overline{g_t}^+] \leq \delta[g(\theta_t)]^+.$$

$$\delta[g(\theta_t)]^- \leq \mathbb{E}_{D,P'_\theta}[\overline{g_t}^-] \leq \frac{1}{\delta}[g(\theta_t)]^-.$$

In terms of variance, the variance of each $\phi(x'_i)$ can be bounded by

$$\begin{aligned}Var_{P'_\theta}(\phi(x'_i)) &= \mathbb{E}_{P'_\theta}[||\phi(x_i)||_2^2] - ||\mathbb{E}_{P'_\theta}[\phi(x_i)]||_2^2,\\ &\leq \delta\mathbb{E}_{P_\theta}[||\phi(x_i)||_2^2],\\ &= \delta(Var_{P_\theta}(\phi(x_i)) + ||\mathbb{E}_{P_\theta}[\phi(x_i)]||_2^2),\\ &\leq \delta(\sigma_2^2 + \varepsilon^2).\end{aligned}$$

Because $\mathbb{E}_{D,P'_\theta}[\overline{g_t}] = \mathbb{E}_{D,P'_\theta}[g_t^i]$, the variance of $\overline{g_t}$, denoted as $Var_{D,P'_\theta}(\overline{g_t})$, can then be bounded as

$$\begin{aligned}&Var_{D,P'_\theta}(\overline{g_t})\\ &= Var_D(\frac{1}{M}\sum_{j=1}^{M}\phi(X_j)) + Var_{P'_\theta}(\frac{1}{K}\sum_{i=1}^{K}\phi(x'_i))\\ &= \frac{1}{M}Var_D(\phi(X_j)) + \frac{1}{K}Var_{P'_\theta}(\phi(x'_i))\end{aligned}$$

$$\leq \frac{1}{M}\sigma_1^2 + \frac{\delta}{K}(\sigma_2^2 + \varepsilon^2)$$

Therefore, since $l(\theta)$ is convex and σ_2^2-smooth from Lemma 1, we can then apply Theorem 3 to get the result in Theorem 2.

$$\begin{aligned} & OPT - \mathbb{E}[l(\overline{\theta^T})] \\ & \leq \frac{\delta||\theta_0 - \theta^*||_2^2}{2\eta T} + \frac{\eta \max_{\theta_t}\{Var_{D,P'_\theta}(\overline{g_t})\}}{\delta} \\ & \leq \frac{\delta||\theta_0 - \theta^*||_2^2}{2\eta T} + \frac{\eta(\sigma_2^2 + \varepsilon^2)}{K} + \frac{\eta\sigma_1^2}{\delta M}. \end{aligned}$$

This completes the proof. □

B.5. Proof of Theorem 4

Proof. (Theorem 4) From Theorem 1 we know that in each iteration of XOR-CD, we need to access $O(n \ln \frac{n}{\gamma})$ queries of NP oracles in order to generate one sample. However, as specified also in Ermon et al. (2013b), only the first sample needs those many queries. Once we have the first sample, the number of XOR constraints to add (depends on the sizes of the set Δ_w stated in supplementary materials section A) can be known in generating future samples for this SGD iteration. Therefore, we fix the number of XOR constraints added starting the generation of the second sample. As a result, we only need one NP oracle query in generating each of the following $K-1$ samples. Therefore, total queries in each iteration will be $O(n \ln \frac{n}{\gamma} + K)$. To complete all T SGD iterations, XOR-CD needs $O(Tn \ln \frac{n}{\gamma} + TK)$ NP oracle queries in total. □

C. Additional Experimental Details

Here we show some additional experiments we have done for this paper and additional details of the experiments discussed in the main text.

C.1. Maximum Likelihood Learning

Here we show XOR-CD is able to learn exponential family models with higher likelihood compared to competing approaches. We consider a discrete exponential family model with n binary variables $x = (x_1, \dots, x_n)^T$, where each $x_i \in \{0, 1\}$ for $i \in \{1, \dots, n\}$. The exponential family model we consider is in the form: $Pr(x) \propto \exp(\sum_{k=1}^{K} \theta_k^T \phi(x^k))$. Here, each x^k is a subset of all n variables, and is often referred to as a *clique*. Suppose x^k is of size l_k, ϕ is the Cartesian product, i.e., it maps a vector of l_k binary variables to a vector of size 2^{l_k}, where each entry in the vector evaluates to 1 if and only if x^k takes a particular assignment (There are 2^{l_k} different value assignments to l_k binary variables).

We synthetically generate a few exponential family models and test if learning algorithms can rediscover these models. In generating one model, we first draw the number of cliques uniformly from $[n, 2n]$. The size of each clique is chosen from the range of $[1, 6]$ at random. Then, to generate $\theta_k = (\theta_{k,1}, \dots, \theta_{k,2^{l_k}})^T$, each $\theta_{k,i}$ is generated in the form of $\theta_{k,i} = v_{ki1} + v_{ki2}v_{ki3}$, where v_{ki1} is uniformly drawn from $(0, 1)$, v_{ki3} uniformly from $(10, 1000)$ and binary variable v_{ki2} uniformly randomly drawn from $\{0, 1\}$. In the experiment, we vary n from 10 to 31 in intervals of 3, and generate 10 models for each n. For each model, we generate 1000 training data points from the ground-truth probability distribution (possibly overlapping) and see if learning algorithms can rediscover the exponential family models.

In learning exponential family models, we keep the structure of the exponential family model to be learned the same as the one that generates training data, and initialize all θ parameters to be the absolute values of samples drawn from a Gaussian distribution $\mathcal{N}(10, 10)$. Learning rate is fixed as 0.1 and parameters in XOR-Sampling are the same as in (Ermon et al., 2013b). For comparison, in addition to Gibbs-CD, we also compare with Belief Propagation equipped Contrastive Divergence (Ping & Ihler, 2017), denoted as BP-CD, and BPChain-CD (Fan & Xue, 2020). We allow the competing methods to draw 10000 samples from the model distribution while XOR-CD only draws 100 samples for a fair comparison (because it takes less time to draw samples using e.g., MCMC). When testing, we use ACE (Barton et al., 2016) to sample exactly from a target distribution. We implement XOR-CD using IBM ILOG CPLEX Optimizer 12.63 for queries to NP oracles. Experiments are carried out on a cluster, where each node has 24 cores and 96GB memory.

Likelihood Comparison Figure 3 shows the results of the four algorithms. The x-axis is exponential family model with different numbers of variables, and y-axis is the average log-likelihood of 1000 randomly generated samples from models learned by each of the learning algorithm. Here we use ACE (Barton et al., 2016) to compute the exact log-likelihood. We can see XOR-CD learns models that generate samples with higher average log-likelihood compared to competing approaches.

Time complexity We test the time complexity of different methods and find XOR-CD runs faster than Gibbs-CD. In particular, XOR-CD with 100 samples takes 1 minute 50 seconds per XOR iteration, while Gibbs-CD with 10,000 MCMC samples needs 2.5 minutes when learning models with dimension $n = 31$. Notice that XOR-CD outperforms Gibbs-CD in likelihood values also.

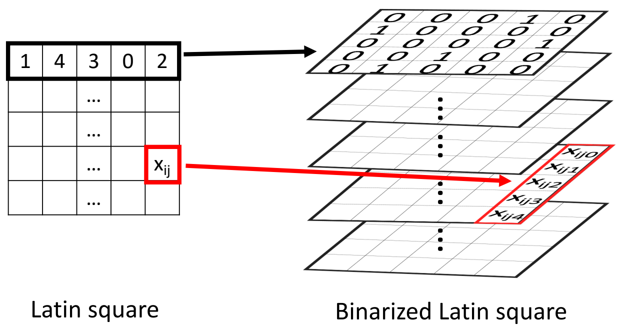


Figure 6. This figure shows how to binarize a Latin square. For each entry $x_{i,j} \in \{1, 2, ..., n\}$ in the $i-$th row and $j-$th column of a Latin square, we use n binary variables $x_{i,j,k} \in \{0, 1\}$ to represent its value where $k \in \{1, 2, ..., n\}$. If the value of $x_{i,j}$ is k, then only $x_{i,j,k}$ is equal to 1 and the other $n - 1$ variables are equal to 0.

C.2. Dispatching Route Generation

Here is additional information regarding the dispatching route generation experiment.

Models and experimental settings We learn an exponential family model to capture the likelihood of different permutations. Specifically, the exponential family model is $Pr(x) \propto \exp(\theta_0 + \sum_{i,j} \theta_{i,j} x_{i,j} + \sum_{i,j,k} \theta_{i,j,k} x_{i,j} x_{i+1,k})$ and the θ's are the parameters to learn. In generating training data, we assume the n locations are fully-connected, i.e., there is an edge between every two locations. Assuming the locations are labeled from 1 to n, we assume the distance between two locations is the difference between its two indices. For example, the distance between 1 and 3 is 2 (=3-1). The total travel distance d of a delivery route is the sum of the distances of each edge traveled in a trip. We randomly sample delivery routes according to the travel distance distribution shown in the rightmost column of the middle figure of Figure 4 to form the training dataset.

In learning the exponential family model, we initialize each model parameter (e.g., θ_0, $\theta_{i,j}$, $\theta_{i,j,k}$) to be the absolute value of samples drawn from a Gaussian distribution $\mathcal{N}(10, 10)$. For XOR-CD, we set $T = 500$ and a time limit of 10 hours. $M = K = 100$, and learning rate is 0.1. Parameters of XOR-Sampling are set the same as in Ermon et al. (2013b) in order to ensure $\delta = \sqrt{2}$. For Gibbs-CD, we let $K = 10000$ and the others are set the same as XOR-CD. As for the structure of GAN, the generator takes the input of random noise vector of dimension n (n is the number of locations), and has the structure $fc\{2n\} - fc\{5n\} - fc\{n^2\}$. Here, $fc\{.\}$ denotes a fully connected layer of output dimension $\{.\}$. For example, the first fully connected layer $fc\{2n\}$ maps an input of dimension n to an output of dimension $2n$. The output of the generator is of the shape n^2 which represent variables x_{ij}. The discriminator has a structure $fc\{n^2\} - fc\{5n\} - fc\{2n\} - fc\{2\} - softmax$. We use ReLU as the activation function between fc layers. Number of training epochs is 1000, and the wall-time limit is also 10 hours.

Hamilton Cycle Constraints In sampling from current model distribution, XOR-CD has to sample the assignments to variables $x_{i,j}$ which form valid Hamilton cycles; ie, the locations to visit in a route form a permutation. We include the following two constraints in the mixed integer program of XOR-sampling to enforce this contraint:

$$\sum_{j=1}^{n} x_{i,j} = 1, \quad \forall i \in \{1, ..., n\}; \tag{21}$$

$$\sum_{i=1}^{n} x_{i,j} = 1, \quad \forall j \in \{1, ..., n\}. \tag{22}$$

C.3. Optimal Experiment Design

Models and experimental settings Let $x_{i,j,k}$ be an indicator variable which is 1 if and only if crop k is planted at the i, j-th entry. The exponential family model is: $Pr(x) \propto \exp(\theta_0 + \sum_{i,j,k} \theta_{i,j,k} x_{i,j,k} + \sum_{i,j,m,l} \theta^1_{i,j,m,l} \phi(x_{i,j,m}, x_{i+1,j,l}) + \theta^2_{i,j,m,l} \phi(x_{i,j,m}, x_{i,j+1,l})$, in which $\theta_{i,j,k}, \theta^1_{i,j,m,l}, \theta^2_{i,j,m,l}$ are the parameters to learn. We define $d_i(j, k)$ as the distance between symbols j and k in row i. This distance is calculated as the absolute difference of the column indices of where symbols j and k appear in row i. The total distance $d(j, k)$ is defined as $d(j, k) = \sum_{i=1}^{n} d_i(j, k)$. The spatial variance of a Latin square is defined as the variance of all the total distances $d(i, j), \forall i \neq j$. This spatial variance is an important metric determining whether an experiment design is good (Gomes et al., 2004; Smith et al.; Le Bras et al., 2012). Notice that we can prove there are only three possible variances for a 5x5 Latin square. In generating training data, we randomly sample 5-by-5 Latin squares to form a training set, the distribution of the spatial variance of which is shown in the rightmost column of the right figure of Figure 4.

We set hyper-parameters of both XOR-CD and Gibbs-CD the same as in the dispatching route generation task. As for the structure of GAN, the generator takes the input of random noise vector of dimension n, and has the structure $fc\{2n\} - fc\{n^2\} - fc\{n^3/2\} - fc\{n^3\}$. The output of the generator is of the shape n^3 which represent the assignments to variables x_{ijk}. The discriminator has a structure $fc\{n^3/2\} - fc\{n^2\} - fc\{n\} - fc\{2\} - softmax$. Here fc denotes fully connected layer and the number denotes the dimension of output of this layer. We use ReLU as the activation function between each two fc layers. Number of epochs is 1000, and the wall-time limit is also 10 hours.

Latin Square Constraints We can enforce the following set of constraints to ensure the output forms a Latin square in XOR-CD:

$$\sum_{k=1}^{n} x_{i,j,k} = 1, \quad \forall i,j \in \{1,...,n\}; \tag{23}$$

$$\sum_{j=1}^{n} x_{i,j,k} = 1, \quad \forall i,k \in \{1,...,n\}; \tag{24}$$

$$\sum_{i=1}^{n} x_{i,j,k} = 1, \quad \forall j,k \in \{1,...,n\}. \tag{25}$$

Constraints (23) indicate each cell in the Latin square must be a valid integer between 1 and n. Constraints (24) indicate cells in each row must be different, and constraints (25) indicate cells in each column must be different.

C.4. Sequence-based Protein Homology Detection

As shown in Figure 1, the alignment matrix of sequence $\mathcal{S}_1$ of size N_1 and sequence $\mathcal{S}_2$ of size N_2 is of size $(N_1+1)\times(N_2+1)$. In the matrix, the rows represent the amino acids in $\mathcal{S}_1$ and the columns represent the ones in $\mathcal{S}_2$. Each alignment forms a path from the upper-left node to the bottom-right node as shown in the right panel of Figure 1, where each transition in the path is either horizontal, representing an insertion in $\mathcal{S}_2$, vertical, representing an insertion in $\mathcal{S}_1$, or diagonal, representing a match. We use symbol M, I_1 and I_2 to represent a match, an insertion in $\mathcal{S}_1$, and an insertion in $\mathcal{S}_2$, respectively. The (i,j)-th node in the alignment matrix is associated with three binary variables: $z^u_{i,j}$, where $u \in \{M, I_1, I_2\}$. $z^u_{i,j}$ is 1 if and only if the path passes the (i,j)-th node with type u. Let $A^p = \{z^u_{i,j} = z^{u(p)}_{i,j}, 1 \le i \le N_1+1, 1 \le j \le N_2+1, u \in \{M, I_1, I_2\}\}$ be one value assignment to all $z^u_{i,j}$ variables that form path p. Notice A^p also represents one alignment between sequence $\mathcal{S}_1$ and $\mathcal{S}_2$. Hence we will refer a path and an alignment interchangeably. Let $A = \{A^p \mid p \text{ is a valid path}\}$ be the set of all alignments. Our exponential family model formulation estimates the probability of having the alignment A^p to be:

$$P_\theta(A^p|\mathcal{S}_1,\mathcal{S}_2) = \frac{e^{\sum_{i,j,u}\theta^u_{i,j} z^{u(p)}_{i,j} + \sum_{i,j,u,k,l,v}\theta^{uv}_{i,j,k,l} z^{u(p)}_{i,j} z^{v(p)}_{k,l}}}{\mathcal{Z}(\mathcal{S}_1,\mathcal{S}_2)},$$

where $\mathcal{Z}(\mathcal{S}_1,\mathcal{S}_2) = \sum_{A^p\in A} e^{\sum \theta^u_{i,j} z^{u(p)}_{i,j} + \sum \theta^{uv}_{i,j,k,l} z^{u(p)}_{i,j} z^{v(p)}_{k,l}}$ is the normalization factor. Here, $\theta = \{\theta^u_{i,j}, \theta^{uv}_{i,j,k,l}\}$ are the set of variables to learn. In practice, we further parameterize $\theta^u_{i,j}$ to be $\theta^u_{i,j} = r^T \Phi_{i,j}(\mathcal{S}_{1i},\mathcal{S}_{2j},u)$, and parameterize $\theta^{uv}_{i,j,k,l}$ to be $\theta^{uv}_{i,j,k,l} = s^T \Xi_{i,j}(\mathcal{S}_{1i},\mathcal{S}_{2j},u,\mathcal{S}_{1k},\mathcal{S}_{2l},v)$, where both Φ and Ξ are features extracted from data, and we instead focus on learning parameters r and s. Features are extracted based on profile conservation, secondary structure, and solvent accessibility of each protein.

Learning. Given a training set of $(A^p_k, \mathcal{S}^k_{1,k}, \mathcal{S}_{2,k})_{k=1}^N$, where $\mathcal{S}_{1,k}, \mathcal{S}_{2,k}$ are a pair of sequences, and A^p_k is the observed alignment between the two sequences. We want to learn the exponential family model via maximizing the likelihood, which translates to the following problem:

$$\max_{r,s} \prod_{k=1}^{N} P_{r,s}(A^p_k|\mathcal{S}_{1,k},\mathcal{S}_{2,k}).$$

Inference. After learning $P(A^p|\mathcal{S}_1,\mathcal{S}_2)$, we can use the model to find the best alignment between two new sequences by solving the following linear programming problem:

$$\begin{aligned} A^{p*} &= \arg\max_{A^p\in A} P(A^p|\mathcal{S}_1,\mathcal{S}_2) \\ &= \arg\max_{A^p\in A} \sum_{i,j,u}\theta^u_{i,j} z^{u(p)}_{i,j} + \sum_{i,j,k,l,u,v}\theta^{uv}_{i,j,k,l} z^{u(p)}_{i,j} z^{v(p)}_{k,l}. \end{aligned}$$

Sequence Alignment Constraints In the task of sequence alignment, each alignment must be a valid path in the alignment matrix as shown in Figure 1. Consider the alignment matrix of size $(N_1+1)\times(N_2+1)$, and $3N_1N_2$ binary variables $z^u_{i,j}$, where $1 \le i \le N_1+1, 1 \le j \le N_2+1$, and $u \in \{M, I_1, I_2\}$. If we consider the alignment matrix as a directed graph and each node (i,j) has three income edges, i.e., $z^{I_2}_{ij}$ denotes the edge from node $(i, j-1)$, $z^{I_1}_{ij}$ denotes the edge from node $(i-1, j)$ and z^M_{ij} denotes the edge from node $(i-1, j-1)$. $z^u_{i,j} = 1$ means the corresponding edge exists and $z^u_{i,j} = 0$ otherwise. Then, to form a valid path (alignment), the following set of constraints $\mathcal{C}$ must be satisfied:

$$z^M_{1,1} = 1, z^{I_1}_{1,1} = 0, z^{I_2}_{1,1} = 0; \tag{26}$$

$$\sum_u z^u_{N_1+1,N_2+1} = 1; \tag{27}$$

$$for \quad \forall i \in \{1,...,N_1\}, j \in \{1,...,N_2\}:$$

$$\sum_u z^u_{i,j} - z^{I_1}_{i+1,j} - z^{I_2}_{i,j+1} - z^M_{i+1,j+1} = 0; \tag{28}$$

$$for \quad \forall i = N_1+1, j \in \{1,...,N_2\}:$$

$$\sum_u z^u_{i,j} - z^{I_2}_{i,j+1} = 0; \tag{29}$$

$$for \quad \forall i \in \{1,...,N_1\}, j = N_2+1,:$$

$$\sum_u z^u_{i,j} - z^{I_1}_{i+1,j} = 0; \tag{30}$$

$$for \quad \forall i = 1, j \in \{2,...,N_2+1\}:$$

$$z^M_{i,j} + z^{I_1}_{i,j} = 0; \tag{31}$$

$$for \quad \forall i \in \{2,...,N_1+1\}, j = 1,:$$

$$z^M_{i,j} + z^{I_2}_{i,j} = 0. \tag{32}$$

Experiment Setup. To run XOR-CD in this experiment, we set $T = 500$ and force each query of NP oracle in XOR-CD to stop in 15 minutes. As a result, not all the queries to

NP oracles are solved up to optimality. We also enforce a timeout of 10 hours for all algorithms. The learning rate is 0.1 for the first 100 epochs and 0.01 for the next 400 epochs for both XOR-CD and Gibbs-CD. Both M and K are set to 100 in XOR-CD and parameters in XOR-Sampling are set the same as in (Ermon et al., 2013b). For Gibbs-CD, we change K to 10000 for a fair comparison, which takes approximately the same amount of time compared with XOR-CD for each SGD iteration.