Learning with Local Search MCMC Layers

Anonymous Author(s)
Affiliation
Address
email

Abstract

Integrating combinatorial optimization layers into neural networks has recently attracted significant research interest. However, many existing approaches lack theoretical guarantees or fail to perform adequately when relying on inexact solvers. This is a critical limitation, as many operations research problems are NP-hard, often necessitating the use of neighborhood-based local search heuristics. These heuristics iteratively generate and evaluate candidate solutions based on an acceptance rule. In this paper, we introduce a theoretically-principled approach for learning with such inexact combinatorial solvers. Inspired by the connection between simulated annealing and Metropolis-Hastings, we propose to transform problem-specific neighborhood systems used in local search heuristics into proposal distributions, implementing MCMC on the combinatorial space of feasible solutions. This allows us to construct differentiable combinatorial layers and associated loss functions. Replacing an exact solver by a local search strongly reduces the computational burden of learning on many applications. We demonstrate our approach on a large-scale dynamic vehicle routing problem with time windows.

1 Introduction

Models that combine neural networks and combinatorial optimization have recently attracted significant attention [14, 39, 8, 6, 50, 5, 34, 43, 7]. Such models enable the transformation of learned continuous latent representations into structured discrete outputs that satisfy complex constraints. They enrich combinatorial optimization algorithms by providing them with context-dependent features, making decisions more resilient to uncertainty. An important subset of this line of research involves integrating, within a neural network, a linear programming layer of the form:

$$\boldsymbol{\theta} \mapsto \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{\theta}, \boldsymbol{y} \rangle \subseteq \operatorname*{argmax}_{\boldsymbol{y} \in \mathsf{conv}(\mathcal{Y})} \langle \boldsymbol{\theta}, \boldsymbol{y} \rangle, \tag{1}$$

where $\mathcal{Y}$ is a finite set of feasible outputs. In the graphical model and structured prediction literature, this is often called the maximum a posteriori (MAP) problem [51]. The main challenge in using such layers lies in their end-to-end model training. Indeed, as piecewise-constant, discontinuous functions, such layers break the differentiable programming computational graph, and prevent one from backpropagating meaningful gradients from the final output of the model to its parameters.

Many approaches have been proposed to derive relaxations and loss functions for this setting; see Appendix A for a detailed review and overview of relevant related work. To methodologically position our work, Table 1 provides a high-level overview of foundational approaches, contrasting them based on the type of oracle they assume access to. Some rely on an oracle for solving a regularized version of Eq. (1), such as a quadratic or entropy-regularized program. They typically perform a single oracle call per data point. Some other approaches assume access to an oracle for solving the original linear program (i.e., a MAP oracle), but perform multiple oracle calls, for smoothing reasons. Their theoretical guarantees usually assume an oracle returning exact solutions.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

Table 1: The proposed approach leverages the neighborhood systems used by local search heuristics (inexact solvers) to obtain a differentiable combinatorial layer when usual oracles are not available.

	Regularization	Oracle	Approach
Differentiable DP (2009, 2018)	Entropy	Exact marginal	DP
SparseMAP (2018)	Quadratic	Exact MAP	Frank-Wolfe
Barrier FW (2015)	TRW Entropy	Exact MAP	Frank-Wolfe
IntOpt (2020)	Log barrier	Interior point solver	Primal-Dual
Perturbed optimizers (2020)	Implicit via noise	Exact MAP	Monte-Carlo
Blackbox solvers (2020)	None	Exact MAP	Interpolation
Contrastive divergences (2000)	Entropy	Gibbs / Langevin sampler	MCMC
Proposed	Entropy	Local search	MCMC

Unfortunately, many problems in operations research are NP-hard in nature (e.g., routing, scheduling, network design), making access to an exact oracle difficult. In contrast, operations research applications often rely on local search heuristics, such as simulated annealing. These heuristics iteratively generate a neighbor of the current solution, and either accept it or reject it based on an acceptance rule. The aim of this work is to provide a theoretically-principled approach for learning with such inexact solvers. To do so, we propose to leverage unexploited links between neighborhood-based, local search heuristics used to approximately solve combinatorial problems, and Markov chain Monte-Carlo (MCMC) methods used to perform approximate marginal inference in graphical models.

Contributions. (i) We enable the integration of local search heuristics as layers into ML models, by converting their neighborhood systems into proposal distributions for a discrete MCMC sampler over the combinatorial set of solutions. (ii) We extend our framework to handle local search heuristics that leverage a diversity of neighborhood systems, enabling this powerful class of solvers to be used as a unified MCMC sampler. (iii) We show that there exist Fenchel-Young losses [8] whose stochastic gradients are given by the proposed layer (even with a single MCMC iteration), leading to principled learning algorithms in both supervised and unsupervised settings, for which we provide a convergence analysis. (iv) We demonstrate our approach on the EURO Meets NeurIPS 2022 challenge [27], a large-scale, ML-enriched dynamic vehicle routing problem with time windows (DVRPTW), which involves an intractable combinatorial optimization problem. In Appendix B, we also empirically validate the quality of the proposed gradient estimators through abundant experiments.

Problem setup. In this paper, our goal is to learn models with an optimization layer of the form

$$\widehat{\boldsymbol{y}} : \boldsymbol{\theta} \mapsto \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{\theta}, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y}), \tag{2}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}^d$ is a finite but combinatorially-large set. This formulation is a generalization of the standard linear objective in Eq. (1). The function $\varphi : \mathcal{Y} \to \mathbb{R}$ is an integral part of the problem definition, capturing any structural costs or preferences (e.g., routing distances, fixed costs) that are independent of $\boldsymbol{\theta}$. We focus on settings where this optimization problem is intractable and only heuristic algorithms are available to obtain an approximate solution. We distinguish between two settings. In the unsupervised setting, our goal will be to learn $\boldsymbol{\theta} \in \mathbb{R}^d$ from observations $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N \in \mathbb{R}^d$. In the supervised setting, we will assume that $\boldsymbol{\theta} = g_W(\boldsymbol{x})$ and our goal will be to learn the parameters W from observation pairs $(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_N, \boldsymbol{y}_N)$.

2 Local search based MCMC layers

In this section, we show how to design principled combinatorial layers without relying on exact MAP solvers, by transforming local search heuristics into MCMC algorithms.

2.1 From local search to MCMC

Local search and neighborhood systems. Local search heuristics [19] iteratively generate a neighbor $\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}^{(k)})$ of the current solution $\boldsymbol{y}^{(k)}$, and either accept it or reject it based on an acceptance rule. In this context, a neighborhood system $\mathcal{N}$ defines, for each feasible solution $\boldsymbol{y} \in \mathcal{Y}$, a set of neighbors $\mathcal{N}(\boldsymbol{y}) \subseteq \mathcal{Y}$. Neighborhoods are problem-specific, and must respect the structure of the problem, i.e., must maintain solution feasibility. They are typically defined implicitly via a set of allowed *moves* from $\boldsymbol{y}$. For instance, Table 4 lists example moves for a vehicle routing problem.

Algorithm 1 SA / MH as a layer

Inputs: $\boldsymbol{\theta} \in \mathbb{R}^d$, $\boldsymbol{y}^{(0)} \in \mathcal{Y}$, (t_k), $K \in \mathbb{N}$, $\mathcal{N}$, q
for $k = 0 : K$ **do**
  Sample a neighbor in $\mathcal{N}(\boldsymbol{y}^{(k)})$:
  $\boldsymbol{y}' \sim q\left(\boldsymbol{y}^{(k)}, \cdot\right)$
  $\alpha(\boldsymbol{y}^{(k)}, \boldsymbol{y}') \leftarrow 1$ (SA) or
  $\alpha(\boldsymbol{y}^{(k)}, \boldsymbol{y}') \leftarrow \frac{q(\boldsymbol{y}', \boldsymbol{y}^{(k)})}{q(\boldsymbol{y}^{(k)}, \boldsymbol{y}')}$ (MH)
  $U \sim \mathcal{U}([0, 1])$
  $\Delta^{(k)} \leftarrow \langle \boldsymbol{\theta}, \boldsymbol{y}' \rangle + \varphi(\boldsymbol{y}') - \langle \boldsymbol{\theta}, \boldsymbol{y}^{(k)} \rangle - \varphi(\boldsymbol{y}^{(k)})$
  $p^{(k)} \leftarrow \alpha(\boldsymbol{y}^{(k)}, \boldsymbol{y}') \exp\left(\Delta^{(k)} / t_k\right)$
  If $U \leq p^{(k)}$, accept move: $\boldsymbol{y}^{(k+1)} \leftarrow \boldsymbol{y}'$
  If $U > p^{(k)}$, reject move: $\boldsymbol{y}^{(k+1)} \leftarrow \boldsymbol{y}^{(k)}$
end for
Output: $\widehat{\boldsymbol{y}}(\boldsymbol{\theta}) \approx \boldsymbol{y}^{(K)}$ (SA) or
$\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \mathbb{E}_{\pi_{\boldsymbol{\theta}, t}}[Y] \approx \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{y}^{(k)}$ (MH)

Algorithm 2 Neighborhood mixture MCMC

Inputs: $\boldsymbol{\theta} \in \mathbb{R}^d$, $\boldsymbol{y}^{(0)} \in \mathcal{Y}$, t, $K \in \mathbb{N}$, $(\mathcal{N}_s, q_s)_{s=1}^{S}$
for $k = 0 : K$ **do**
  Sample a neighborhood system:
  $s \sim \mathcal{U}(Q(\boldsymbol{y}^{(k)}))$
  Sample a neighbor in $\mathcal{N}_s(\boldsymbol{y}^{(k)})$:
  $\boldsymbol{y}' \sim q_s(\boldsymbol{y}^{(k)}, \cdot)$
  $\alpha_s(\boldsymbol{y}^{(k)}, \boldsymbol{y}') \leftarrow \frac{|Q(\boldsymbol{y}^{(k)})|}{|Q(\boldsymbol{y}')|} \frac{q_s(\boldsymbol{y}', \boldsymbol{y}^{(k)})}{q_s(\boldsymbol{y}^{(k)}, \boldsymbol{y}')}$
  $U \sim \mathcal{U}([0, 1])$
  $\Delta^{(k)} \leftarrow \langle \boldsymbol{\theta}, \boldsymbol{y}' \rangle + \varphi(\boldsymbol{y}') - \langle \boldsymbol{\theta}, \boldsymbol{y}^{(k)} \rangle - \varphi(\boldsymbol{y}^{(k)})$
  $p^{(k)} \leftarrow \alpha_s(\boldsymbol{y}^{(k)}, \boldsymbol{y}') \exp\left(\Delta^{(k)} / t\right)$
  If $U \leq p^{(k)}$, accept move: $\boldsymbol{y}^{(k+1)} \leftarrow \boldsymbol{y}'$
  If $U > p^{(k)}$, reject move: $\boldsymbol{y}^{(k+1)} \leftarrow \boldsymbol{y}^{(k)}$
end for
Output: $\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \mathbb{E}_{\pi_{\boldsymbol{\theta}, t}}[Y] \approx \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{y}^{(k)}$

Formally, we denote the neighborhood graph by $G_{\mathcal{N}} := (\mathcal{Y}, E_{\mathcal{N}})$, where edges are defined by $\mathcal{N}$. We assume the graph is undirected, i.e., $\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})$ if and only if $\boldsymbol{y} \in \mathcal{N}(\boldsymbol{y}')$, and without self-loops – i.e., $\boldsymbol{y} \notin \mathcal{N}(\boldsymbol{y})$. A stochastic neighbor generating function is also provided, in the form of a proposal distribution $q(\boldsymbol{y}, \cdot)$ with support either equal to $\mathcal{N}(\boldsymbol{y})$ or $\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}$.

Link between simulated annealing and Metropolis-Hastings. A well-known example of local search heuristic is simulated annealing (SA) [26]. It is intimately related to Metropolis-Hastings (MH) [21], an instance of a MCMC algorithm. We provide a unified view of both in Algorithm 1.

The difference lies in the acceptance rule, which incorporates a proposal correction ratio for MH, and in the choice of the sequence of temperatures $(t_k)_{k \in \mathbb{N}}$. In the case of SA, it is chosen to verify $t_k \to 0$. In the case of MH, it is such that $t_k \equiv t$. In this case, the iterates $\boldsymbol{y}^{(k)}$ of Algorithm 1 follow a time-homogenous Markov chain on $\mathcal{Y}$, defined by the following transition kernel:

$$P_{\boldsymbol{\theta}, t}(\boldsymbol{y}, \boldsymbol{y}') = \begin{cases} q(\boldsymbol{y}, \boldsymbol{y}') \min\left[1, \frac{q(\boldsymbol{y}', \boldsymbol{y})}{q(\boldsymbol{y}, \boldsymbol{y}')} \exp\left(\frac{\langle \boldsymbol{\theta}, \boldsymbol{y}' \rangle + \varphi(\boldsymbol{y}') - \langle \boldsymbol{\theta}, \boldsymbol{y} \rangle - \varphi(\boldsymbol{y})}{t}\right)\right] & \text{if } \boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}), \\ 1 - \sum_{\boldsymbol{y}'' \in \mathcal{N}(\boldsymbol{y})} P_{\boldsymbol{\theta}, t_k}(\boldsymbol{y}, \boldsymbol{y}'') & \text{if } \boldsymbol{y}' = \boldsymbol{y}, \\ 0 & \text{else.} \end{cases} \tag{3}$$

In past work, the link between the two algorithms has primarily been used to show that SA converges to the exact MAP solution in the limit of infinite iterations [36, 17]. Under mild conditions – if the neighborhood graph $G_{\mathcal{N}}$ is connected and the chain is aperiodic – the iterates $\boldsymbol{y}^{(k)}$ of Algorithm 1 (MH case) converge in distribution to the Gibbs distribution (see Appendix E.1 for a proof):

$$\pi_{\boldsymbol{\theta}, t}(\boldsymbol{y}) \propto \exp\left(\left[\langle \boldsymbol{\theta}, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y})\right] / t\right). \tag{4}$$

Proposed layer. Algorithm 1 and this result motivate us to define the combinatorial MCMC layer

$$\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) := \mathbb{E}_{\pi_{\boldsymbol{\theta}, t}}[Y], \tag{5}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ are logits and $t > 0$ is a temperature parameter, defaulting to $t = 1$. Naturally, the estimate of $\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta})$ returned by Algorithm 1 (MH case) is biased, as the Markov chain cannot perfectly mix in a finite number of iterations, except if it is initialized at $\pi_{\boldsymbol{\theta}, t}$. In Section 3, we will show that this does not hinder the convergence of the proposed learning algorithms. The next proposition, proved in Appendix E.2, states some useful properties of the proposed layer.

Proposition 2.1. *Let $\boldsymbol{\theta} \in \mathbb{R}^d$. We have the following properties:*

$$\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) \in \mathsf{relint}(\mathcal{C}), \quad \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) \xrightarrow[t \to 0^+]{} \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{\theta}, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y}), \quad \textit{and} \quad \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) \xrightarrow[t \to \infty]{} \frac{1}{|\mathcal{Y}|} \sum_{\boldsymbol{y} \in \mathcal{Y}} \boldsymbol{y}.$$

Moreover, $\widehat{\boldsymbol{y}}_t$ is differentiable and its Jacobian matrix is given by $J_{\boldsymbol{\theta}} \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \frac{1}{t} \mathsf{cov}_{\pi_{\boldsymbol{\theta}, t}}[Y]$.

2.2 Mixing neighborhood systems

Central to local search algorithms in combinatorial optimization is the use of multiple neighborhood systems to more effectively explore the solution space [37, 10]. We now present a tractable way to incorporate such diversity of neighborhood systems into the proposed layer, while preserving the correct stationary distribution, by mixing the corresponding proposal distributions. A discussion giving intuition on why the proposed method is crucial to get tractable updates is given in Appendix C.1.

Definitions. Let $(\mathcal{N}_s)_{s=1}^S$ be a set of different neighborhood systems. Typically, all neighborhood systems are not defined on all solutions $\boldsymbol{y} \in \mathcal{Y}$, so we note $Q(\boldsymbol{y}) \subseteq [\![1, S]\!]$ the set of neighborhood systems defined on $\boldsymbol{y}$ (i.e., the set of allowed moves on $\boldsymbol{y}$). Let $(q_s)_{s \in Q(\boldsymbol{y})}$ be the corresponding proposal distributions, such that the support of $q_s(\boldsymbol{y}, \cdot)$ is either $\mathcal{N}_s(\boldsymbol{y})$ or $\mathcal{N}_s(\boldsymbol{y}) \cup \{\boldsymbol{y}\}$. Let $\bar{\mathcal{N}}$ be the aggregate neighborhood system defined by $\bar{\mathcal{N}} : \boldsymbol{y} \mapsto \bigcup_{s \in Q(\boldsymbol{y})} \mathcal{N}_s(\boldsymbol{y})$.

We assume that the individual Metropolis correction ratios $\tilde{\alpha}_s(\boldsymbol{y}, \boldsymbol{y}') \coloneqq \frac{q_s(\boldsymbol{y}', \boldsymbol{y})}{q_s(\boldsymbol{y}, \boldsymbol{y}')}$ are tractable. The proposed procedure is summarized in Algorithm 2.

Proposition 2.2. *If each neighborhood graph $G_{\mathcal{N}_s}$ is undirected and without self-loops, and the aggregate neighborhood graph $G_{\bar{\mathcal{N}}}$ is connected, the iterations $\boldsymbol{y}^{(k)}$ produced by Algorithm 2 follow a Markov chain with unique stationary distribution $\pi_{\boldsymbol{\theta},t}$.*

See Appendix E.3 for the proof. Importantly, only the connectedness of the aggregate neighborhood graph $G_{\bar{\mathcal{N}}}$ is required. This allows us to combine neighborhood systems that could not connect $\mathcal{Y}$ if used individually, i.e., an irreducible Markov chain can be obtained by mixing the proposal distributions of reducible ones. Such an example is given with the moves defined in Table 4.

3 Loss functions and theoretical analysis

We now derive and study loss functions for learning models using the proposed layer. The analysis for the case where only one iteration of MCMC is performed ($K = 1$) is given in Appendix C.2.

3.1 Negative log-likelihood and associated Fenchel-Young loss

We now show that the proposed layer $\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta})$ can be viewed as the solution of a regularized optimization problem on $\mathcal{C} = \mathsf{conv}(\mathcal{Y})$. Let $A_t(\boldsymbol{\theta}) \coloneqq t \cdot \log \sum_{\boldsymbol{y} \in \mathcal{Y}} \exp\left([\langle \boldsymbol{\theta}\,,\, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y})]\,/t\right)$ be the cumulant function [51] associated to the exponential family defined by $\pi_{\boldsymbol{\theta},t}$, scaled by the temperature t. We define the regularization function Ω_t and the corresponding Fenchel-Young loss [8] as:

$$\Omega_t(\boldsymbol{\mu}) \coloneqq A_t^*(\boldsymbol{\mu}) = \sup_{\boldsymbol{\theta} \in \mathbb{R}^d} \langle \boldsymbol{\mu}\,,\, \boldsymbol{\theta} \rangle - A_t(\boldsymbol{\theta}), \quad \text{and} \quad \ell_t(\boldsymbol{\theta}\,;\boldsymbol{y}) \coloneqq (\Omega_t)^*(\boldsymbol{\theta}) + \Omega_t(\boldsymbol{y}) - \langle \boldsymbol{\theta},\, \boldsymbol{y} \rangle.$$

Since $\Omega_t = A_t^*$ is strictly convex on $\mathsf{relint}(\mathcal{C})$ (see Appendix E.4 for a proof) and $\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} A_t(\boldsymbol{\theta})$, the proposed layer is the solution of the regularized optimization problem

$$\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \operatorname*{\mathsf{argmax}}_{\boldsymbol{\mu} \in \mathcal{C}} \left\{ \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - \Omega_t(\boldsymbol{\mu}) \right\}, \tag{6}$$

the Fenchel-Young loss ℓ_t is differentiable, satisfies $\ell_t(\boldsymbol{\theta}, \boldsymbol{y}) = 0 \Leftrightarrow \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \boldsymbol{y}$, and has gradient $\nabla_{\boldsymbol{\theta}} \ell_t(\boldsymbol{\theta}\,;\boldsymbol{y}) = \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) - \boldsymbol{y}$ [8]. It is therefore equivalent, up to a constant, to the negative log-likelihood loss, as we have $-\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta},t}(\boldsymbol{y}) = (\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) - \boldsymbol{y})/t$. Algorithms 1 and 2 can thus be used to perform maximum likelihood estimation, by returning a (biased) stochastic estimate of the gradient of ℓ_t.

3.2 Empirical risk minimization

In the supervised learning setting, we are given observations $(\boldsymbol{x}_i,\, \boldsymbol{y}_i)_{i=1}^N \in (\mathbb{R}^p \times \mathcal{Y})^N$, and want to fit a model $g_W : \mathbb{R}^p \to \mathbb{R}^d$ such that $\widehat{\boldsymbol{y}}_t(g_W(\boldsymbol{x}_i)) \approx \boldsymbol{y}_i$. This is motivated by a generative model where, for some weights $W_0 \in \mathbb{R}^p$, the data is generated with $\boldsymbol{y}_i \sim \pi_{g_{W_0}(\boldsymbol{x}_i),t}$. We aim at minimizing the empirical risk L_N, defined below along with its exact gradient ∇L_N :

$$L_N(W) \coloneqq \frac{1}{N} \sum_{i=1}^N \ell_t\left(g_W(\boldsymbol{x}_i); \boldsymbol{y}_i\right) \quad \text{and} \quad \nabla_W L_N(W) = \frac{1}{N} \sum_{i=1}^N J_W g_W(\boldsymbol{x}_i)\left(\widehat{\boldsymbol{y}}_t(g_W(\boldsymbol{x}_i)) - \boldsymbol{y}_i\right).$$

Doubly stochastic gradient estimator. In practice, we cannot compute the exact gradient above. Using Algorithm 1 to get a MCMC estimate of $\widehat{\boldsymbol{y}}_t(g_W(\boldsymbol{x}_i))$, we propose the following estimator:

$$\nabla_W L_N(W) \approx J_W g_W(\boldsymbol{x}_i) \left(\frac{1}{K} \sum_{k=1}^{K} \boldsymbol{y}_i^{(k)} - \boldsymbol{y}_i \right),$$

where $\boldsymbol{y}_i^{(k)}$ is the k-th iterate of Algorithm 1 with maximization direction $\boldsymbol{\theta}_i = g_W(\boldsymbol{x}_i)$ and temperature t. This estimator is doubly stochastic, since we sample both data points and iterations of Algorithm 1, and can be seamlessly used with batches. The vector-jacobian product with $J_W g_W(\boldsymbol{x}_i)$ is computed via autodiff. The Markov chain initialization methods for the supervised and unsupervised settings are inspired from the contrastive divergence literature [22] and detailed in Appendix C.5.

3.3 Convergence analysis in the unsupervised setting

In the unsupervised setting, we are only given observations $(\boldsymbol{y}_i)_{i=1}^N \in \mathcal{Y}^N$ and want to fit a model $\pi_{\boldsymbol{\theta},t}$, motivated by an underlying generative model such that $\boldsymbol{y}_i \sim \pi_{\boldsymbol{\theta}_0,t}$ for an unknown true parameter $\boldsymbol{\theta}_0$. We assume here that $\mathcal{C} = \mathsf{conv}(\mathcal{Y})$ is of full dimension in $\mathbb{R}^d$. We have the following empirical L_N and population $L_{\boldsymbol{\theta}_0}$ Fenchel-Young losses:

$$L_N(\boldsymbol{\theta}; \boldsymbol{y}_1, \dots, \boldsymbol{y}_N) \coloneqq \frac{1}{N} \sum_{i=1}^{N} \ell_t(\boldsymbol{\theta}; \boldsymbol{y}_i) \quad , \quad L_{\boldsymbol{\theta}_0}(\boldsymbol{\theta}) \coloneqq \mathbb{E}_{(\boldsymbol{y}_i)_{i=1}^N \sim (\pi_{\boldsymbol{\theta}_0,t})^{\otimes N}} \left[L_N(\boldsymbol{\theta}; \boldsymbol{y}_1, \dots, \boldsymbol{y}_N) \right],$$

which are minimized for $\boldsymbol{\theta}$ such that $\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \bar{Y}_N \coloneqq \frac{1}{N} \sum_{i=1}^N \boldsymbol{y}_i$, and for $\boldsymbol{\theta}$ such that $\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}_0)$, respectively. Let $\boldsymbol{\theta}_N^\star$ as the minimizer of the empirical loss L_N. For it to be defined, we assume N is large enough to have $\bar{Y}_N \in \mathsf{relint}(\mathcal{C})$ (which is always possible as $\pi_{\boldsymbol{\theta}_0,t}$ has dense support on $\mathcal{Y}$). A slight variation on Proposition 4.1 in Berthet et al. [6], proved in Appendix E.5, gives the following asymptotic normality as $N \to \infty$.

Proposition 3.1 (Convergence of the empirical loss minmizer to the true parameter)**.**

$$\sqrt{N}(\boldsymbol{\theta}_N^\star - \boldsymbol{\theta}_0) \xrightarrow[N \to \infty]{\mathcal{D}} \mathcal{N}\left(\mathbf{0}, \, t^2 \, \mathsf{cov}_{\pi_{\boldsymbol{\theta}_0,t}}[Y]^{-1} \right).$$

We now consider the sample size as fixed to N samples, and define $\hat{\boldsymbol{\theta}}_n$ as the n-th iterate of the following stochastic gradient algorithm:

$$\hat{\boldsymbol{\theta}}_{n+1} = \hat{\boldsymbol{\theta}}_n + \gamma_{n+1} \left[\bar{Y}_N - \frac{1}{K_{n+1}} \sum_{k=1}^{K_{n+1}} \boldsymbol{y}^{(n+1,\,k)} \right], \tag{7}$$

where $\boldsymbol{y}^{(n+1,k)}$ is the k-th iterate of Algorithm 1 with temperature t, maximization direction $\hat{\boldsymbol{\theta}}_n$, and initialized at $\boldsymbol{y}^{(n+1,1)} = \boldsymbol{y}^{(n,K_n)}$. This initialization corresponds to the persistent contrastive divergences (PCD) algorithm [48], and is further discussed in Appendix C.5.

Proposition 3.2 (Convergence of the stochastic gradient estimate)**.** *Suppose the following assumptions on the step sizes $(\gamma_n)_{n\geq 1}$, sample sizes $(K_n)_{n\geq 1}$, and proposal distribution q hold:*

- $\gamma_n = an^{-b}$, *with* $b \in \left]\frac{1}{2}, \, 1\right]$ *and* $a > 0$,
- $K_{n+1} > \left\lfloor 1 + a' \exp\left(\frac{8R_{\mathcal{C}}}{t} \cdot ||\hat{\boldsymbol{\theta}}_n|| \right) \right\rfloor$ *with* $a' > 0$ *and* $R_{\mathcal{C}} = \max_{\boldsymbol{y} \in \mathcal{Y}} ||\boldsymbol{y}||$,
- $\frac{1}{\sqrt{K_n}} - \frac{1}{\sqrt{K_{n-1}}} \leq a'' n^{-c}$, *with* $a'' > 0$ *and* $c > 1 - \frac{b}{2}$,
- $q(\boldsymbol{y}, \boldsymbol{y}') = \begin{cases} \frac{1}{2d^*} & \text{if } \boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}), \\ 1 - \frac{d(\boldsymbol{y})}{2d^*} & \text{if } \boldsymbol{y}' = \boldsymbol{y}, \\ 0 & \text{else}, \end{cases}$

  where $d(\boldsymbol{y}) \coloneqq |\mathcal{N}(\boldsymbol{y})|$ *is the degree of* $\boldsymbol{y}$ *in* $G_{\mathcal{N}}$, *and* $d^* \coloneqq \max_{\boldsymbol{y} \in \mathcal{Y}} d(\boldsymbol{y})$.

Then, we have the almost sure convergence $\hat{\boldsymbol{\theta}}_n \xrightarrow{a.s.} \boldsymbol{\theta}_N^\star$ *of the iterates* $\hat{\boldsymbol{\theta}}_n$ *defined by Eq. (7).*

See Appendix E.6 for the proof. The assumptions on q are used for obtaining a closed-form convergence rate bound for the Markov chain, using graph-based geometric bounds [23].

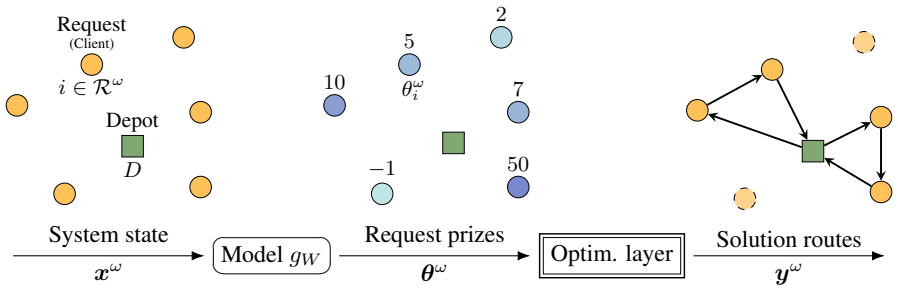


Figure 1: Overview of the vehicle routing pipeline, represented at request wave ω.

4 Experiments on dynamic vehicle routing

We demonstrate our approach on the EURO Meets NeurIPS 2022 Vehicle Routing Competition, a large-scale, ML-enriched dynamic vehicle routing problem with time windows (DVRPTW). A detailed introduction to the challenge with precise notations, together with precise explications on the reduction to supervised learning, the proposed approach, the perturbation-based baseline, full experimental details and additional results, are given in Appendix D.

Approach. We adopt the winning strategy of Baty et al. [4], which reduces the problem to a supervised learning task. This involves decomposing the DVRPTW as multiple prize-collecting problems (PC-VRPTW) for each wave ω, where a model g_W predicts a prize vector $\boldsymbol{\theta}^\omega$ for serving each request. This fits our general problem formulation $\widehat{\boldsymbol{y}}(\boldsymbol{\theta}^\omega) = \mathsf{argmax}_{\boldsymbol{y}} \langle \boldsymbol{\theta}^\omega, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y})$, where $\varphi(\boldsymbol{y})$ represents the negative routing cost. The overall pipeline is shown in Fig. 1. To train the model g_W, we use the Fenchel-Young loss associated with our proposed MCMC layer. The proposal distributions for the MCMC sampler are derived from the local search moves used by the state-of-the-art PC-HGS solver $\widetilde{\boldsymbol{y}}$, which are summarized in Table 4. At inference, we use the trained model g_W with the PC-HGS solver, forming the policy $f_W := \widetilde{\boldsymbol{y}} \circ g_W$. We compare our learning algorithms to the perturbation-based baseline from Baty et al. [4], which sacrifices the theoretical guarantees of the general framework from Berthet et al. [6] it instantiates by using an inexact solver ($\widetilde{\boldsymbol{y}}$).

Results. We evaluate performance using the competition's metric: the routing cost relative to an anticipative (oracle) baseline. Fig. 2 shows that initializing the MCMC chain with the ground-truth solution significantly outperforms a random start and that performance improves with more MCMC iterations (K). Table 2 compares our method with the baseline under a fixed time budget for the layer's forward pass. Our approach significantly outperforms the perturbation-based method in low-time-limit regimes (1-100 ms), enabling faster and more efficient training.

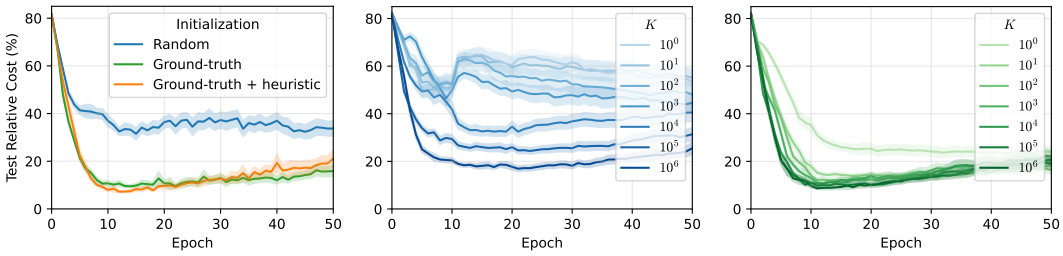


Figure 2: Test relative cost (%). **Left**: varying initialization method. **Center**: varying number of Markov iterations K with random initialization. **Right**: varying K with ground-truth initialization.

Table 2: Best test relative cost (%) for different training methods and time limits.

Time limit (ms)	**1**	**5**	**10**	**50**	**100**	**1000**
Perturbed inexact oracle	65.2 ± 5.8	13.1 ± 3.4	8.7 ± 1.9	6.5 ± 1.1	6.3 ± 0.76	$\mathbf{5.5} \pm 0.4$
Proposed ($\boldsymbol{y}^{(0)} = \boldsymbol{y}$)	10.0 ± 1.7	12.0 ± 2.6	11.8 ± 2.8	9.1 ± 1.7	8.4 ± 1.7	7.7 ± 1.1
Proposed ($\boldsymbol{y}^{(0)} = \boldsymbol{y}$+heuristic)	$\mathbf{7.8} \pm 0.8$	$\mathbf{7.2} \pm 0.6$	$\mathbf{6.3} \pm 0.7$	$\mathbf{6.2} \pm 0.8$	$\mathbf{5.9} \pm 0.7$	5.9 ± 0.6

References

[1] Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and Zico Kolter. Differentiable convex optimization layers, 2019. URL https://arxiv.org/abs/1910.12430.

[2] Kareem Ahmed, Zhe Zeng, Mathias Niepert, and Guy Van den Broeck. SIMPLE: A gradient estimator for k-subset sampling, 2024. URL http://arxiv.org/abs/2210.01941.

[3] Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pages 136–145. PMLR, 2017.

[4] Léo Baty, Kai Jungel, Patrick S. Klein, Axel Parmentier, and Maximilian Schiffer. Combinatorial optimization enriched machine learning to solve the dynamic vehicle routing problem with time windows, 2023. URL http://arxiv.org/abs/2304.00789.

[5] Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon, 2020. URL http://arxiv.org/abs/1811.06128.

[6] Quentin Berthet, Mathieu Blondel, Olivier Teboul, Marco Cuturi, Jean-Philippe Vert, and Francis Bach. Learning with differentiable perturbed optimizers, 2020. URL http://arxiv.org/abs/2002.08676.

[7] Mathieu Blondel and Vincent Roulet. The Elements of Differentiable Programming. *arXiv preprint arXiv:2403.14606*, 2024.

[8] Mathieu Blondel, André F. T. Martins, and Vlad Niculae. Learning with fenchel-young losses, 2020. URL http://arxiv.org/abs/1901.02324.

[9] Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation. *Advances in neural information processing systems*, 35:5230–5242, 2022.

[10] Christian Blum and Andrea Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. 35(3):268–308, 2003. ISSN 0360-0300. doi: 10.1145/937503.937505. URL https://doi.org/10.1145/937503.937505.

[11] Miguel A Carreira-Perpiñán and Geoffrey Hinton. On contrastive divergence learning. In *International Workshop on Artificial Intelligence and Statistics*, pages 33–40. PMLR, 2005. URL https://proceedings.mlr.press/r5/carreira-perpinan05a.html.

[12] Bor-Liang Chen and Ko-Wei Lih. Hamiltonian uniform subset graphs. 42(3):257–263, 1987. ISSN 0095-8956. doi: 10.1016/0095-8956(87)90044-X. URL https://www.sciencedirect.com/science/article/pii/009589568790044X.

[13] John M. Danskin. The theory of max-min, with applications. 14(4):641–664, 1966. ISSN 0036-1399. doi: 10.1137/0114053. URL https://epubs.siam.org/doi/abs/10.1137/0114053.

[14] Priya Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning in stochastic optimization. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[15] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models, 2020. URL https://arxiv.org/abs/1903.08689.

[16] Yilun Du, Shuang Li, Joshua Tenenbaum, and Igor Mordatch. Improved contrastive divergence training of energy based models, 2021. URL https://arxiv.org/abs/2012.01316.

[17] Ulrich Faigle and Rainer Schrader. On the convergence of stationary distributions in simulated annealing algorithms. 27(4):189–194, 1988. ISSN 0020-0190. doi: 10.1016/0020-0190(88)90024-5. URL https://www.sciencedirect.com/science/article/pii/0020019088900245.

[18] Ari Freedman. CONVERGENCE THEOREM FOR FINITE MARKOV CHAINS. 2017. URL https://www.semanticscholar.org/paper/CONVERGENCE-THEOREM-FOR-FINITE-MARKOV-CHAINS-%E2%8B%82t/65f7c092bd9c59cbbc88dd69266d39cd79840648.

[19] Michel Gendreau, Jean-Yves Potvin, et al. *Handbook of metaheuristics*, volume 2. Springer, 2010.

[20] Will Grathwohl, Kevin Swersky, Milad Hashemi, David Duvenaud, and Chris J. Maddison. Oops i took a gradient: Scalable sampling for discrete distributions, 2021. URL https://arxiv.org/abs/2102.04509.

[21] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2334940.

[22] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. 2000. URL https://www.semanticscholar.org/paper/Training-Products-of-Experts-by-Minimizing-Hinton/9360e5ce9c98166bb179ad479a9d2919ff13d022.

[23] Salvatore Ingrassia. On the rate of convergence of the metropolis algorithm and gibbs sampler by geometric bounds. 4(2):347–389, 1994. ISSN 1050-5164. URL https://www.jstor.org/stable/2245161.

[24] Gareth A. Jones. Automorphisms and regular embeddings of merged johnson graphs. 26 (3):417–435, 2005. ISSN 0195-6698. doi: 10.1016/j.ejc.2004.01.012. URL https://www.sciencedirect.com/science/article/pii/S0195669804000630.

[25] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL http://arxiv.org/abs/1412.6980.

[26] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. doi: 10.1126/science.220.4598.671. URL https://www.science.org/doi/abs/10.1126/science.220.4598.671.

[27] Wouter Kool, Laurens Bliek, Danilo Numeroso, Yingqian Zhang, Tom Catshoek, Kevin Tierney, Thibaut Vidal, and Joaquim Gromicho. The EURO meets NeurIPS 2022 vehicle routing competition. In *Proceedings of the NeurIPS 2022 Competitions Track*, pages 35–49. PMLR, 2023. URL https://proceedings.mlr.press/v220/kool23a.html.

[28] Rahul G. Krishnan, Simon Lacoste-Julien, and David Sontag. Barrier frank-wolfe for marginal inference, 2015. URL https://arxiv.org/abs/1511.02124.

[29] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558607781.

[30] Yann Lecun, Sumit Chopra, Raia Hadsell, Marc Aurelio Ranzato, and Fu Jie Huang. *A tutorial on energy-based learning*. MIT Press, 2006.

[31] Zhifei Li and Jason Eisner. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In Philipp Koehn and Rada Mihalcea, editors, *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 40–51. Association for Computational Linguistics, August 2009. URL https://aclanthology.org/D09-1005/.

[32] Neal Madras and Dana Randall. Markov chain decomposition for convergence rate analysis. 12 (2):581–606, 2002. ISSN 1050-5164, 2168-8737. doi: 10.1214/aoap/1026915617. URL https://projecteuclid.org/journals/annals-of-applied-probability/volume-12/issue-2/Markov-chain-decomposition-for-convergence-rate-analysis/10.1214/aoap/1026915617.full.

[33] Jayanta Mandi and Tias Guns. Interior point solving for LP-based prediction+optimisation, 2020. URL http://arxiv.org/abs/2010.13943.

[34] Jayanta Mandi, James Kotary, Senne Berden, Maxime Mulamba, Victor Bucarey, Tias Guns, and Ferdinando Fioretto. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. 80:1623–1701, 2024. ISSN 1076-9757. doi: 10.1613/jair.1.15320. URL http://arxiv.org/abs/2307.13565.

[35] Arthur Mensch and Mathieu Blondel. Differentiable dynamic programming for structured prediction and attention, 2018. URL https://arxiv.org/abs/1802.03676.

[36] Debasis Mitra, Fabio Romeo, and Alberto Sangiovanni-Vincentelli. Convergence and finite-time behavior of simulated annealing. *Advances in Applied Probability*, 18(3):747–771, 1986. ISSN 0001-8678. doi: 10.2307/1427186. URL https://www.jstor.org/stable/1427186.

[37] Nenad Mladenović and Pierre Hansen. Variable neighborhood search. *Computers & operations research*, 24(11):1097–1100, 1997.

[38] Volodymyr Mnih, Hugo Larochelle, and Geoffrey E. Hinton. Conditional restricted boltzmann machines for structured output prediction, 2012. URL http://arxiv.org/abs/1202.3748.

[39] Vlad Niculae, André F. T. Martins, Mathieu Blondel, and Claire Cardie. Sparsemap: Differentiable sparse structured inference, 2018. URL https://arxiv.org/abs/1802.04223.

[40] Benjamin Rhodes and Michael Gutmann. Enhanced gradient-based MCMC in discrete spaces, 2022. URL http://arxiv.org/abs/2208.00040.

[41] Fred J. Rispoli. The graph of the hypersimplex, 2008. URL http://arxiv.org/abs/0811.2981.

[42] R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970. ISBN 9780691015866. URL http://www.jstor.org/stable/j.ctt14bs1ff.

[43] Utsav Sadana, Abhilash Chenreddy, Erick Delage, Alexandre Forel, Emma Frejinger, and Thibaut Vidal. A survey of contextual optimization methods for decision making under uncertainty, 2024. URL http://arxiv.org/abs/2306.10374.

[44] Yang Song and Diederik P. Kingma. How to train your energy-based models, 2021. URL https://arxiv.org/abs/2101.03288.

[45] Haoran Sun, Hanjun Dai, Bo Dai, Haomin Zhou, and Dale Schuurmans. Discrete langevin sampler via wasserstein gradient flow, 2023. URL http://arxiv.org/abs/2206.14897.

[46] Haoran Sun, Katayoon Goshvadi, Azade Nova, Dale Schuurmans, and Hanjun Dai. Revisiting sampling for combinatorial optimization. In *Proceedings of the 40th International Conference on Machine Learning*, pages 32859–32874. PMLR, 2023. URL https://proceedings.mlr.press/v202/sun23c.html.

[47] Ilya Sutskever and Tijmen Tieleman. On the convergence properties of contrastive divergence. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 789–795. JMLR Workshop and Conference Proceedings, 2010. URL https://proceedings.mlr.press/v9/sutskever10a.html.

[48] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 1064–1071. Association for Computing Machinery, 2008. ISBN 9781605582054. doi: 10.1145/1390156.1390290. URL https://doi.org/10.1145/1390156.1390290.

[49] Thibaut Vidal. Hybrid genetic search for the cvrp: Open-source implementation and swap* neighborhood. *Computers & Operations Research*, 140:105643, April 2022. ISSN 0305-0548. doi: 10.1016/j.cor.2021.105643. URL http://dx.doi.org/10.1016/j.cor.2021.105643.

[50] Marin Vlastelica, Anselm Paulus, Vít Musil, Georg Martius, and Michal Rolínek. Differentiation of blackbox combinatorial solvers, 2020. URL http://arxiv.org/abs/1912.02175.

[51] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. 1(1):1–305, 2008. ISSN 1935-8237, 1935-8245. doi: 10.1561/2200000001. URL https://www.nowpublishers.com/article/Details/MAL-001.

[52] Laurent Younes. Stochastic gradient estimation strategies for markov random fields. In *Bayesian Inference for Inverse Problems*, volume 3459, pages 315–325. SPIE, 1998. doi: 10.1117/12.323811. URL https://www.spiedigitallibrary.org/conference-proceedings-of-spie/3459/0000/Stochastic-gradient-estimation-strategies-for-Markov-random-fields/10.1117/12.323811.full.

[53] Ruqi Zhang, Xingchao Liu, and Qiang Liu. A langevin-like sampler for discrete distributions, 2022. URL https://arxiv.org/abs/2206.09914.

A Background and Related Work

A.1 Combinatorial optimization as a layer

Because the function in Eq. (1) is piecewise constant and discontinuous, a frequent strategy consists in introducing regularization in the problem so as to obtain a continuous relaxation. In some cases, we may have access to an oracle for directly solving the regularized problem. For instance, when the unregularized problem can be solved by dynamic programming, its entropic regularization can be computed using a change of semi-ring [31] or by algorithmic smoothing [35]. As another example, interior point solvers can be used to compute a logarithmic barrier regularized solution [33].

We focus settings where only a MAP oracle is available for the original, unregularized optimization problem. While many prior works are limited to the linear form in Eq. (1) for the latter, our framework is more general and also handles problems of the form in Eq. (2). Frank-Wolfe-like methods can be used to solve the regularized problem using only MAP oracle calls [39, 28]. Another strategy consists in injecting noise perturbations [6] in the oracle. This approach can be shown to be implicitly using regularization. In both cases, a Fenchel-Young loss can be associated, giving a principled way to learn with the optimization layer. However, formal guarantees only hold if the oracle used is exact, and in practice it is typically called multiple times during the forward pass. Our proposal enjoys guarantees even with inexact solvers and a single call.

Regarding differentiation, several strategies are possible. When the approach only needs to differentiate through a (regularized) $\max$, as is the case of Fenchel-Young losses, we can use Danskin's theorem [13]. When the approach needs to differentiate a (regularized) argmax, we can either use autodiff on the unrolled solver iterations or implicit differentiation [3, 1, 9]. Differently, Vlastelica et al. [50] propose to compute gradients via continuous interpolation of the solver.

A.2 Contrastive divergences

An alternative approach to learning in combinatorial spaces is to use energy-based models (EBMs) [30], which define a distribution over outputs via a parameterized energy function $E_{\boldsymbol{\theta}}$:

$$\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{y}) \propto \exp(E_{\boldsymbol{\theta}}(\boldsymbol{y})), \quad \text{with} \quad \nabla_{\boldsymbol{\theta}} \log \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{y}) = \nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\boldsymbol{y}) - \mathbb{E}_{Y \sim \boldsymbol{p}_{\boldsymbol{\theta}}} \left[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(Y) \right].$$

Therefore, we can perform maximum likelihood estimation (MLE) if we can sample from $\boldsymbol{p}_{\boldsymbol{\theta}}$, but this is hard both in continuous and combinatorial settings, due to its intractable normalization constant. Contrastive divergences [22, 11, 44] address this by using MCMC to obtain (biased) stochastic gradients. Originally developed for restricted Boltzmann machines with $\mathcal{Y} = \{0, 1\}^d$ and a Gibbs sampler, they have also been applied in continuous domains via Langevin dynamics [15, 16].

MCMC in discrete spaces. Contrastive divergences rely on MCMC to sample from the current model distribution. Unfortunately, designing an MCMC sampler is usually case by case, and MCMC on discrete domains has received comparatively less attention than continuous domains. Recent efforts adapt continuous techniques, such as Langevin dynamics [53, 45] or gradient-informed proposals [20, 40], to discrete settings. However, these works typically assume simple state space structure, like the hypercube or categorical codebooks, and do not handle complex constraints that are ubiquitous in operations research problems. Sun et al. [46] allow structured state spaces via relaxed constraints in the energy function, yet ignore these structures in the proposal supports. Notably, we emphasize that all these works focus on sampling, not on designing differentiable MCMC layers.

B Experiments on empirical convergence of gradients and parameters

In this section, we evaluate the proposed approach on two discrete output spaces: sets and κ-subsets. These output spaces are for instance useful for multilabel classification. We focus on these output spaces because the exact MAP and marginal inference oracles are available, allowing us to compare our gradient estimators to exact gradients.

B.1 Polytopes and corresponding oracles

The vertex set of the first polytope is the set of binary vectors in $\mathbb{R}^d$, which we denote $\mathcal{Y}^d \coloneqq \{0, 1\}^d$, and $\mathsf{conv}(\mathcal{Y}^d) = [0, 1]^d$ is the "hypercube". The vertex set of the second is the set of binary vectors

with exactly κ ones and $d - \kappa$ zeros (with $0 < \kappa < d$),

$$\mathcal{Y}_\kappa^d \coloneqq \left\{ \boldsymbol{y} \in \{0,1\}^d : \langle \boldsymbol{y}, \mathbf{1} \rangle = \kappa \right\},$$

and $\mathsf{conv}(\mathcal{Y}_\kappa^d)$ is referred to as "top-κ polytope" or "hypersimplex". Although these polytopes would not provide relevant use cases of the proposed approach in practice, since exact marginal inference oracles are available (see below), they allow us to compare the Fenchel-Young loss value and gradient estimated by our algorithm to their true value.

Marginal inference. For the hypercube, we have:

$$\begin{aligned}
\mathbb{E}_{\pi_{\boldsymbol{\theta},t}}[Y_i] &= \sum_{\boldsymbol{y} \in \mathcal{Y}^d} \frac{\exp\left(\langle \boldsymbol{\theta}, \boldsymbol{y} \rangle / t\right)}{\sum_{\boldsymbol{y}' \in \mathcal{Y}^d} \exp\left(\langle \boldsymbol{\theta}, \boldsymbol{y}' \rangle / t\right)} y_i \\
&= \sum_{\boldsymbol{y} \in \{0,1\}^d} \frac{\exp\left(\sum_{j=1}^d \theta_j y_j / t\right)}{\sum_{\boldsymbol{y}' \in \{0,1\}^d} \exp\left(\sum_{j=1}^d \theta_j y'_j / t\right)} y_i \\
&= \sum_{y_i \in \{0,1\}} \sum_{\boldsymbol{y}_{-i} \in \{0,1\}^{d-1}} \frac{\exp\left(\theta_i y_i / t + \sum_{j \neq i} \theta_j y_j / t\right)}{\sum_{y'_i \in \{0,1\}} \sum_{\boldsymbol{y}'_{-i} \in \{0,1\}^{d-1}} \exp\left(\theta_i y'_i / t + \sum_{j \neq i} \theta_j y'_j / t\right)} y_i \\
&= \sum_{y_i \in \{0,1\}} \frac{\exp\left(\theta_i y_i / t\right)}{\sum_{y'_i \in \{0,1\}} \exp\left(\theta_i y'_i / t\right)} y_i \sum_{\boldsymbol{y}_{-i} \in \{0,1\}^{d-1}} \frac{\exp\left(\sum_{j \neq i} \theta_j y_j / t\right)}{\sum_{\boldsymbol{y}'_{-i} \in \{0,1\}^{d-1}} \exp\left(\sum_{j \neq i} \theta_j y'_j / t\right)} \\
&= \sum_{y_i \in \{0,1\}} \frac{\exp\left(\theta_i y_i / t\right)}{\sum_{y'_i \in \{0,1\}} \exp\left(\theta_i y'_i / t\right)} y_i \\
&= \frac{0 \cdot \exp(0) + 1 \cdot \exp\left(\theta_i / t\right)}{\exp(0) + \exp\left(\theta_i / t\right)} \\
&= \frac{\exp\left(\theta_i / t\right)}{1 + \exp\left(\theta_i / t\right)} \\
&= \sigma\left(\frac{\theta_i}{t}\right),
\end{aligned}$$

which gives $\mathbb{E}_{\pi_{\boldsymbol{\theta},t}}[Y] = \sigma\left(\frac{\boldsymbol{\theta}}{t}\right)$, where the logistic sigmoid function σ is applied component-wise. The cumulant function is also tractable, as we have

$$\begin{aligned}
\log \sum_{\boldsymbol{y} \in \mathcal{Y}^d} \exp\left(\langle \boldsymbol{\theta}, \boldsymbol{y} \rangle / t\right) &= \log \sum_{\boldsymbol{y} \in \{0,1\}^d} \exp\left(\sum_{i=1}^d \theta_i y_i / t\right) \\
&= \log \sum_{\boldsymbol{y}_1=0}^{1} \sum_{\boldsymbol{y}_2=0}^{1} \cdots \sum_{\boldsymbol{y}_d=0}^{1} \exp\left(\sum_{i=1}^d \theta_i y_i / t\right) \\
&= \log \prod_{i=1}^d \sum_{y_i=0}^{1} \exp\left(\theta_i y_i / t\right) \\
&= \log \prod_{i=1}^d \left(\exp(0) + \exp\left(\theta_i / t\right)\right) \\
&= \log \prod_{i=1}^d \left(1 + \exp\left(\theta_i / t\right)\right) \\
&= \sum_{i=1}^d \log\left(1 + \exp\left(\theta_i / t\right)\right).
\end{aligned}$$

Another way to derive this is via the Fenchel conjugate.

For the top-κ polytope, such closed-form formulas do not exist for the cumulant and its gradient. However, we implement them with dynamic programming, by viewing the top-κ MAP problem as a $0/1$-knapsack problem with constant item weights, and by changing the $(\max, +)$ semiring into a $(\mathrm{LSE}, +)$ semiring. This returns the cumulant function, and we leverage PyTorch's automatic differentiation framework to compute its gradient. This simple implementation allows us to compute true Fenchel-Young losses values and their gradients in $\mathcal{O}(d\kappa)$ time and space complexity.

Sampling. For the hypercube, sampling from the Gibbs distribution on $\mathcal{Y}^d$ has closed form. Indeed, the latter is fully factorized, and we can sample $\boldsymbol{y} \sim \pi_{\boldsymbol{\theta},t}$ by sampling independently each component with $y_i \sim \mathrm{Bern}\left(\sigma(\theta_i/t)\right)$. Sampling from $\pi_{\boldsymbol{\theta},t}$ is also possible on $\mathcal{Y}_\kappa^d$, by sampling coordinates iteratively using the dynamic programming table used to compute the cumulant function (see, e.g., Algorithm 2 in Ahmed et al. [2] for a detailed explanation).

B.2 Neighborhood graphs

Hypercube. On $\mathcal{Y}^d$, we use a family of neighborhood systems $\mathcal{N}_\leq^r$ parameterized by a Hamming distance radius $r \in [d-1]$. The graph is defined by:

$$\forall \boldsymbol{y}, \boldsymbol{y}' \in \mathcal{Y}^d : \ \boldsymbol{y}' \in \mathcal{N}_\leq^r(\boldsymbol{y}) \Leftrightarrow 1 \leq d_H\left(\boldsymbol{y}, \boldsymbol{y}'\right) \leq r.$$

That is, two vertices are neighbors if their Hamming distance is at most r. This graph is regular, with degree $|\mathcal{N}_\leq^r(\boldsymbol{y})| = \sum_{i=1}^r \binom{d}{i}$. This graph is naturally connected, as any binary vector $\boldsymbol{y}'$ can be reached from any other binary vector $\boldsymbol{y}$ in $||\boldsymbol{y}' - \boldsymbol{y}||_1$ moves, by flipping each bit where $y'_i \neq y_i$, iteratively. Indeed, this trajectory consists in moves between vertices with Hamming distance equal to 1, and are therefore along edges of the neighborhood graph, regardless of the value of r.

We also use a slight variation on this family of neighborhood systems, the graphs $\mathcal{N}_=^r$, defined by:

$$\forall \boldsymbol{y}, \boldsymbol{y}' \in \mathcal{Y}^d : \ \boldsymbol{y}' \in \mathcal{N}_=^r(\boldsymbol{y}) \Leftrightarrow d_H\left(\boldsymbol{y}, \boldsymbol{y}'\right) = r.$$

These graphs, on the contrary, are not always connected: indeed, if r is even, they contain two connected components (binary vectors with an even sum, and binary vectors with an odd sum). We only use such graphs when experimenting with neighborhood mixtures (see Algorithm 2), by aggregating them into a connected graph.

Top-κ polytope. On $\mathcal{Y}_\kappa^d$, we use a family of neighborhoods systems $\mathcal{N}^s$ parameterized by a number of "swaps" $s \in [\![1, \min(\kappa, d-\kappa)]\!]$. The graph is defined by

$$\forall \boldsymbol{y}, \boldsymbol{y}' \in \mathcal{Y}_\kappa^d : \ \boldsymbol{y}' \in \mathcal{N}^s(\boldsymbol{y}) \Leftrightarrow d_H\left(\boldsymbol{y}, \boldsymbol{y}'\right) = 2s.$$

That is, two vertices are neighbors if one can be reached from the other by performing s "swaps", each swap corresponding to flipping a 1 to a 0 and vice-versa. This ensures that the resulting vector is still in $\mathcal{Y}_\kappa^d$. All s swaps must be performed on distinct components. The resulting graph is known as the *Generalized Johnson Graph* $J(d, \kappa, \kappa - s)$, or *Uniform Subset Graph* [12]. It is a regular graph, with degree $|\mathcal{N}^s(\boldsymbol{y})| = \binom{\kappa}{s}\binom{d-\kappa}{s}$. It is proved to be connected in Jones [24], except if $d = 2\kappa$ and $s = \kappa$ (in this case, it consists in $\frac{1}{2}\binom{d}{\kappa}$ disjoint edges).

When $s = 1$, the neighborhood graph is the Johnson Graph $J(d, \kappa)$, which coincides with the graph associated to the polytope $\mathsf{conv}(\mathcal{Y}_\kappa^d) = \Delta_{d,\kappa}$ [41].

B.3 Convergence to exact gradients

In this section, we conduct experiments on the convergence of the MCMC estimators to the exact corresponding expectation (that is, convergence of the stochastic gradient estimator to the true gradient). The estimators are defined as

$$\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \mathbb{E}_{\pi_{\boldsymbol{\theta},t}}\left[Y\right] \approx \frac{1}{K - K_0} \sum_{k=K_0+1}^{K} \boldsymbol{y}^{(k)},$$

where $\boldsymbol{y}^{(k)}$ is the k-th iterate of Algorithm 1 with maximization direction $\boldsymbol{\theta}$, final temperature t, and K_0 is a number of burn-in (or warm-up) iterations. The obtained estimator is compared to the exact expectation by performing marginal inference as described in Appendix B.1 (with a closed-form formula in the case of $\mathcal{Y}^d$, and by dynamic programming in the case of $\mathcal{Y}^d_\kappa$).

Setup. For $T > K_0$, let $\tilde{\mathbb{E}}(\boldsymbol{\theta}, T) \coloneqq \frac{1}{T-K_0} \sum_{k=K_0+1}^{T} \boldsymbol{y}^{(k)}$ be the stochastic estimate of the expectation at step T. We proceed by first randomly generating $\boldsymbol{\Theta} \in \mathbb{R}^{M \times d}$, with M being the number of instances, by sampling $\boldsymbol{\Theta}_{i,j} \sim \mathcal{N}(0, 1)$ independently. Then, we evaluate the impact of the following hyperparameters on the estimation of $\mathbb{E}_{\pi_{\boldsymbol{\Theta}_i}, t}[Y]$, for $i \in [M]$:

1. K_0, the number of burn-in iterations,
2. t, the temperature parameter,
3. C, the number of parallel Markov chains.

Metric. The metric used is the squared Euclidean distance to the exact expectation, averaged on the M instances

$$\frac{1}{M} \sum_{i=1}^{M} ||\mathbb{E}_{\pi_{\boldsymbol{\Theta}_i}, t}[Y] - \tilde{\mathbb{E}}(\boldsymbol{\Theta}_i, T)||_2^2,$$

which we measure for $T \in [\![K_0 + 1, K]\!]$.

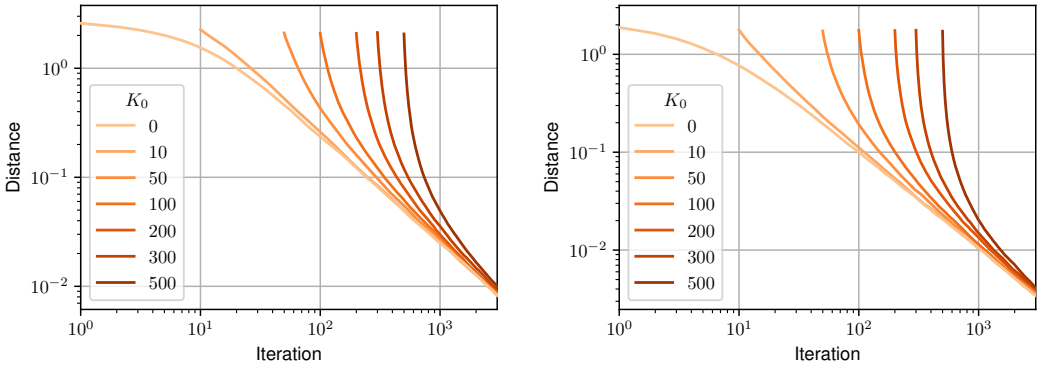
Polytopes. For the hypercube $\mathcal{Y}^d$ and its neighborhood system $\mathcal{N}^r_{\leq}$, we use $d = 10$ and $r = 1$, which gives $|\mathcal{Y}^d| = 2^{10}$ and $|\mathcal{N}^r_{\leq}(\boldsymbol{y})| = 10$. For the top-$\kappa$ polytope $\mathcal{Y}^d_\kappa$ and its neighborhood system $\mathcal{N}^s$, we use $d = 10$, $\kappa = 3$ and $s = 1$, which gives $|\mathcal{Y}^d_\kappa| = 120$ and $|\mathcal{N}^s(\boldsymbol{y})| = 30$. We also use a larger scale for both polytopes in order to highlight the varying impact of the temperature t depending on the combinatorial size of the problem, in the second experiment. For the large scale, we use $d = 1000$ and $r = 10$ for the hypercube, which give $|\mathcal{Y}^d| = 2^{1000} \approx 10^{301}$ and $|\mathcal{N}^r_{\leq}(\boldsymbol{y})| \approx 2.7 \times 10^{23}$, and we use $d = 1000$, $\kappa = 50$ and $s = 10$ for the top-κ polytope, which give $|\mathcal{Y}^d_\kappa| \approx 9.5 \times 10^{84}$ and $|\mathcal{N}^s(\boldsymbol{y})| \approx 1.6 \times 10^{33}$.

Hyperparameters. For each experiment, we use $K = 3000$. We average over $M = 1000$ problem instances for statistical significance. We use $K_0 = 0$, except for the first experiment, where it varies. We use a final temperature $t = 1$, except for the second experiment, where it varies. We use an initial temperature $t_0 = t = 1$ (leading to a constant temperature schedule), except for the first experiment, where it depends on K_0. We use only one Markov chain and thus have $C = 1$, except for the third experiment, where it varies.

(1) Impact of burn-in. First, we evaluate the impact of K_0, the number of burn-in iterations. We use a truncated geometric cooling schedule $t_k = \max(\gamma^k \cdot t_0, t)$ with $\gamma = 0.995$. The initial temperature t_0 is set to $1/(\gamma^{K_0})$, so that $\forall k \geq K_0 + 1,\ t_k = t = 1$. The results are gathered in Fig. 3.

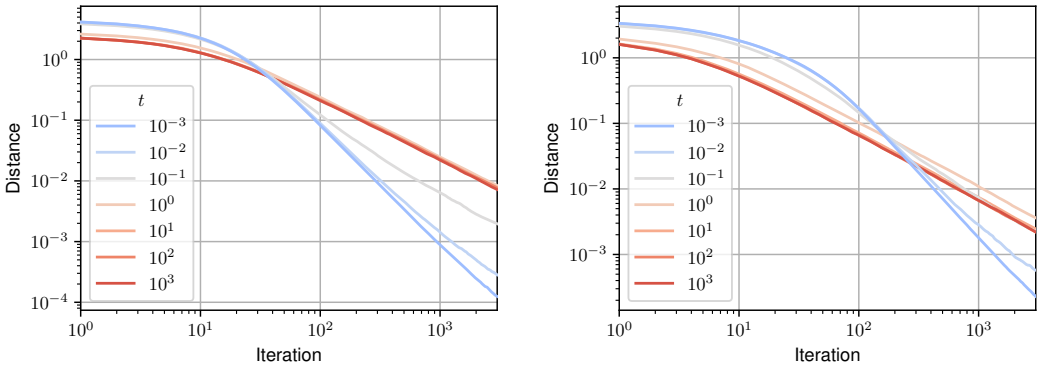
(2) Impact of temperature. We then evaluate the impact of the final temperature t on the difficulty of the estimation task (different temperatures lead to different target expectations). The results for the small scale are gathered in Fig. 4, and the results for the large scale are gathered in Fig. 5.

(3) Impact of the number of parallel Markov chains. Finally, we evaluate the impact of the number of parallel Markov chains C on the estimation. The results are gathered in Fig. 6.



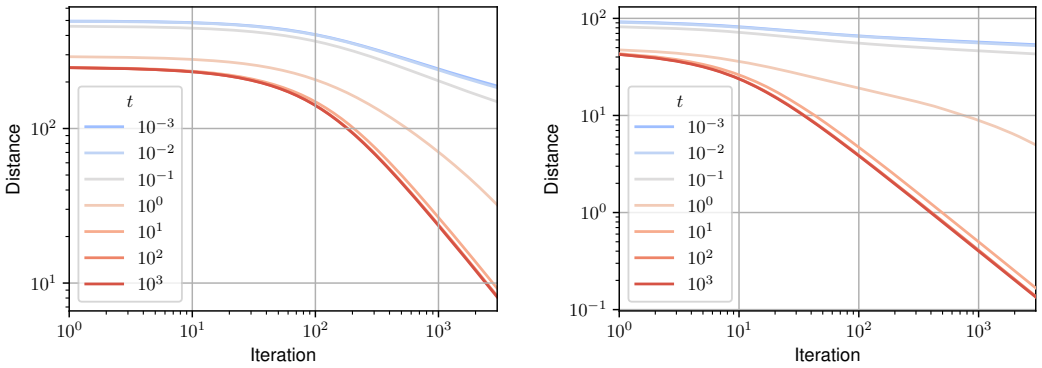
(a) Hypercube (b) top-κ polytope

Figure 3: Convergence to exact expectation on the hypercube and the top-κ polytope, for varying number of burn-in iterations K_0. We conclude that burn-in is not beneficial to the estimation, and taking $K_0 = 0$ is the best option.



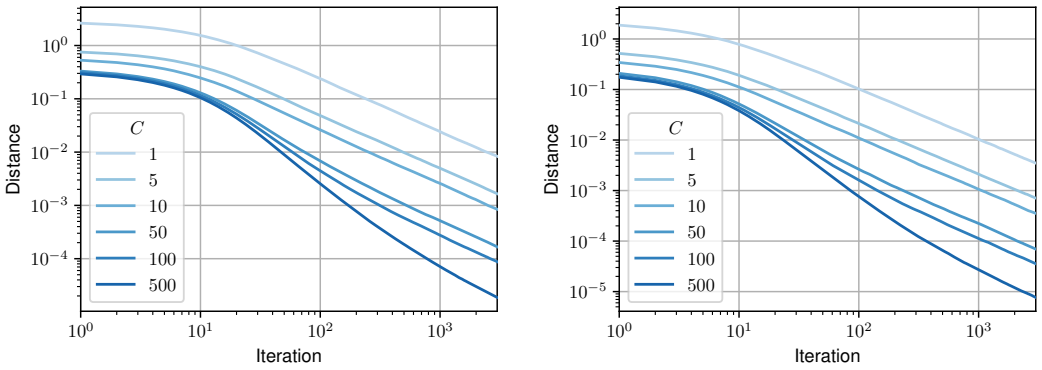
(a) Hypercube (b) top-κ polytope

Figure 4: Convergence to exact expectation on the hypercube and the top-κ polytope, for varying final temperature t (small scale experiment). We conclude that lower temperatures facilitate the estimation.



(a) Hypercube (b) top-κ polytope

Figure 5: Convergence to exact expectation on the hypercube and the top-κ polytope, for varying final temperature t (large scale experiment). Contrary to the small scale case, larger temperatures are beneficial to the estimation when the solution set is combinatorially large.



(a) SA, hypercube

(b) SA, top-κ polytope

Figure 6: Convergence to exact expectation on the hypercube and the top-κ polytope, for varying number of parallel Markov chains C. Running 10 times more chains in parallel provides roughly the same benefit as extending each chain by 10 times more iterations, highlighting the advantage of massively parallelized estimation.

B.4 Convergence to exact parameters

In this section, we conduct experiments in the unsupervised setting described in Section 3.3. As a reminder, the empirical L_N and population $L_{\boldsymbol{\theta}_0}$ Fenchel-Young losses are given by:

$$\begin{aligned} L_N(\boldsymbol{\theta}; \boldsymbol{y}_1, \dots, \boldsymbol{y}_N) &\coloneqq \frac{1}{N} \sum_{i=1}^{N} \ell_t\left(\boldsymbol{\theta}; \boldsymbol{y}_i\right) \\ &= A_t(\boldsymbol{\theta}) + \frac{1}{N} \sum_{i=1}^{N} \Omega_t(\boldsymbol{y}_i) - \langle \boldsymbol{\theta}, \bar{Y}_N \rangle \\ &= \ell_t(\boldsymbol{\theta}; \bar{Y}_N) + C_1(Y), \end{aligned} \tag{8}$$

and

$$\begin{aligned} L_{\boldsymbol{\theta}_0}(\boldsymbol{\theta}) &\coloneqq \mathbb{E}_{(\boldsymbol{y}_i)_{i=1}^N \sim (\pi_{\boldsymbol{\theta}_0, t})^{\otimes N}} \left[L_N(\boldsymbol{\theta}; \boldsymbol{y}_1, \dots, \boldsymbol{y}_N) \right] \\ &= A_t(\boldsymbol{\theta}) + \mathbb{E}_{\pi_{\boldsymbol{\theta}_0, t}} \left[\Omega_t(Y) \right] - \langle \boldsymbol{\theta}, \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}_0) \rangle \\ &= \ell_t(\boldsymbol{\theta}; \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}_0)) + C_2(\boldsymbol{\theta}_0), \end{aligned} \tag{9}$$

where the constants $C_1(Y) = \frac{1}{N} \sum_{i=1}^{N} \Omega_t(\boldsymbol{y}_i) - \Omega_t(\bar{Y}_N)$ and $C_2(\boldsymbol{\theta}_0) = \mathbb{E}_{\pi_{\boldsymbol{\theta}_0, t}} \left[\Omega_t(Y) \right] - \Omega_t \left(\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}_0) \right)$ do not depend on $\boldsymbol{\theta}$. As Jensen gaps, they are non-negative by convexity of Ω_t.

2D visualization. As an introductory example, we display stochastic gradient trajectories in Fig. 7. The parameter $\boldsymbol{\theta} \in \mathbb{R}^d$ is updated following Eq. (7) to minimize the population loss $L_{\boldsymbol{\theta}_0}$ defined in Eq. (9), with $\boldsymbol{\theta}_0 = (1/2, 1/2)$. The polytope used is the 2-dimensional hypercube $\mathcal{Y}^2$, with neighborhood graph $\mathcal{N}_1$ (neighbors are adjacent vertices of the square). We present trajectories obtained using MCMC-sampled gradients, comparing results from both 1 and 100 Markov chain iterations with Algorithm 1. For comparison, we include trajectories obtained using Monte Carlo-sampled (i.e., unbiased) gradients, using 1 and 100 samples.

General setup. We proceed by first randomly generating true parameters $\boldsymbol{\Theta}_0 \in \mathbb{R}^{M \times d}$, with M being a number of problem instances we average on (in order to reduce noise in our observations), by sampling $\boldsymbol{\Theta}_{i,j} \sim \mathcal{N}(0, 1)$ independently. The goal is to learn each parameter vector $(\boldsymbol{\Theta}_0)_i \in \mathbb{R}^d, i \in [M]$, as M independent problems. The model is randomly initialized at $\hat{\boldsymbol{\Theta}}_0$, and updated with Adam [25] to minimize the loss. In order to better separate noise due to the optimization process and noise due to the sampling process, we use the population loss $L_{(\boldsymbol{\Theta}_0)_i}$ for general experiments, and use the empirical loss L_N only when focusing on the impact of the dataset size N. In this case,

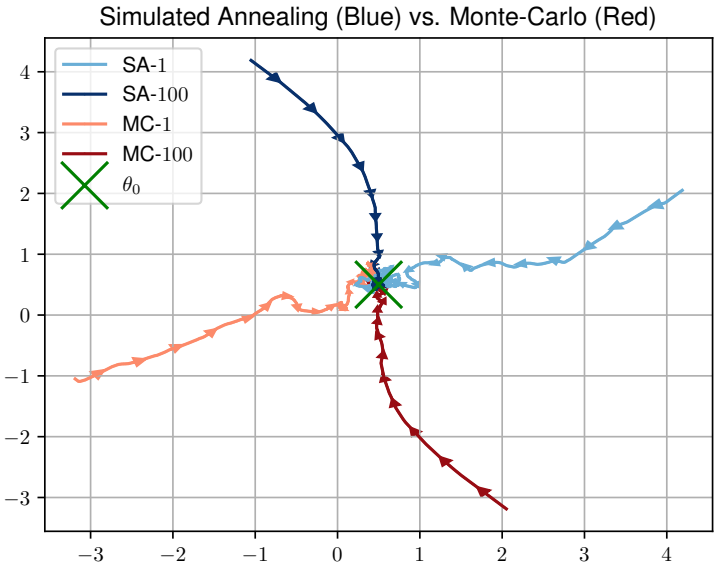


Figure 7: Comparison of stochastic gradient trajectories for a SA / M-H oracle on $\mathcal{Y}^2$ and unbiased stochastic gradients obtained via Monte Carlo sampling. Increasing the number of Markov chain iterations yields smoother trajectories, similar to the effect of using more Monte Carlo samples in the case of perturbation-based methods [6].

we create a dataset $Y \in \mathbb{R}^{M \times N \times d}$, with N being the number of samples, by sampling independently $Y_{i,j} \sim \pi_{(\mathbf{\Theta}_0)_i}$, $\forall i \in [M]$, $\forall j \in [N]$.

We study the impact of the following hyperparameters on learning:

1. K, the number of Markov chain iterations,
2. C, the number of parallel Markov chains,
3. the initialization method used for the chains (either random, persistent, or data-based),
4. N, the number of samples in the dataset.

Metrics. The first metric used is the objective function actually minimized, i.e., the population loss, averaged on the M instances:

$$\frac{1}{M} \sum_{i=1}^{M} L_{(\mathbf{\Theta}_0)_i}((\hat{\mathbf{\Theta}}_n)_i),$$

where $(\hat{\mathbf{\Theta}}_n)_i$ is the n-th iterate of the optimization process for the problem instance $i \in [M]$. We measure this loss for $n \in [n_{\text{max}}]$, with n_{max} the total number of gradient iterations. For the fourth experiment, where we evaluate the impact of the number of samples N, we measure instead the empirical Fenchel-Young loss:

$$\frac{1}{M} \sum_{i=1}^{M} L_N((\hat{\mathbf{\Theta}}_n)_i\,;\, Y_{i,1}, \dots Y_{i,N})$$

In both cases, the best loss value that can be reached is positive but cannot be computed: it corresponds to the constants C_1 and C_2 in Eq. (8) and Eq. (9). Thus, we also provide "stretched" figures, where we plot the loss minus the best loss found during the optimization process.

The second metric used is the squared euclidean distance of the estimate to the true parameter, also averaged on the M instances:

$$\frac{1}{M}\sum_{i=1}^{M}||(\boldsymbol{\Theta}_0)_i - (\hat{\boldsymbol{\Theta}}_n)_i||_2^2.$$

As the top-κ polytope is of dimension $d-1$, the model is only specified up to vectors orthogonal to the direction of the smallest affine subspace it spans. Thus, in this case, we measure instead:

$$\frac{1}{M}\sum_{i=1}^{M}||\,\mathsf{P}_D^{\perp}\left((\boldsymbol{\Theta}_0)_i\right) - \mathsf{P}_D^{\perp}\left((\hat{\boldsymbol{\Theta}}_n)_i\right)||_2^2,$$

where $\mathsf{P}^{\perp}{}_D$ is the orthogonal projector on the hyperplane $D = \{\boldsymbol{x} \in \mathbb{R}^d : \langle \mathbf{1}, \boldsymbol{x}\rangle = 0\}$, which is the corresponding direction.

Polytopes. For the hypercube $\mathcal{Y}^d$ and its neighborhood system $\mathcal{N}^r_{\leq}$, we use $d = 10$ and $r = 1$, except in the fifth experiment, where we use a mixture of $\mathcal{N}^r_{=}$ neighborhoods (detailed below). For the top-κ polytope $\mathcal{Y}^d_\kappa$ and its neighborhood system $\mathcal{N}^s$, we use $d = 10$, $\kappa = 3$ and $s = 1$.

Hyperparameters. For each experiment, we perform 1000 gradient steps. We use $K_0 = 0$, final temperature $t = 1$ and initial temperature $t_0 = t = 1$ (leading to a constant temperature schedule). We use $K = 1000$ Markov chain iterations, except in the first experiment, where it varies. We use only one Markov chain and thus have $C = 1$, except for the second experiment, where it varies. We use a persistent initialization method for the Markov chains, except in the third experiment, where we compare the three different methods. For statistical significance, we average over $M = 100$ problem instances for each experiment, except in the third experiment, where we use $M = 1000$. We work in the limit case $N \to \infty$, except in the fourth experiment, where N varies.

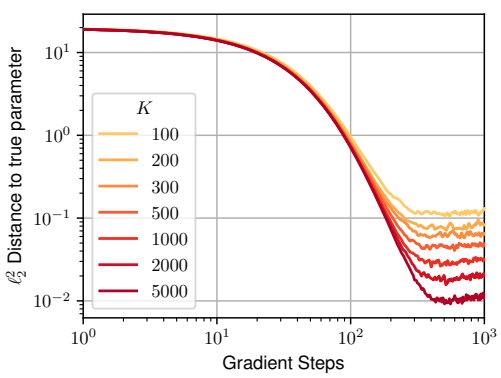
(1) Impact of the length of Markov chains. First, we evaluate the impact of K, the number of inner iterations, i.e., the length of each Markov chain. The results are gathered in Fig. 8.

(2) Impact of the number of parallel Markov chains. We now evaluate the impact of the number of Markov chains C run in parallel to perform each gradient estimation on the learning process. The results are gathered in Fig. 9.

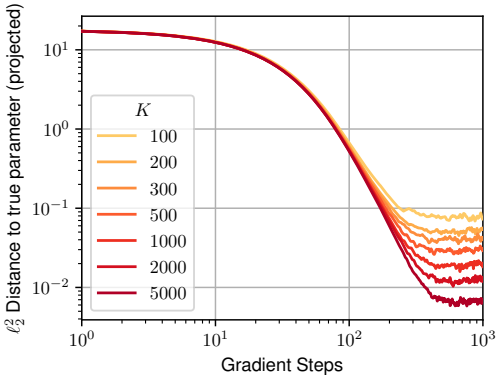
(3) Impact of the initialization method. Then, we evaluate the impact of the method to initialize each Markov chain used for gradient estimation. The persistent method consists in setting $\boldsymbol{y}^{(n+1,0)} = \boldsymbol{y}^{(n,K)}$, the data-based method consists in setting $\boldsymbol{y}^{(n+1,0)} = \boldsymbol{y}_i$ with $i \sim \mathcal{U}([N])$, and the random method consists in setting $\boldsymbol{y}^{(n+1,0)} \sim \mathcal{U}(\mathcal{Y})$ (see Appendix C.5 and Table 3 for a detailed explanation). The results are gathered in Fig. 10.

(4) Impact of the dataset size. We now evaluate the impact of the number of samples N from $\pi_{\boldsymbol{\theta}_0}$ (i.e., the size of the dataset $(\boldsymbol{y}_i)_{i=1}^N$) on the estimation of the true parameter $\boldsymbol{\theta}_0$. The results are gathered in Fig. 11.

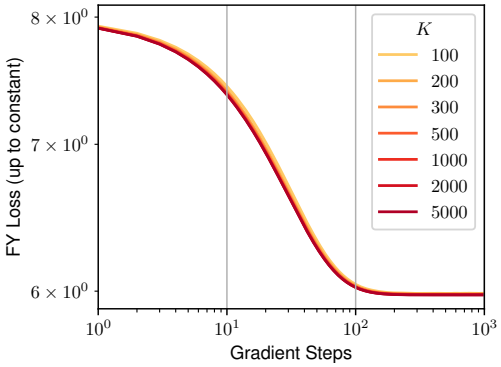
(5) Impact of neigborhood mixtures. Finally, we evaluate the impact of the use of neighborhood mixtures. To do so, we use mixtures $\{\mathcal{N}^{r_s}_{=}\}_{s=1}^S$, once with $\{r_s\}_{s=1}^S = \{5\}$ opposed to $\{r_s\}_{s=1}^S = \{1, 5\}$, and once with $\{r_s\}_{s=1}^S = \{6\}$ (which gives a reducible Markov chain as 6 is even, so that the individual neighborhood graph $\mathcal{N}^6_{=}$ is not connected, and has to connected components) opposed to $\{r_s\}_{s=1}^S = \{1, 2, 3, 6\}$. The results are gathered in Fig. 12.



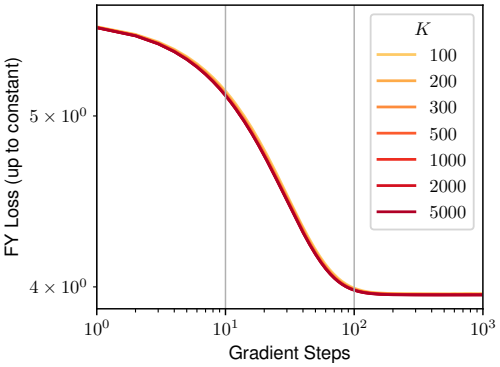
(a) Distance to true parameter, hypercube



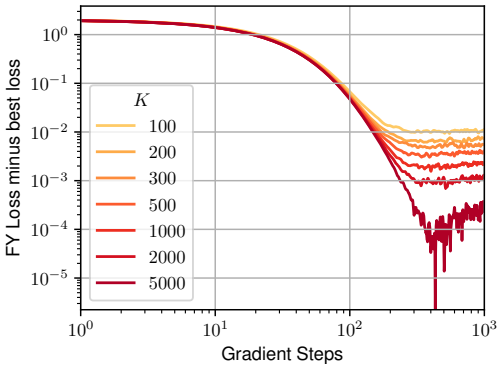
(b) Distance to true parameter, top-κ polytope



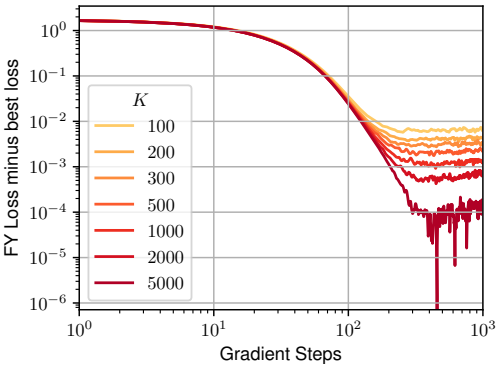
(c) FY loss (up to constant), hypercube



(d) FY loss (up to constant), top-κ polytope

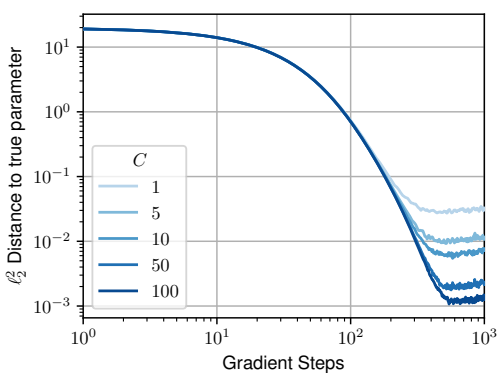


(e) FY loss minus best loss, hypercube

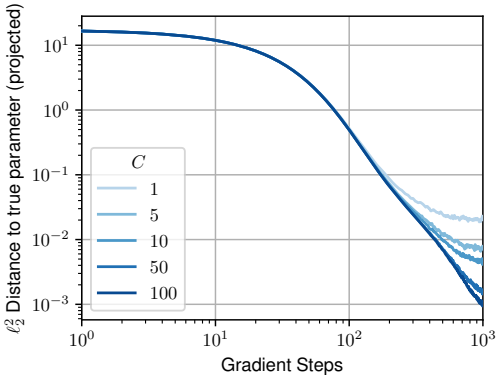


(f) FY loss minus best loss, top-κ polytope

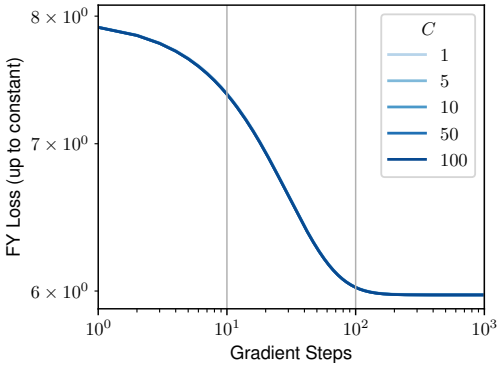
Figure 8: Convergence to the true parameter on the hypercube (left) and the top-κ polytope (right), for varying number of Markov chain iterations K. Longer chains improve learning.



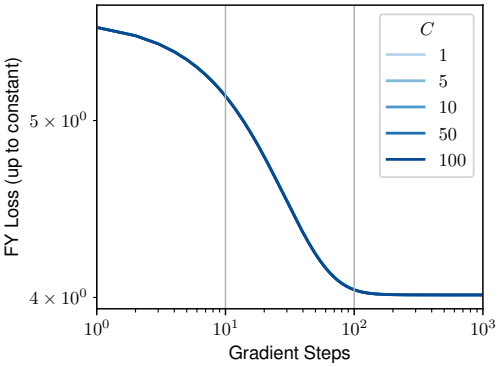
(a) Distance to true parameter, hypercube



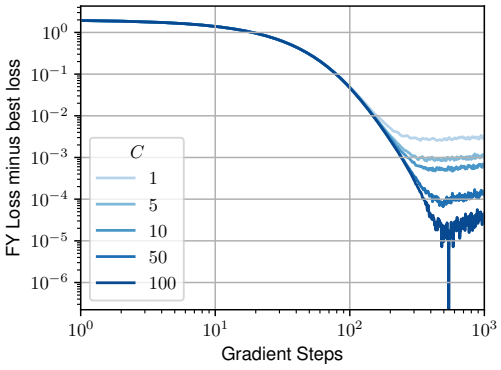
(b) Distance to true parameter, top-κ polytope



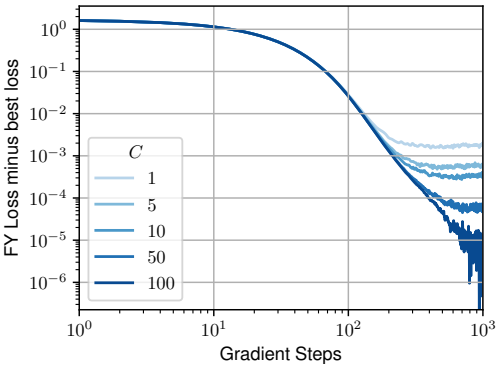
(c) FY loss (up to constant), hypercube



(d) FY loss (up to constant), top-κ polytope

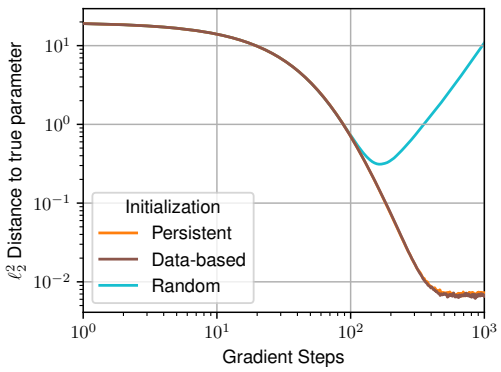


(e) FY loss minus best loss, hypercube

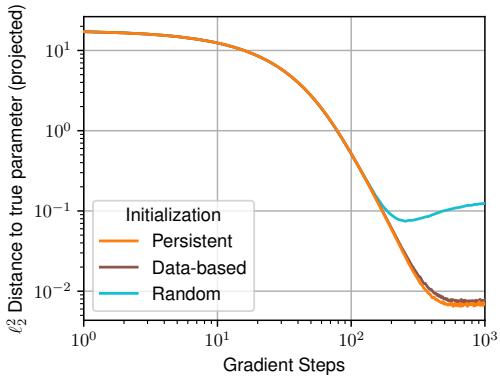


(f) FY loss minus best loss, top-κ polytope

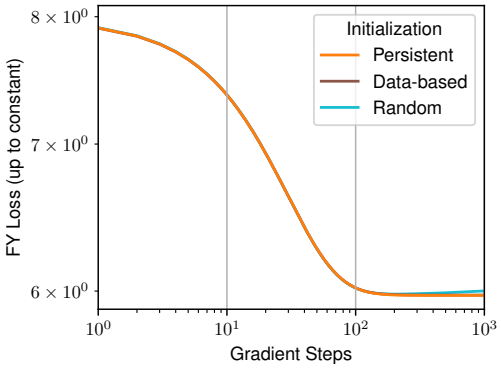
Figure 9: Convergence to the true parameter on the hypercube (left) and the top-κ polytope (right), for varying number of parallel Markov chains C. Adding Markov chains improves estimation.



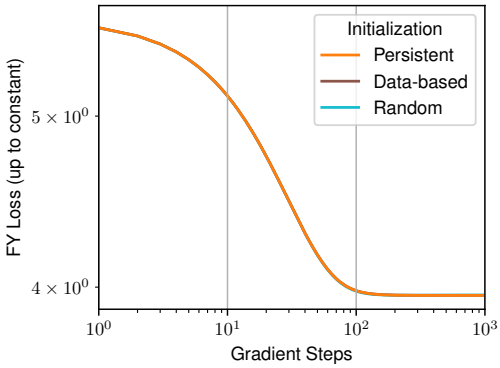
(a) Distance to true parameter, hypercube



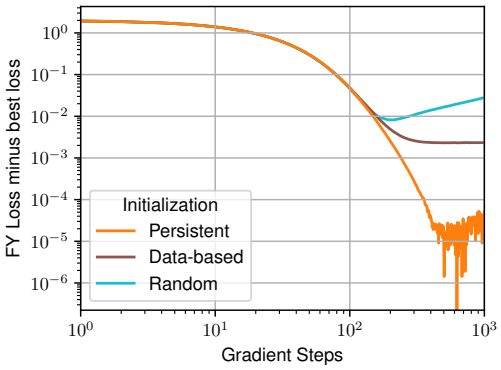
(b) Distance to true parameter, top-κ polytope



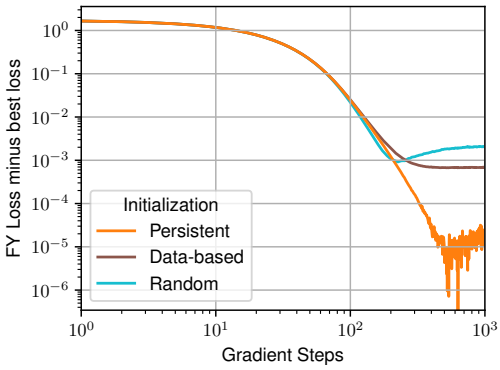
(c) FY loss (up to constant), hypercube



(d) FY loss (up to constant), top-κ polytope

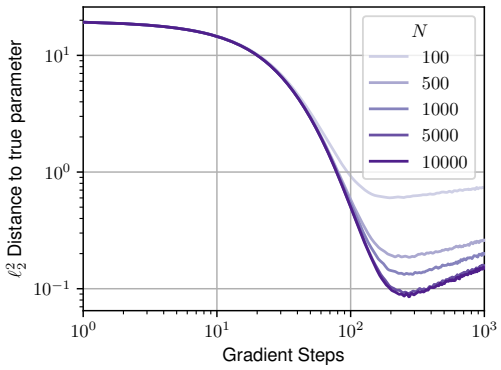


(e) FY loss minus best loss, hypercube

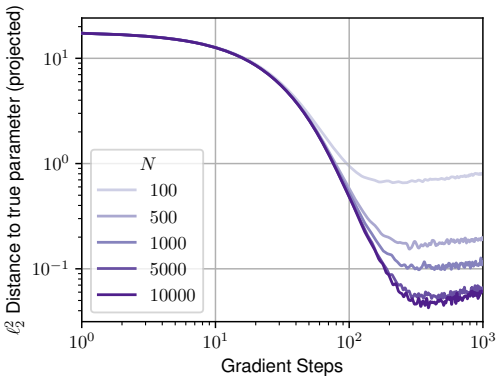


(f) FY loss minus best loss, top-κ polytope

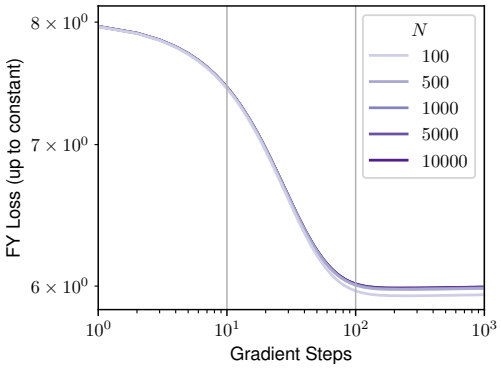
Figure 10: Convergence to the true parameter on the hypercube (left) and the top-κ polytope (right), for varying Markov chain initialization method. The persistent and data-based initialization methods significantly outperform the random initialization method.



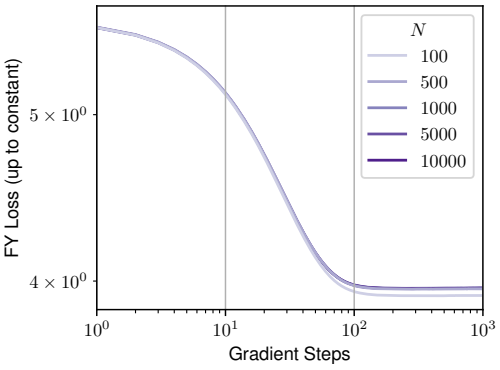
(a) Distance to true parameter



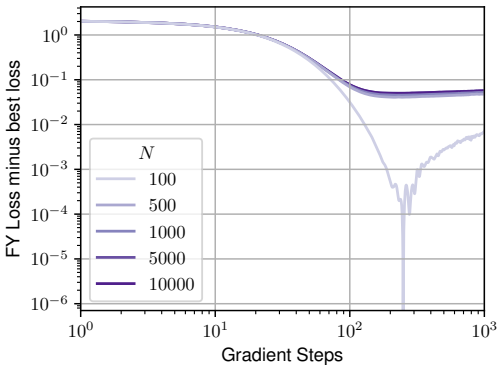
(b) Distance to true parameter



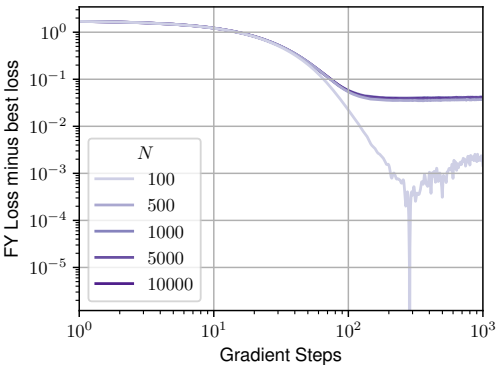
(c) FY loss (up to constant)



(d) Fenchel-Young loss (up to constant)

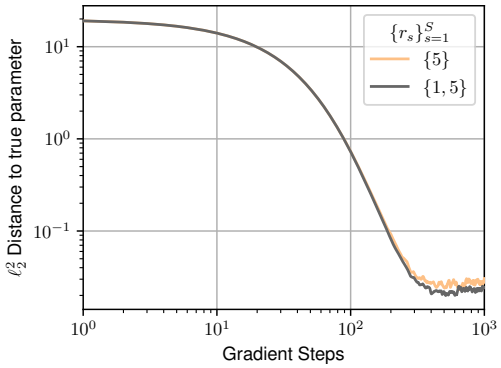


(e) FY loss minus best loss

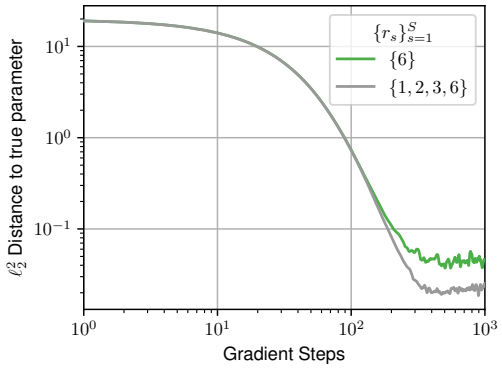


(f) Fenchel-Young loss minus best loss

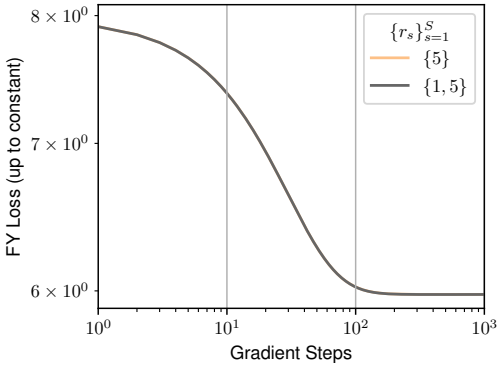
Figure 11: Convergence to the true parameter on the hypercube (left) and the top-κ polytope (right), for varying number of samples N in the dataset. As the dataset is different for each task, the empirical Fenchel-Young loss L_N, which is the minimized objective function (contrary to other experiments, where we minimize $L_{\boldsymbol{\theta}_0}$), also varies. Although empirical Fenchel-Young losses associated to smaller datasets appear easier to minimize, increasing the dataset size reduces the bias and thus the distance to $\boldsymbol{\theta}_0$, as expected.



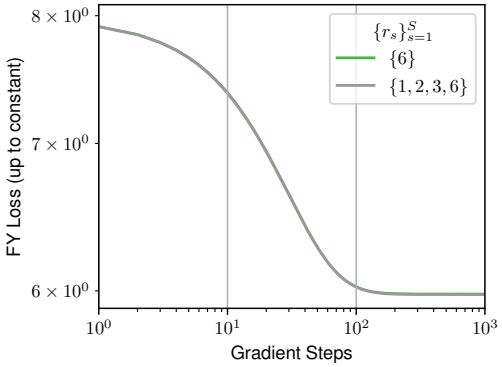
(a) Distance to true parameter, $r_s \in \{5\}$ or $\{1,5\}$



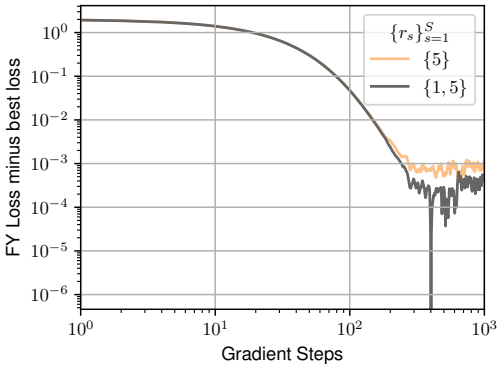
(b) Distance to true parameter, $r_s \in \{6\}$ or $\{1,2,3,6\}$



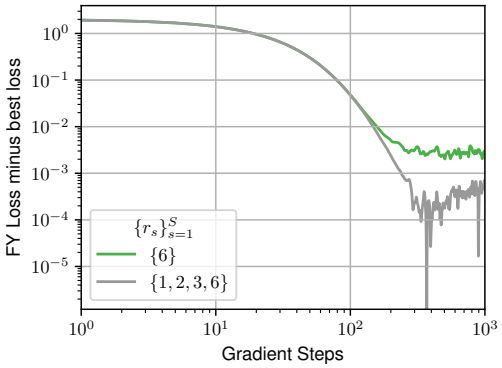
(c) FY loss (up to constant), $r_s \in \{5\}$ or $\{1,5\}$



(d) FY loss (up to constant), $r_s \in \{6\}$ or $\{1,2,3,6\}$



(e) FY loss minus best loss, $r_s \in \{5\}$ or $\{1,5\}$



(f) FY loss minus best loss, $r_s \in \{6\}$ or $\{1,2,3,6\}$

Figure 12: Convergence to the true parameter on the hypercube, with different mixtures of neighborhood systems $\{\mathcal{N}_=^{r_s}\}_{s=1}^S$: comparing $r_s \in \{5\}$ to $r_s \in \{1,5\}$ (left), and comparing $r_s \in \{6\}$ to $r_s \in \{1,2,3,6\}$ (right). Using more neighborhoods in the mixture improves learning.

C Additional material

C.1 Mixing neighborhood systems: a discussion

In this section, we give intuition on why the update proposed in Section 2.2 and Algorithm 2 is crucial as a tractable way to mix different neighborhood systems.

A naive way to combine these neighborhoods $(\mathcal{N}_s)_{s=1}^S$ and proposals $(q_s)_{s=1}^S$ would be to use Algorithm 1 by defining a unique aggregated proposal $q(\boldsymbol{y}, \cdot)$ with support $\mathcal{N}(\boldsymbol{y})$ or $\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}$ as, i.e.:

$$q(\boldsymbol{y}, \boldsymbol{y}') \coloneqq \frac{1}{|Q(\boldsymbol{y})|} \sum_{s \in Q(\boldsymbol{y})} q_s(\boldsymbol{y}, \boldsymbol{y}').$$

However, this would lead to non-tractable updates because of the computation of the Metropolis-Hastings correction ratio. Indeed, the latter would be equal to:

$$\alpha(\boldsymbol{y}, \boldsymbol{y}') = \frac{|Q(\boldsymbol{y})|}{|Q(\boldsymbol{y}')|} \cdot \frac{\sum_{s \in Q(\boldsymbol{y}')} q_s(\boldsymbol{y}', \boldsymbol{y})}{\sum_{s \in Q(\boldsymbol{y})} q_s(\boldsymbol{y}, \boldsymbol{y}')}.$$

This calculation is prohibitively expensive because it involves summing the forward proposal probabilities for all move types in $Q(\boldsymbol{y})$ and the reverse probabilities for all move types in $Q(\boldsymbol{y}')$. The main difficulty is that multiple, distinct proposal types can generate the same solution $\boldsymbol{y}'$ from $\boldsymbol{y}$. For example, in the vehicle routing application presented in Section 4, relocating a pair of clients (using the `relocate pair` move from Table 4) before the first one in a route of 3 gives the same solution $\boldsymbol{y}'$ as relocating the first client (with the `relocate` move) at the last position. Identifying and calculating all these potential forward and reverse pathways for every step is a significant computational hurdle.

In contrast, the update we propose in Algorithm 2 only requires computing the single individual ratio $\frac{q_s(\boldsymbol{y}', \boldsymbol{y})}{q_s(\boldsymbol{y}, \boldsymbol{y}')}$ for the unique move type s that was actually sampled.

C.2 Associated Fenchel-Young loss with a single MCMC iteration

To obtain an unbiased gradient estimator for the Fenchel-Young loss ℓ_t associated with $\widehat{\boldsymbol{y}}_t$, the MCMC sampler must be run until it reaches its stationary distribution $\pi_{\boldsymbol{\theta},t}$. This requirement makes any practical estimator with a finite number of steps K inherently biased.

Although our convergence analysis in Section 3.3 shows that this bias does not hinder the convergence of the proposed learning algorithms, we now demonstrate that when a single MCMC iteration is used ($K = 1$), there exists *another* target-dependent Fenchel-Young loss such that the stochastic gradient estimator is *unbiased* with respect to that loss. See Appendix E.7 for the construction of $\Omega_{\boldsymbol{y}}$ and the proof.

Proposition C.1 (Existence of a Fenchel-Young loss when $K = 1$)**.** *Let $\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}$ denote the distribution of the first iterate of the Markov chain defined by the Markov transition kernel given in Eq. (3), with proposal distribution q and initialized at ground-truth $\boldsymbol{y} \in \mathcal{Y}$. There exists a target-dependent regularization function $\Omega_{\boldsymbol{y}}$ with the following properties: $\Omega_{\boldsymbol{y}}$ is $t/\mathbb{E}_{q(\boldsymbol{y},\cdot)}||Y - \boldsymbol{y}||_2^2$-strongly convex; it is such that*

$$\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y] = \operatorname*{argmax}_{\boldsymbol{\mu} \in \mathsf{conv}(\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\})} \{\langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - \Omega_{\boldsymbol{y}}(\boldsymbol{\mu})\}$$

and the Fenchel-Young loss $\ell_{\Omega_{\boldsymbol{y}}}$ generated by $\Omega_{\boldsymbol{y}}$ is $\mathbb{E}_{q(\boldsymbol{y},\cdot)}||Y - \boldsymbol{y}||_2^2/t$-smooth in its first argument, and such that $\nabla_{\boldsymbol{\theta}} \ell_{\Omega_{\boldsymbol{y}}}(\boldsymbol{\theta}\,; \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y] - \boldsymbol{y}$.

A similar result in the unsupervised setting with data-based initialization is given in Proposition C.2. Interestingly, theses results contrast with prior work on the expected CD-1 update. Indeed, when applied with Gibbs sampling to train restricted Boltzmann machines, the latter was shown in Sutskever and Tieleman [47] not to be the gradient of any function – let alone a convex one.

Note that, similarly to the regularization Ω_t, the target-dependent $\Omega_{\boldsymbol{y}}$ extends the influence of φ from the set $\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}$ to its convex hull in a principled way. As a verification, we give properties of the regularized maximizer $\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y]$ in Proposition C.3.

C.3 Fenchel-Young loss for $K = 1$ in the unsupervised setting

This proposition is analogous to Proposition C.1, but in the unsupervised setting, when using a data-based initialization method – i.e., the original CD initialization scheme, without persistent Markov chains. See Appendix C.5 for a detailed discussion about this.

Proposition C.2. *Let $\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\bar{Y}_N}$ denote the distribution of the first iterate of the Markov chain defined by the Markov transition kernel given in Eq. (3), with proposal distribution q and initialized by $\boldsymbol{y}^{(0)} = \boldsymbol{y}_i$, with $i \sim \mathcal{U}([\![1, N]\!])$. There exists a dataset-dependent regularization $\Omega_{\bar{Y}_N}$ with the following properties: $\Omega_{\bar{Y}_N}$ is $tN/\sum_{i=1}^N \mathbb{E}_{q(\boldsymbol{y}_i,\cdot)}||Y - \boldsymbol{y}_i||_2^2$-strongly convex; it is such that:*

$$\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\bar{Y}_N}}[Y] = \operatorname*{argmax}_{\boldsymbol{\mu}\in \mathsf{conv}\left(\bigcup_{i=1}^N \{\mathcal{N}(\boldsymbol{y}_i)\cup\{\boldsymbol{y}_i\}\}\right)} \left\{\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle - \Omega_{\bar{Y}_N}(\boldsymbol{\mu})\right\};$$

and the Fenchel-Young loss $L_{\Omega_{\bar{Y}_N}}$ generated by $\Omega_{\bar{Y}_N}$ is $\frac{1}{N}\sum_{i=1}^N \mathbb{E}_{q(\boldsymbol{y}_i,\cdot)}||Y - \boldsymbol{y}_i||_2^2/t$-smooth in its first argument, and such that $\nabla_{\boldsymbol{\theta}} L_{\Omega_{\bar{Y}_N}}(\boldsymbol{\theta}\,;\boldsymbol{y}) = \mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\bar{Y}_N}}[Y] - \boldsymbol{y}$.

The proof is given in Appendix E.7.

C.4 Properties of the expected first iterate

Proposition C.3. *Let $\boldsymbol{\theta} \in \mathbb{R}^d$, $\boldsymbol{y} \in \mathcal{Y}$. Let*

$$\mathcal{N}_{better}(\boldsymbol{y}) \coloneqq \{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}) \mid \langle \boldsymbol{\theta}, \boldsymbol{y}'\rangle + \varphi(\boldsymbol{y}') > \langle \boldsymbol{\theta}, \boldsymbol{y}\rangle + \varphi(\boldsymbol{y})\}$$

denote the set of improving neighbors of $\boldsymbol{y}$ for the unregularized objective function. We have the following properties:

$$\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y] \in \mathsf{conv}\left(\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}\right),$$

$$\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y] \xrightarrow[t\to 0^+]{} \boldsymbol{y} + \sum_{\boldsymbol{y}'\in\mathcal{N}_{better}(\boldsymbol{y})} q(\boldsymbol{y},\boldsymbol{y}')\cdot(\boldsymbol{y}' - \boldsymbol{y}),$$

$$\textit{and}\quad \mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y] \xrightarrow[t\to +\infty]{} \boldsymbol{y} + \sum_{\boldsymbol{y}'\in\mathcal{N}(\boldsymbol{y})} \min\left[q(\boldsymbol{y},\boldsymbol{y}'), q(\boldsymbol{y}',\boldsymbol{y})\right]\cdot(\boldsymbol{y}' - \boldsymbol{y}).$$

The proof is given in Appendix E.8. Thus, as the set $\mathcal{N}_{\text{better}}$ is defined according the value of the original, unregularized objective function $\boldsymbol{y} \mapsto \langle \boldsymbol{\theta}, \boldsymbol{y}\rangle + \varphi(\boldsymbol{y})$, the low temperature behavior of the regularized maximizer $\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y]$ effectively reflects the fact that the regularization function $\Omega_{\boldsymbol{y}}$ extends the influence of φ from the vertices $\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}$ to their convex hull.

C.5 Markov chain initialization

In contrastive divergence (CD) learning, the intractable expectation in the log-likelihood gradient is approximated by short-run MCMC, initialized at the data distribution [22] (using a Gibbs sampler in the setting of Restricted Boltzmann Machines).

Here, we note, at the n-th iteration of gradient descent:

$$\nabla_W L_N(\hat{W}_n) \approx \frac{1}{|B_n|}\sum_{i\in B_n} J_W g_{\hat{W}_n}(\boldsymbol{x}_i)\left(\frac{1}{K}\sum_{k=1}^K \boldsymbol{y}_i^{(n+1,\,k)} - \boldsymbol{y}_i\right),$$

for the supervised setting, with B_n being the mini-batch (or full batch) used at iteration n, $\boldsymbol{y}_i$ the ground-truth structure associated to $\boldsymbol{x}_i$ in the dataset, and $\boldsymbol{y}_i^{(n+1,\,k)}$ the k-th iterate of Algorithm 1, with maximization direction $g_{\hat{W}_n}(\boldsymbol{x}_i)$, and initialization point $\boldsymbol{y}_i^{(n+1,\,0)}$. We also note:

$$\nabla_{\boldsymbol{\theta}} L_N(\hat{\boldsymbol{\theta}}_n) \approx \frac{1}{K}\sum_{k=1}^K \boldsymbol{y}^{(n+1,\,k)} - \bar{Y}_N$$

for the unsupervised setting, with $\boldsymbol{y}^{(n+1,\,k)}$ being the k-th iterate of Algorithm 1, with maximization direction $\hat{\boldsymbol{\theta}}_n$, and initialization point $\boldsymbol{y}^{(n+1,\,0)}$.

In CD learning of unconditional EBMs (i.e., in our unsupervised setting), the Markov Chain is initialized at the empirical data distribution [22, 11], as explained earlier. Persistent Contrastive Divergence (PCD) learning [48] modifies CD by maintaining a persistent Markov chain. Thus, instead of initializing the chain from the data distribution in each iteration, the chain continues from its last state in the previous iteration, by setting $\boldsymbol{y}^{(n+1,\,0)} = \boldsymbol{y}^{(n,\,K)}$. This approach aims to provide a better approximation of the model distribution and to reduce the bias introduced by the initialization of the Markov chain in CD. These are two types of informative initialization methods, which aim at reducing the mixing times of the Markov Chains.

However, neither of these can be applied to the supervised (or conditional) setting, as observed in [38] in the context of conditional Restricted Boltzmann Machines (which are a type of EBMs). Indeed, on the one hand, PCD takes advantage of the fact that the parameter $\hat{\boldsymbol{\theta}}$ does not change too much from one iteration to the next, so that a Markov Chain that has reached equilibrium on $\hat{\boldsymbol{\theta}}_n$ is not far from equilibrium on $\hat{\boldsymbol{\theta}}_{n+1}$. This does not hold in the supervised setting, as each $\boldsymbol{x}_i$ leads to a different $\hat{\boldsymbol{\theta}}_i = g_{\hat{W}}(\boldsymbol{x}_i)$. On the other hand, the data-based initialization method in CD would amount to initialize the chains at the empirical marginal data distribution on $\mathcal{Y}$, and would be irrelevant in a supervised setting, since the distribution we want each Markov Chain to approximate is conditioned on the input $\boldsymbol{x}_i$.

An option is to use persistent chains if training for multiple epochs, and to initialize the Markov Chain associated to $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ for epoch j at the final state of the one associated to the same data point $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ at epoch $j-1$. However, this method is relevant than PCD in the unsupervised setting, as $\hat{w}$ changes a lot more in a full epoch than $\hat{\theta}$ in just one gradient step in the unsupervised setting. It might be relevant, however, if each epoch consists in a single, full-batch gradient step. Nevertheless, it would require to store a significant number of states $\boldsymbol{y}_i^{(n,\,K)}$ (one for each point in the dataset). The solution we propose, for both full-batch and mini-batch settings, is to initialize the chains at the empirical data distribution conditioned on the input $\boldsymbol{x}_i$, which amounts to initialize them at the ground-truth $\boldsymbol{y}_i$.

This discussion is summed up in Table 3.

Table 3: Possible Markov Chain Initialization Methods under each Learning Setting

Init. Method \ Setting	**Unsupervised**	**Supervised, Batch**	**Supervised, Mini-Batch**
Persistent	$\boldsymbol{y}^{(n+1,0)} = \boldsymbol{y}^{(n,K)}$	$\boldsymbol{y}_i^{(n+1,\,0)} = \boldsymbol{y}_i^{(n,\,K)}$	/
Data-Based	$\boldsymbol{y}^{(n+1,0)} = \boldsymbol{y}_j$, with $j \sim \mathcal{U}(\llbracket 1, N \rrbracket)$	$\boldsymbol{y}_i^{(n+1,\,0)} = \boldsymbol{y}_i$	$\boldsymbol{y}_i^{(n+1,\,0)} = \boldsymbol{y}_i$
Random	$\boldsymbol{y}^{(n+1,0)} \sim \mathcal{U}(\mathcal{Y})$	$\boldsymbol{y}_i^{(n+1,\,0)} \sim \mathcal{U}(\mathcal{Y})$	$\boldsymbol{y}_i^{(n+1,\,0)} \sim \mathcal{U}(\mathcal{Y})$

Remark C.4. The use of uniform distributions on $\mathcal{Y}$ for the random initialization method can naturally be replaced by any other different prior distribution.

C.6 Proposal distribution design for the DVRPTW

Original deterministic moves. The selected moves, designed for Local Search algorithms on vehicle routing problems (specifically for the PC-VRPTW for `serve request` and `remove request`), are given in Table 4.

Name	Description
`relocate`	removes request i from its route and re-inserts it before or after request j
`relocate pair`	removes pair of requests $(i, \mathsf{next}(i))$ from their route and re-inserts them before or after request j
`swap`	exchanges the position of requests i and j in the solution
`swap pair`	exchanges the positions of the pairs $(i, \mathsf{next}(i))$ and $(j, \mathsf{next}(j))$ in the solution
`2-opt`	reverses the route segment between i and j
`serve request`	inserts currently undispatched request i before or after request j
`remove request`	removes currently dispatched request i from the solution

Table 4: PC-VRPTW Local search moves

Move	$V_s^1(\boldsymbol{y})$	$V_s^2(\boldsymbol{y})[i]$
`relocate`	$\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})$	$\mathcal{D}(\boldsymbol{y})$
`relocate pair`	$\mathcal{D}(\boldsymbol{y}) \setminus \{\mathcal{D}_2(\boldsymbol{y}) \cup \mathcal{D}^{\text{last}}(\boldsymbol{y})\}$	$\mathcal{D}(\boldsymbol{y}) \setminus \{\mathsf{next}(i)\}$
`swap`	$\mathcal{D}(\boldsymbol{y})$	$\mathcal{D}(\boldsymbol{y})$
`swap pair`	$\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}^{\text{last}}(\boldsymbol{y})$	$\mathcal{D}(\boldsymbol{y}) \setminus \{\mathcal{D}^{\text{last}}(\boldsymbol{y}) \cup \{\mathsf{prev}(i), \mathsf{next}(i)\}\}$
`2-opt`	$\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_2(\boldsymbol{y})$	$\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_2(\boldsymbol{y})$
`serve request`	$\overline{\mathcal{D}}(\boldsymbol{y})$	$\mathcal{D}(\boldsymbol{y}) \cup \mathcal{I}_D(\boldsymbol{y})$
`remove request`	$\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})$	

Table 5: Sets of valid clients for each move. $\mathcal{D}(\boldsymbol{y})$ contains all dispatched clients in solution $\boldsymbol{y}$. $\mathcal{D}_1(\boldsymbol{y})$ contains all dispatched clients that are the only client in their route. $\mathcal{D}_2(\boldsymbol{y})$ contains all dispatched clients that are in a route with 2 clients or less. $\mathcal{D}^{\text{last}}(\boldsymbol{y})$ contains all dispatched clients that are the last of their route. $\overline{\mathcal{D}}(\boldsymbol{y})$ contains all non-dispatched clients. $\mathcal{I}_D(\boldsymbol{y})$ contains the depot of the first empty route, if it exists (all routes may be non-empty), or else is the empty set. $\mathcal{I}_1(\boldsymbol{y})$ contains the only client in the last non-empty route if it contains exactly one client, or else is the empty set.

All of these moves (except for `remove request`) involve selecting two clients i and j from the request set $\mathcal{R}^\omega$ (for example, the `relocate` move relocates client i after client j in the solution).

In the Local Search part of the PC-HGS algorithm from Vidal [49], they are implemented as deterministic functions used within a quadratic loop over clients, and are performed only if they improve the solution's objective value. The search is narrowed down to client pairs (i, j) such that $d(i, j)$ is among the N_{prox} lowest values in $\{d(i, k) \mid k \in \mathcal{R}^\omega \setminus \{D, i\}\}$, where d is a problem-specific heuristic distance measure between clients, based on spatial features and time windows, and N_{prox} is a hyperparameter. These distances are independent from the chosen solution routes (they are computed once at the start of the algorithm, from the problem features), non-negative, and symmetric: $d(i, j) = d(j, i)$.

Randomization. In order to transform these deterministic moves into proposals, we first adapt the choice of clients i and j, by sampling i uniformly from $V_s^1(\boldsymbol{y})$, which contains the set of valid choices of client i for move s from solution $\boldsymbol{y}$. Then, we sample j from $V_s^2(\boldsymbol{y})[i] \setminus \{i\}$ using the following softmax distribution: $P_s(j \mid i) = \frac{\exp[-d(i,j)/\beta]}{\sum_{k \in V_s^2(\boldsymbol{y})[i] \setminus \{i\}} \exp[-d(i,k)/\beta]}$, where $\beta > 0$ is a neighborhood sampling temperature. The set $V_s^2(\boldsymbol{y})[i]$ contains all valid choices of client j for move s from solution $\boldsymbol{y}$, and is precised along with $V_s^1(\boldsymbol{y})$ in Table 5. We normalize the distance measures inside the softmax, by dividing them by the maximum distance: $d(i, \cdot) \leftarrow d(i, \cdot) / \max_{k \in V_s^2(\boldsymbol{y})[i] \setminus \{i\}} d(i, k)$.

Neighborhood graph symmetrization. Then, we ensure that each individual neighborhood graph $\mathcal{N}_s$ is undirected. This is already the case for the moves `swap`, `swap pair` and `2-opt`, as they are actually involutions (applying the same move on the same couple (i, j) from $\boldsymbol{y}'$ will result in $\boldsymbol{y}$). However, this is obviously not the case for `serve request` and `remove request`. Indeed, if solution $\boldsymbol{y}'$ is obtained from $\boldsymbol{y}$ by removing a dispatched client (respectively serving an non-dispatched one), $\boldsymbol{y}$ cannot be obtained by removing another one (respectively, serving another one). To fix this, we merge these two moves into a single one. First, it evaluates which of the two moves are allowed (i.e., if they are such that $V_s^1(\boldsymbol{y}) \neq \emptyset$). Then, it samples one (the probability of selecting "remove" is chosen to be equal to the number of removable clients divided by the number of removable clients plus the number of servable clients) in the case where both are possible, or else simply performs the only move allowed. Thus, the corresponding neighborhood graph is undirected as it is always

possible to perform the reverse operation (as when removing a client, it becomes unserved, thus allowing the `serve request` move from $\boldsymbol{y}'$, and vice-versa). We also allow the `serve request` move to insert a client after the depot of the first empty route, to allow the creation of new routes. In consequence, we allow the `remove request` move to remove the only client in the last non-empty route if it contains exactly one client (to maintain symmetry of the neighborhood graph).

For the `relocate` and `relocate pair` moves, the non-reversibility comes from the fact that they only relocate client i (or clients i and $\mathsf{next}(i)$ in the pair case) after client j, so that if client i was the first in its route, relocating it back would be impossible (the depot, which is the start of the route, cannot be selected as j). Thus, we allow insertions before clients too, and add a random choice with probability $(\frac{1}{2}, \frac{1}{2})$ to determine if the relocated client(s) will be inserted before or after j. We also add this feature to the `serve request` move.

Correction ratio computation. Next, we implement the computation of the individual correction ratio $\tilde{\alpha}_s(\boldsymbol{y}, \boldsymbol{y}') = \frac{q_s(\boldsymbol{y}', \boldsymbol{y})}{q_s(\boldsymbol{y}, \boldsymbol{y}')}$ for each proposal q_s.

- In the case of `swap` and `2-opt`, we have $\tilde{\alpha}_s(\boldsymbol{y}, \boldsymbol{y}') = 1$. Indeed, let $\boldsymbol{y}'$ be the result of applying one of these moves s on $\boldsymbol{y}$ when sampling $i \in V_s^1(\boldsymbol{y})$ and $j \in V_s^2(\boldsymbol{y})[i] \setminus \{i\}$. We then have:
$$q_s(\boldsymbol{y}, \boldsymbol{y}') = \frac{1}{|V_s^1(\boldsymbol{y})|} \cdot \frac{\exp\left[-d(i,j)/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y})[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right]} + \frac{1}{|V_s^1(\boldsymbol{y})|} \cdot \frac{\exp\left[-d(j,i)/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y})[j] \setminus \{j\}} \exp\left[-d(j,k)/\beta\right]},$$
where the first term accounts for the probability of selecting i then j and the second term accounts for that of selecting j then i (one can easily check that these two cases are the only way of sampling $\boldsymbol{y}'$ from $\boldsymbol{y}$). Then, noticing that we have $|V_s^1(\boldsymbol{y}')| = |V_s^1(\boldsymbol{y})|$, that these moves are involutions (selecting (i,j) or (j,i) from $\boldsymbol{y}'$ is also the only way to sample $\boldsymbol{y}$), and that we have the equalities $V_s^2(\boldsymbol{y})[i] = V_s^2(\boldsymbol{y}')[i]$ and $V_s^2(\boldsymbol{y})[j] = V_s^2(\boldsymbol{y}')[j]$, we actually have $q_s(\boldsymbol{y}', \boldsymbol{y}) = q_s(\boldsymbol{y}, \boldsymbol{y}')$.
- For `swap pair`, the same arguments hold (leading to the same form for q_s), except for the equalities $V_s^2(\boldsymbol{y})[i] = V_s^2(\boldsymbol{y}')[i]$ and $V_s^2(\boldsymbol{y})[j] = V_s^2(\boldsymbol{y}')[j]$. Thus, we have the following form for the correction ratio:
$$\frac{q_s(\boldsymbol{y}', \boldsymbol{y})}{q_s(\boldsymbol{y}, \boldsymbol{y}')} = \frac{\sum_{k \in V_s^2(\boldsymbol{y})[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right] + \sum_{k \in V_s^2(\boldsymbol{y})[j] \setminus \{j\}} \exp\left[-d(j,k)/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y}')[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right] + \sum_{k \in V_s^2(\boldsymbol{y}')[j] \setminus \{j\}} \exp\left[-d(j,k)/\beta\right]}.$$
- In the case of `relocate`, let j' denote $\mathsf{next}(j)$ if the selected insertion type was "after", and $\mathsf{prev}(j)$ if it was "before" – where $\mathsf{next}(j) \in \mathcal{R}^\omega$ denotes the request following j in solution $\boldsymbol{y}$, i.e., the only index k such that $\boldsymbol{y}_{j,k} = 1$, and $\mathsf{prev}(j)$ is the one preceding it, i.e., the only k such that $\boldsymbol{y}_{k,j} = 1$. We have:
$$q_s(\boldsymbol{y}, \boldsymbol{y}') = \frac{1}{2} \cdot \frac{1}{|V_s^1(\boldsymbol{y})|} \cdot \frac{\exp\left[-d(i,j)/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y})[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right]} + \frac{1}{2} \cdot \frac{1}{|V_s^1(\boldsymbol{y})|} \cdot \frac{\exp\left[-d(i,j')/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y})[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right]}$$
Indeed, if i was relocated *after* j, the same solution $\boldsymbol{y}'$ could have been obtained by relocating i *before* $j' = \mathsf{next}(j)$. Similarly, if i was relocated *before* j, the same solution $\boldsymbol{y}'$ could have been obtained by relocating i *after* $j' = \mathsf{prev}(j)$. For the reverse move probability, the way of obtaining $\boldsymbol{y}$ from $\boldsymbol{y}'$ is either to select $(i, \mathsf{prev}(i))$ in the after-type insertion case, or $(i, \mathsf{next}(i))$ in the before-type insertion case (where prev and next are taken w.r.t. $\boldsymbol{y}$, i.e., before applying the move). Thus, we have:
$$q_s(\boldsymbol{y}', \boldsymbol{y}) = \frac{1}{2} \cdot \frac{1}{|V_s^1(\boldsymbol{y}')|} \cdot \frac{\exp\left[-d(i, \mathsf{prev}(i)/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y}')[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right]} + \frac{1}{2} \cdot \frac{1}{|V_s^1(\boldsymbol{y}')|} \cdot \frac{\exp\left[-d(i, \mathsf{next}(i))/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y}')[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right]}.$$

- For the `relocate pair` move, the exact same reasoning and proposal probability form hold for the forward move, but we have for the reverse direction:

$$q_s(\boldsymbol{y}', \boldsymbol{y}) = \frac{1}{2} \cdot \frac{1}{|V_s^1(\boldsymbol{y}')|} \cdot \frac{\exp\left[-d(i, \mathsf{prev}(i)/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y}')[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right]} + \frac{1}{2} \cdot \frac{1}{|V_s^1(\boldsymbol{y}')|} \cdot \frac{\exp\left[-d(i, \mathsf{next}(\mathsf{next}(i)))/\beta\right]}{\sum_{k \in V_s^2(\boldsymbol{y}')[i] \setminus \{i\}} \exp\left[-d(i,k)/\beta\right]},$$

  as client $\mathsf{next}(i)$ is also relocated.

- For the `serve request` / `remove request` move, we have the forward probability:

$$q_s(\boldsymbol{y}, \boldsymbol{y}') = \frac{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})|}{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})| + |\overline{\mathcal{D}}(\boldsymbol{y})|} \times \frac{1}{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})|} = \frac{1}{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})| + |\overline{\mathcal{D}}(\boldsymbol{y})|}$$

  if the chosen move is `remove request`. The expression corresponds to the composition of move choice sampling and uniform sampling over removable clients.

  Still in the same case (`remove request` is chosen) and if the removed request i was in $\mathcal{I}_1(\boldsymbol{y})$ (i.e., was the only client in the last non-empty route if the latter contained exactly one client), we have the reverse move probability:

$$\begin{aligned} q_s(\boldsymbol{y}', \boldsymbol{y}) &= \frac{1}{|\{\mathcal{D}(\boldsymbol{y}') \setminus \mathcal{D}_1(\boldsymbol{y}')\} \cup \mathcal{I}_1(\boldsymbol{y}')| + |\overline{\mathcal{D}}(\boldsymbol{y}')|} \\ &\quad \times \frac{\exp\left[-\bar{d}(i)/\beta\right]}{\exp\left[-\bar{d}(i)/\beta\right] + \sum_{k \in \mathcal{D}(\boldsymbol{y}')} \exp\left[-d(i,k)/\beta\right]} \\ &= \frac{1}{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})| + |\overline{\mathcal{D}}(\boldsymbol{y})|} \\ &\quad \times \frac{\exp\left[-\bar{d}(i)/\beta\right]}{\exp\left[-\bar{d}(i)/\beta\right] + \sum_{\substack{k \in \mathcal{D}(\boldsymbol{y}) \\ k \neq i}} \exp\left[-d(i,k)/\beta\right]}. \end{aligned}$$

  The expression corresponds to the composition of move choice sampling and softmax sampling of the depot of the first empty route (which was the route of the removed client i, so that $\mathcal{I}_D(\boldsymbol{y}') \neq \emptyset$ in this case). We use the average distance to dispatched clients $\bar{d}(i) \coloneqq \frac{1}{|\mathcal{D}(\boldsymbol{y}')|} \sum_{k \in \mathcal{D}(\boldsymbol{y}')} d(i,k)$ as distance to the depot.

  In the case where the removed request i was not in $\mathcal{I}_1(\boldsymbol{y})$, we have instead:

$$\begin{aligned} q_s(\boldsymbol{y}', \boldsymbol{y}) &= \frac{1}{|\{\mathcal{D}(\boldsymbol{y}') \setminus \mathcal{D}_1(\boldsymbol{y}')\} \cup \mathcal{I}_1(\boldsymbol{y}')| + |\overline{\mathcal{D}}(\boldsymbol{y}')|} \\ &\quad \times \frac{\frac{1}{2} \cdot \exp\left[-d(i, \mathsf{prev}(i))\right] + \frac{1}{2} \cdot \exp\left[-d(i, \mathsf{next}(i))\right]}{\mathbf{1}_{\{\mathcal{I}_D(\boldsymbol{y}') \neq \emptyset\}} \cdot \exp\left[-\bar{d}(i)/\beta\right] + \sum_{k \in \mathcal{D}(\boldsymbol{y}')} \exp\left[-d(i,k)/\beta\right]} \\ &= \frac{1}{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})| + |\overline{\mathcal{D}}(\boldsymbol{y})|} \\ &\quad \times \frac{\frac{1}{2} \cdot \exp\left[-d(i, \mathsf{prev}(i))\right] + \frac{1}{2} \cdot \exp\left[-d(i, \mathsf{next}(i))\right]}{\mathbf{1}_{\{\mathcal{I}_D(\boldsymbol{y}') \neq \emptyset\}} \cdot \exp\left[-\bar{d}(i)/\beta\right] + \sum_{\substack{k \in \mathcal{D}(\boldsymbol{y}) \\ k \neq i}} \exp\left[-d(i,k)/\beta\right]}. \end{aligned}$$

  The right term corresponds to softmax sampling of the previous node with "after" insertion type (which has probability $1/2$) and of the next node with "before" insertion type. The non-emptiness of $\mathcal{I}_D(\boldsymbol{y}')$ is not guaranteed anymore, as all routes might be non-empty (indeed, we did not create an empty one by removing i, as $i \in \mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})$ in this case).

Similarly, if the chosen move is `serve request`, we have the forward probability:

$$q_s(\boldsymbol{y}, \boldsymbol{y}') = \frac{|\overline{\mathcal{D}}(\boldsymbol{y})|}{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})| + |\overline{\mathcal{D}}(\boldsymbol{y})|} \times \frac{\frac{1}{2} \cdot \exp\left[-d(i,j)\right] + \frac{1}{2} \cdot \exp\left[-d(i,j')\right]}{\mathbf{1}_{\{\mathcal{I}_D(\boldsymbol{y}) \neq \emptyset\}} \cdot \exp\left[-\bar{d}(i)/\beta\right] + \sum_{k \in \mathcal{D}(\boldsymbol{y})} \exp\left[-d(i,k)/\beta\right]}$$

if the selected insertion node j is not in $\mathcal{I}_D(\boldsymbol{y})$ (i.e., is not the depot of the first empty route in $\boldsymbol{y}$), where $j' = \mathsf{prev}(j)$ if the insertion type selected was "before" (which has probability $1/2$), and $j' = \mathsf{next}(j)$ if it was "after".

We have instead the forward probability:

$$q_s(\boldsymbol{y}, \boldsymbol{y}') = \frac{1}{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})| + |\overline{\mathcal{D}}(\boldsymbol{y})|} \times \frac{\exp\left[-\bar{d}(i)/\beta\right]}{\exp\left[-\bar{d}(i)/\beta\right] + \sum_{k \in \mathcal{D}(\boldsymbol{y})} \exp\left[-d(i,k)/\beta\right]}$$

if the selected insertion node j is in $\mathcal{I}_D(\boldsymbol{y})$ (i.e., is the depot of the first empty route in $\boldsymbol{y}$).

In every case, we have the reverse move probability:

$$q_s(\boldsymbol{y}', \boldsymbol{y}) = \frac{1}{|\{\mathcal{D}(\boldsymbol{y}) \setminus \mathcal{D}_1(\boldsymbol{y})\} \cup \mathcal{I}_1(\boldsymbol{y})| + |\overline{\mathcal{D}}(\boldsymbol{y})|}.$$

In each case, we set $d(i, D) = +\infty$ to account for the fact that the depot can never be sampled during the process (except in the `serve request` / `remove request` move, where we allow the depot of the first empty route / last non-empty route to be selected, for which we use the average distance to other requests as explained earlier) – in fact, the distance measure from a client to the depot is not even defined in the original HGS implementation.

The second correction factor needed is $\frac{|Q(\boldsymbol{y})|}{|Q(\boldsymbol{y}')|}$ (see Algorithm 2). We compute it by checking if each move is allowed, i.e., if there exists at least one $i \in V_s^1(\boldsymbol{y})$ such that $V_s^2(\boldsymbol{y})[i] \setminus \{i\} \neq \emptyset$. This can be determined in $\mathcal{O}(\mathcal{R}^\omega)$ for each move.

D Details on the DVRPTW

D.1 Overview of the challenge.

We evaluate the proposed approach on a large-scale, ML-enriched combinatorial optimization problem: the *EURO Meets NeurIPS 2022 Vehicle Routing Competition* [27]. In this dynamic vehicle routing problem with time windows (DVRPTW), requests arrive continuously throughout a planning horizon, which is partitioned into a series of delivery waves $\mathcal{W} = \{[\tau_0, \tau_1], [\tau_1, \tau_2], \ldots, [\tau_{|\mathcal{W}|-1}, \tau_{|\mathcal{W}|}]\}$.

At the start of each wave ω, a dispatching and vehicle routing problem must be solved for the set of requests $\mathcal{R}^\omega$ specific to that wave (in which we include the depot D), encoded into the system state $\boldsymbol{x}^\omega$. We note $\mathcal{Y}(\boldsymbol{x}^\omega)$ the set of feasible decisions associated to state $\boldsymbol{x}^\omega$.

A feasible solution $\boldsymbol{y}^\omega \in \mathcal{Y}(\boldsymbol{x}^\omega)$ must contain all requests that must be dispatched before τ_ω (the rest are postponable), allow each of its routes to visit the requests they dispatch within their respective time windows, and be such that the cumulative customer demand on each of its routes does not exceed a given vehicle capacity. It is encoded thanks to a vector $\left(y^\omega_{i,j}\right)_{i,j \in \mathcal{R}^\omega}$, where $y^\omega_{i,j} = 1$ if the solution contains the directed route segment from i to j, and $y^\omega_{i,j} = 0$ otherwise. The set of requests $\mathcal{R}^{\omega+1}$ is obtained by removing all requests dispatched by the chosen solution $\boldsymbol{y}^\omega$ from $\mathcal{R}^\omega$ and adding all new requests which arrived between τ_ω and $\tau_{\omega+1}$.

The aim of the challenge is to find an optimal policy $f \colon \mathcal{X} \to \mathcal{Y}$ assigning decisions $\boldsymbol{y}^\omega \in \mathcal{Y}(\boldsymbol{x}^\omega)$ to system states $\boldsymbol{x}^\omega \in \mathcal{X}$. This can be cast as a reinforcement learning problem:

$$\min_f \mathbb{E}\left[c_{\mathcal{W}}(f)\right], \quad \text{with} \quad c_{\mathcal{W}}(f) \coloneqq \sum_{\omega \in \mathcal{W}} c(f(\boldsymbol{x}^\omega)),$$

where $c : \boldsymbol{y}^\omega \mapsto \sum_{i,j \in \mathcal{R}^\omega} c_{i,j}\, y^\omega_{i,j}$ gives the routing cost of $\boldsymbol{y}^\omega \in \mathcal{Y}^\omega$ and where $c_{i,j} \geq 0$ is the routing cost from i to j. The expectation is taken over full problem instances.

D.2 Reduction to supervised learning.

We follow the method of [4], which was the winning approach for the challenge. Central to this approach is the concept of prize-collecting dynamic vehicle routing problem with time windows (PC-VRPTW). In this setting, each request $i \in \mathcal{R}^\omega$ is assigned an artificial *prize* $\theta_i^\omega \in \mathbb{R}$, that reflects the benefit of serving it. The prize of the depot D is set to $\theta_D^\omega = 0$. The objective is then to identify a set of routes that maximizes the total prize collected while minimizing the associated travel costs. The model g_W predicts the prize vector $\boldsymbol{\theta}^\omega = g_W(\boldsymbol{x}^\omega)$. Denoting $\varphi(\boldsymbol{y}) \coloneqq -\langle \boldsymbol{c}, \boldsymbol{y} \rangle$, the corresponding optimization problem can be written as

$$\widehat{\boldsymbol{y}}(\boldsymbol{\theta}^\omega) = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x}^\omega)} \sum_{i,j \in \mathcal{R}^\omega} \theta_j^\omega \, y_{i,j} - \sum_{i,j \in \mathcal{R}^\omega} c_{i,j} \, y_{i,j} = \langle \boldsymbol{\theta}^\omega, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y}). \tag{10}$$

The overall pipeline is summarized in Fig. 1. Following [4], we approximately solve the problem in Eq. (10) using the prize-collecting HGS heuristic (PC-HGS), a variant of hybrid genetic search (HGS) [49]. We denote this approximate solver $\widetilde{\boldsymbol{y}} \approx \widehat{\boldsymbol{y}}$, so that their proposed policy decomposes as $f_W \coloneqq \widetilde{\boldsymbol{y}} \circ g_W$. The ground-truth routes are created by using an anticipative strategy, i.e., by solving multiple instances where all future information is revealed from the start, and the requests' arrival times information is translated into time windows (thus removing the dynamic aspect of the problem). This anticipative policy, which we note $f^\star$ (which cannot be attained as it needs unavailable information) is thus the target policy imitated by the model – see Appendix D.7 for more details.

D.3 Perturbation-based baseline.

In [4], a perturbation-based method [6] was used. This method is based on injecting noise in the PC-HGS solver $\widetilde{\boldsymbol{y}}$. Similarly to our approach, the parameters W can then be learned using a Fenchel-Young loss, since the loss is minimized when the perturbed solver correctly predicts the ground truth. However, since $\widetilde{\boldsymbol{y}}$ is not an exact solver, all theoretical learning guarantees associated with this method (e.g., correctness of the gradients) no longer hold.

D.4 Proposed approach.

Our proposed approach instead uses the Fenchel-Young loss associated with the proposed layer, which is minimized when the proposed layer correctly predicts the ground-truth. At inference time, however, we use $f_W \coloneqq \widetilde{\boldsymbol{y}} \circ g_W$. We use a mixture of proposals, as defined in Algorithm 2. To design each proposal q_s, we build randomized versions of moves specifically designed for the prize-collecting dynamic vehicle routing problem with time windows. More precisely, we base our proposals on moves used in the local search part of the PC-HGS algorithm, which are summarized in Table 4. The details of turning these moves into proposal distributions with tractable individual correction ratios are given in Appendix C.6.

We evaluate three different initialization methods: (i) initialize $\boldsymbol{y}^{(0)}$ by constructing routes dispatching random requests, (ii) initialize $\boldsymbol{y}^{(0)}$ to the ground-truth solution, (iii) initialize $\boldsymbol{y}^{(0)}$ by starting from the dataset ground-truth and applying a heuristic initialization algorithm to improve it. This heuristic initialization, similar to a short local search, is also used by the PC-HGS algorithm $\widetilde{\boldsymbol{y}}$, and is set to take up to half the time allocated to the layer (a limit it does not reach in practice).

D.5 Performance metric.

As the Fenchel-Young loss ℓ_t actually minimized is intractable to compute exactly, we only use the challenge metric. More precisely, we measure the cost relative to that of the anticipative baseline, $\frac{c_{\mathcal{W}}(f_W) - c_{\mathcal{W}}(f^\star)}{c_{\mathcal{W}}(f^\star)}$, which we average over a test dataset of unseen instances.

D.6 Results.

In Fig. 2, we observe that the initialization method plays an important role, and the ground-truth-based ones greatly outperform the random one.

We observe that the number of Markov iterations K is an important performance factor. Interestingly, the ground-truth initialization significantly improves the learning process for small K.

In Table 2, we compare training methods with fixed compute time budget for the layer (perturbed solver or proposed MCMC approach), which is by far the main computational bottleneck. This parameter limits the time allowed for a single forward pass through the combinatorial optimization layer (be it the perturbed inexact oracle or the proposed method). In both cases, the backward pass through the layer is immediate, as a property of the expression of the gradient of Fenchel-Young losses. The models are selected using a validation set and evaluated on the test set. We observe that the proposed approach significantly outperforms the perturbation-based method [6] using $\widetilde{\boldsymbol{y}}$ in low time limit regimes, thus allowing for faster and more efficient training.

Full experimental details and additional results on the impact of temperature are given in Appendix D.7.

D.7 Additional experimental details and results for Section 4

Model, features, dataset, hyperparameters, compute. Following Baty et al. [4], the differentiable ML model g_W is implemented as a sparse graph neural network. We also use the same feature set, which represents the system state $\boldsymbol{x}^\omega$ as a vector comprising request-level features, such as coordinates, time windows, demands, travel time to the depot, and quantiles from the distribution of the travel time to all other requests (named *complete* feature set, and described in the Table 4 of their paper). We use the same training, validation, and testing datasets, which are created from 30, 15 and 25 problem instances respectively. The training set uses a sample size of 50 requests per wave, while the rest use 100. The solutions in the training dataset, i.e., the examples from the anticipative strategy $f^\star$ imitated by the model, are obtained by solving the corresponding offline VRPTWs using HGS [49] with a time limit of 3600 seconds. During evaluation, the PC-HGS solver $\tilde{\boldsymbol{y}}$ is used with a constant time limit of 60 seconds for all models. We use Adam [25] together with the proposed stochastic gradient estimators, with a learning rate of $5 \cdot 10^{-3}$. Each training is performed using only a single CPU worker. For Fig. 2, we use a temperature $t = 10^2$. For Table 2, we use 1 Monte-Carlo sample for the perturbation-based method and 1 Markov chain for the proposed approach (in order to have a fair comparison: an equal number of oracle calls / equal compute).

Statistical significance. Each training is performed 50 times with the same parameters and different random seeds. Then, the learning curves are averaged, and plotted with a 95% confidence interval. For the results in Table 2, we report the performance of the best model iteration (selected with respect to the validation set) on the test set. This procedure is also averaged over 50 trainings, and reported with 95% confidence intervals.

Additional results. In Fig. 13, we report model performance for varying temperature t. Interestingly, lower temperatures perform better when using random initialization. In the ground-truth initialization setting, a sweet spot is found at $t = 10^2$, but lower temperatures do not particularly decrease performance.

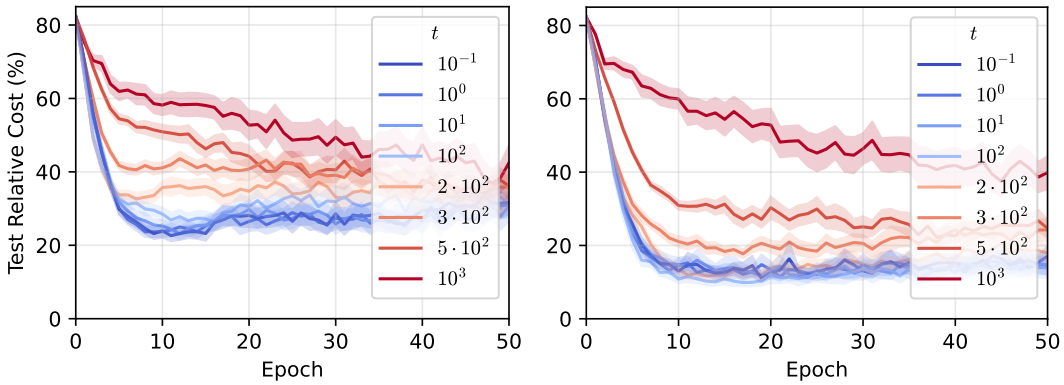


Figure 13: Test relative cost (%). **Left**: varying temperature t with random initialization. **Right**: varying temperature t with ground-truth initialization.

E Proofs

E.1 Proof of Eq. (4)

Proof. At fixed temperature $t_k = t$, the iterates of Algorithm 1 (MH case) follow a time-homogenous Markov chain, defined by the following transition kernel $P_{\boldsymbol{\theta},t}$:

$$P_{\boldsymbol{\theta},t}(\boldsymbol{y},\boldsymbol{y}') = \begin{cases} q\left(\boldsymbol{y},\boldsymbol{y}'\right) \min\left[1, \frac{q(\boldsymbol{y}',\boldsymbol{y})}{q(\boldsymbol{y},\boldsymbol{y}')} \exp\left(\frac{\langle \boldsymbol{\theta}\,,\,\boldsymbol{y}'\rangle + \varphi(\boldsymbol{y}') - \langle \boldsymbol{\theta},\,\boldsymbol{y}\rangle - \varphi(\boldsymbol{y})}{t}\right)\right] & \text{if } \boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}), \\ 1 - \sum_{\boldsymbol{y}'' \in \mathcal{N}(\boldsymbol{y})} P_{\boldsymbol{\theta},t}(\boldsymbol{y},\boldsymbol{y}'') & \text{if } \boldsymbol{y}' = \boldsymbol{y}, \\ 0 & \text{else.} \end{cases}$$

Irreducibility. As we assumed the neighborhood graph $G_{\mathcal{N}}$ to be connected and undirected, the Markov Chain is irreducible as we have $\forall \boldsymbol{y} \in \mathcal{Y}, \forall \boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}), P_{\boldsymbol{\theta},t}(\boldsymbol{y},\boldsymbol{y}') > 0$.

Aperiodicity. For simplicity, we directly assumed aperiodicity in the main text. Here, we show that this is a mild condition, which is verified for instance if there is a solution $\boldsymbol{y} \in \mathcal{Y}$ such that $q(\boldsymbol{y},\boldsymbol{y}) > 0$. Indeed, we then have:

$$\begin{aligned} P_{\boldsymbol{\theta},t}(\boldsymbol{y},\boldsymbol{y}) &= 1 - \sum_{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})} P_{\boldsymbol{\theta},t}(\boldsymbol{y},\boldsymbol{y}') \\ &= 1 - \sum_{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})} q\left(\boldsymbol{y},\boldsymbol{y}'\right) \min\left[1, \frac{q(\boldsymbol{y}',\boldsymbol{y})}{q(\boldsymbol{y},\boldsymbol{y}')} \exp\left(\frac{\langle \boldsymbol{\theta}\,,\,\boldsymbol{y}'\rangle + \varphi(\boldsymbol{y}') - \langle \boldsymbol{\theta},\,\boldsymbol{y}\rangle - \varphi(\boldsymbol{y})}{t}\right)\right] \\ &\geq 1 - \sum_{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})} q\left(\boldsymbol{y},\boldsymbol{y}'\right) \\ &\geq q(\boldsymbol{y},\boldsymbol{y}') \\ &> 0. \end{aligned}$$

Thus, we have $P_{\boldsymbol{\theta},t}(\boldsymbol{y},\boldsymbol{y}) > 0$, which implies that the chain is aperiodic. As an irreducible and aperiodic Markov Chain on a finite state space, it converges to its stationary distribution and the latter is unique [18]. Finally, one can easily check that the detailed balance equation is satisfied for $\pi_{\boldsymbol{\theta},t}$, i.e.:

$$\forall \boldsymbol{y},\boldsymbol{y}' \in \mathcal{Y},\ \pi_{\boldsymbol{\theta},t}(\boldsymbol{y}) P_{\boldsymbol{\theta},t}(\boldsymbol{y},\boldsymbol{y}') = \pi_{\boldsymbol{\theta},t}(\boldsymbol{y}') P_{\boldsymbol{\theta},t}(\boldsymbol{y}',\boldsymbol{y}),$$

giving that $\pi_{\boldsymbol{\theta},t}$ is indeed the stationary distribution of the chain, which concludes the proof. □

E.2 Proof of Proposition 2.1

Proof. Let $\boldsymbol{\theta} \in \mathbb{R}^d$ and $t > 0$. The fact that $\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) \in \mathsf{relint}(\mathcal{C}) = \mathsf{relint}(\mathsf{conv}(\mathcal{Y}))$ follows directly from the fact that $\widehat{\boldsymbol{y}}_t(\boldsymbol{\theta})$ is a convex combination of the elements of $\mathcal{Y}$ with positive coefficients, as $\forall \boldsymbol{y} \in \mathcal{Y},\ \pi_{\boldsymbol{\theta},t}(\boldsymbol{y}) > 0$.

Low temperature limit. Let $\boldsymbol{y}^\star \coloneqq \mathsf{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{\theta}, \boldsymbol{y}\rangle + \varphi(\boldsymbol{y})$. The argmax is assumed to be single-valued. Let $\boldsymbol{y} \in \mathcal{Y} \setminus \{\boldsymbol{y}^\star\}$. We have:

$$\begin{aligned} \pi_{\boldsymbol{\theta},t}(\boldsymbol{y}) &= \frac{\exp\left(\frac{\langle \boldsymbol{\theta},\boldsymbol{y}\rangle + \varphi(\boldsymbol{y})}{t}\right)}{\sum_{\boldsymbol{y}' \in \mathcal{Y}} \exp\left(\frac{\langle \boldsymbol{\theta},\boldsymbol{y}'\rangle + \varphi(\boldsymbol{y}')}{t}\right)} \\ &\leq \frac{\exp\left(\frac{\langle \boldsymbol{\theta},\boldsymbol{y}\rangle + \varphi(\boldsymbol{y})}{t}\right)}{\exp\left(\frac{\langle \boldsymbol{\theta},\boldsymbol{y}^\star\rangle + \varphi(\boldsymbol{y}^\star)}{t}\right)} \\ &\leq \exp\left(\frac{(\langle \boldsymbol{\theta},\boldsymbol{y}\rangle + \varphi(\boldsymbol{y})) - (\langle \boldsymbol{\theta},\boldsymbol{y}^\star\rangle + \varphi(\boldsymbol{y}^\star))}{t}\right) \\ &\xrightarrow[t \to 0^+]{} 0, \end{aligned}$$

as $\langle \boldsymbol{\theta}, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y}) < \langle \boldsymbol{\theta}, \boldsymbol{y}^\star \rangle + \varphi(\boldsymbol{y}^\star)$ by definition of $\boldsymbol{y}^\star$. Thus, we have:

$$\pi_{\boldsymbol{\theta},t}(\boldsymbol{y}^\star) = 1 - \sum_{\boldsymbol{y} \in \mathcal{Y} \setminus \{\boldsymbol{y}^\star\}} \pi_{\boldsymbol{\theta},t}(\boldsymbol{y}) \xrightarrow[t \to 0^+]{} 1.$$

Thus, the expectation of $\pi_{\boldsymbol{\theta},t}$ converges to $\boldsymbol{y}^\star$. Naturally, if the argmax is not unique, the distribution converges to a uniform distribution on the maximizing structures.

High temperature limit. For all $\boldsymbol{y} \in \mathcal{Y}$, we have:

$$\pi_{\boldsymbol{\theta},t}(\boldsymbol{y}) = \frac{\exp\left(\frac{\langle \boldsymbol{\theta}, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y})}{t}\right)}{\sum_{\boldsymbol{y}' \in \mathcal{Y}} \exp\left(\frac{\langle \boldsymbol{\theta}, \boldsymbol{y}' \rangle + \varphi(\boldsymbol{y}')}{t}\right)} \xrightarrow[t \to +\infty]{} \frac{1}{|\mathcal{Y}|},$$

as $\exp(x/t) \xrightarrow[t \to +\infty]{} 1$ for all $x \in \mathbb{R}$. Thus, $\pi_{\boldsymbol{\theta},t}$ converges to the uniform distribution on $\mathcal{Y}$, and its expectation converges to the average of all structures.

Expression of the Jacobian. Let $A_t : \boldsymbol{\theta} \mapsto t \cdot \log \sum_{\boldsymbol{y} \in \mathcal{Y}} \exp\left(\langle \boldsymbol{\theta}, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y})\right)$ be the cumulant function of the exponential family defined by $\pi_{\boldsymbol{\theta},t}$, scaled by t. One can easily check that we have $\nabla_{\boldsymbol{\theta}} A_t(\boldsymbol{\theta}) = \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta})$. Thus, we have $J_{\boldsymbol{\theta}} \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) = \nabla^2_{\boldsymbol{\theta}} A_t(\boldsymbol{\theta})$. However, we also have that the hessian matrix of the cumulant function $\boldsymbol{\theta} \mapsto \frac{1}{t} A_t(\boldsymbol{\theta})$ is equal to the covariance matrix of the random vector $\frac{Y}{t}$ under $\pi_{\boldsymbol{\theta},t}$ [51]. Thus, we have:

$$\begin{aligned} J_{\boldsymbol{\theta}} \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta}) &= \nabla^2_{\boldsymbol{\theta}} A_t(\boldsymbol{\theta}) \\ &= t \cdot \nabla^2_{\boldsymbol{\theta}} \left(\frac{1}{t} A_t(\boldsymbol{\theta})\right) \\ &= t \cdot \mathsf{cov}_{\pi_{\boldsymbol{\theta},t}} \left[\frac{Y}{t}\right] \\ &= \frac{1}{t} \mathsf{cov}_{\pi_{\boldsymbol{\theta},t}} \left[Y\right]. \end{aligned}$$

□

E.3 Proof of Proposition 2.2

Proof. Let $K_{\boldsymbol{\theta},t}$ be the Markov transition kernel associated to Algorithm 2, which can be written as:

$$K_{\boldsymbol{\theta},t}(\boldsymbol{y}, \boldsymbol{y}') = \begin{cases} \sum_{\substack{s \in Q(\boldsymbol{y}) \\ \text{s.t. } q_s(\boldsymbol{y}, \boldsymbol{y}') > 0}} \frac{1}{|Q(\boldsymbol{y})|} q_s(\boldsymbol{y}, \boldsymbol{y}') \min\left(1, \frac{|Q(\boldsymbol{y})|}{|Q(\boldsymbol{y}')|} \cdot \frac{q_s(\boldsymbol{y}', \boldsymbol{y}) \pi_{\theta,t}(\boldsymbol{y}')}{q_s(\boldsymbol{y}, \boldsymbol{y}') \pi_{\theta,t}(\boldsymbol{y})}\right) & \text{if } \boldsymbol{y}' \in \bar{\mathcal{N}}(\boldsymbol{y}), \\ 1 - \sum_{\boldsymbol{y}'' \in \bar{\mathcal{N}}(\boldsymbol{y})} K_{\boldsymbol{\theta},t}(\boldsymbol{y}, \boldsymbol{y}'') & \text{if } \boldsymbol{y}' = \boldsymbol{y}, \\ 0 & \text{else.} \end{cases}$$

As $\forall \boldsymbol{y} \in \mathcal{Y}$, $\forall \boldsymbol{y}' \in \bar{\mathcal{N}}(\boldsymbol{y})$, $K_{\boldsymbol{\theta},t}(\boldsymbol{y}, \boldsymbol{y}') > 0$, the irreducibility of the chain on $\mathcal{Y}$ is directly implied by the connectedness of $G_{\bar{\mathcal{N}}}$.

Thus, we only have to check that the detailed balance equation $\pi_{\theta,t}(\boldsymbol{y}) K_{\boldsymbol{\theta},t}(\boldsymbol{y}, \boldsymbol{y}') = \pi_{\theta,t}(\boldsymbol{y}') K_{\boldsymbol{\theta},t}(\boldsymbol{y}', \boldsymbol{y})$ is satisfied for all $\boldsymbol{y}' \in \bar{\mathcal{N}}(\boldsymbol{y})$. We have:

$$\pi_{\theta,t}(\boldsymbol{y}) K_{\boldsymbol{\theta},t}(\boldsymbol{y}, \boldsymbol{y}') = \sum_{\substack{s \in Q(\boldsymbol{y}) \\ \text{s.t. } q_s(\boldsymbol{y}, \boldsymbol{y}') > 0}} \left[\frac{q_s(\boldsymbol{y}, \boldsymbol{y}') \pi_{\theta,t}(\boldsymbol{y})}{|Q(\boldsymbol{y})|} \min\left(1, \frac{|Q(\boldsymbol{y})|}{|Q(\boldsymbol{y}')|} \cdot \frac{q_s(\boldsymbol{y}', \boldsymbol{y}) \pi_{\theta,t}(\boldsymbol{y}')}{q_s(\boldsymbol{y}, \boldsymbol{y}') \pi_{\theta,t}(\boldsymbol{y})}\right)\right].$$

The main point consists in noticing that the undirectedness assumption for each neighborhood graph $G_{\mathcal{N}_s}$ implies:

$$\{s \in Q(\boldsymbol{y}) : q_s(\boldsymbol{y}, \boldsymbol{y}') > 0\} = \{s \in Q(\boldsymbol{y}') : q_s(\boldsymbol{y}', \boldsymbol{y}) > 0\}.$$

Thus, a simple case analysis on how $|Q(\boldsymbol{y})| q_s(\boldsymbol{y}', \boldsymbol{y}) \pi_{\theta,t}(\boldsymbol{y}')$ and $|Q(\boldsymbol{y}')| q_s(\boldsymbol{y}, \boldsymbol{y}') \pi_{\theta,t}(\boldsymbol{y})$ compare allows us to observe that the expression of $\pi_{\theta,t}(\boldsymbol{y}) K_{\boldsymbol{\theta},t}(\boldsymbol{y}, \boldsymbol{y}')$ is symmetric in $\boldsymbol{y}$ and $\boldsymbol{y}'$, which concludes the proof. □

E.4 Proof of strict convexity

Proof. As A_t is a differentiable convex function on $\mathbb{R}^d$ (as the log-sum-exp of such functions), it is an essentially smooth closed proper convex function. Thus, it is such that

$$\mathsf{relint}\left(\mathsf{dom}((A_t)^*)\right) \subseteq \nabla A_t(\mathbb{R}^d) \subseteq \mathsf{dom}((A_t)^*),$$

and we have that the restriction of $(A_t)^*$ to $\nabla A_t(\mathbb{R}^d)$ is strictly convex on every convex subset of $\nabla A_t(\mathbb{R}^d)$ (corollary 26.4.1 in Rockafellar [42]). As the range of the gradient of the cumulant function $\boldsymbol{\theta} \mapsto A_t(\boldsymbol{\theta})/t$ is exactly the relative interior of the marginal polytope $\mathsf{conv}\left(\{\boldsymbol{y}/t, \boldsymbol{y} \in \mathcal{Y}\}\right)$ (see appendix B.1 in Wainwright and Jordan [51]), and $(A_t)^* =: \Omega_t$, we actually have that

$$\mathsf{relint}\left(\mathsf{dom}(\Omega_t)\right) \subseteq \mathsf{relint}(\mathcal{C}) \subseteq \mathsf{dom}(\Omega_t),$$

and that Ω_t is stricly convex on every convex subset of $\mathsf{relint}(\mathcal{C})$, i.e., strictly convex on $\mathsf{relint}(\mathcal{C})$ (as $\mathsf{relint}(\mathcal{C})$ is itself convex).

As A_t is closed proper convex, it is equal to its biconjugate by the Fenchel-Moreau theorem. Thus, we have:

$$A_t(\boldsymbol{\theta}) = \sup_{\boldsymbol{\mu} \in \mathbb{R}^d} \left\{\langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - (A_t)^*(\boldsymbol{\mu})\right\} = \sup_{\boldsymbol{\mu} \in \mathbb{R}^d} \left\{\langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - \Omega_t(\boldsymbol{\mu})\right\}.$$

Moreover, as $\nabla A_t(\mathbb{R}^d) = \mathsf{relint}(\mathcal{C})$, we have $||\nabla A_t(\boldsymbol{\theta})|| \leq R_{\mathcal{C}} := \max_{\boldsymbol{\mu} \in \mathcal{C}} ||\boldsymbol{\mu}||$, which gives $\mathsf{dom}(\Omega_t) \subset B(\mathbf{0}, R_{\mathcal{C}})$. Thus we can actually write:

$$A_t(\boldsymbol{\theta}) = \sup_{\boldsymbol{\mu} \in B(\mathbf{0}, R_{\mathcal{C}})} \left\{\langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - \Omega_t(\boldsymbol{\mu})\right\},$$

and now apply Danksin's theorem as $B(\mathbf{0}, R_{\mathcal{C}})$ is compact, which further gives:

$$\partial A_t(\boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\mu} \in B(\mathbf{0}, R_{\mathcal{C}})} \left\{\langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - \Omega_t(\boldsymbol{\mu})\right\},$$

and the fact that A_t is differentiable gives that both sides are single-valued. Moreover, as $\nabla A_t(\mathbb{R}^d) = \mathsf{relint}(\mathcal{C})$, we know that the right hand side is maximized in $\mathcal{C}$, and we can actually write:

$$\nabla A_t(\boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\mu} \in \mathcal{C}} \left\{\langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - \Omega_t(\boldsymbol{\mu})\right\}.$$

We end this proof by noting that a simple calculation yields $\nabla A_t(\boldsymbol{\theta}) = \mathbb{E}_{\pi_{\boldsymbol{\theta},t}}[Y] = \widehat{\boldsymbol{y}}_t(\boldsymbol{\theta})$. The expression of $\nabla_{\boldsymbol{\theta}} \ell_t(\boldsymbol{\theta}\,; \boldsymbol{y})$ follows. □

Remark E.1. The proposed Fenchel-Young loss can also be obtained via distribution-space regularization. Let $s_{\boldsymbol{\theta}} := \left(\langle \boldsymbol{\theta}\,, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y})\right)_{\boldsymbol{y} \in \mathcal{Y}} \in \mathbb{R}^{|\mathcal{Y}|}$ be a vector containing the score of all structures, and $L_{-tH} : \mathbb{R}^{|\mathcal{Y}|} \times \Delta^{|\mathcal{Y}|} \to \mathbb{R}$ be the Fenchel-Young loss generated by $-tH$, where H is the Shannon entropy. We have $\nabla_{s_{\boldsymbol{\theta}}}(-tH)^*(s_{\boldsymbol{\theta}}) = \pi_{\boldsymbol{\theta},t}$. The chain rule further gives $\nabla_{\boldsymbol{\theta}}(-tH)^*(s_{\boldsymbol{\theta}}) = \mathbb{E}_{\pi_{\boldsymbol{\theta},t}}[Y]$. Thus, we have $\nabla_{\boldsymbol{\theta}} L_{-tH}(s_{\boldsymbol{\theta}}\,; \boldsymbol{p}_{\boldsymbol{y}}) = \nabla_{\boldsymbol{\theta}} \ell_t(\boldsymbol{\theta}\,; \boldsymbol{y})$, where $\boldsymbol{p}_{\boldsymbol{y}}$ is the dirac distribution on $\boldsymbol{y}$. In the case where $\varphi \equiv 0$ and $t = 1$, we have $\Omega_t(\boldsymbol{\mu}) = -\left(\max_{\boldsymbol{p} \in \Delta^{|\mathcal{Y}|}} H^s(\boldsymbol{p}) \text{ s.t. } \mathbb{E}_{\boldsymbol{p}}[Y] = \boldsymbol{\mu}\right)$, with H^s the Shannon entropy [8], and ℓ_t is known as the CRF loss [29].

E.5 Proof of Proposition 3.1

Proof. The proof is exactly the proof of Proposition 4.1 in Berthet et al. [6], in which the setting is similar, and all the same arguments hold (we also have that $\pi_{\boldsymbol{\theta}_0}$ is dense on $\mathcal{Y}$, giving $\bar{Y}_N \in \mathsf{relint}(\mathcal{C})$ for N large enough). The only difference is the choice of regularization function, and we have to prove that it is also convex and smooth in our case. While the convexity of Ω_t is directly implied by its definition as a Fenchel conjugate, the fact that is is smooth is due to Theorem 26.3 in Rockafellar [42] and the essential strict convexity of A_t (which is itself closed proper convex). The latter relies on the fact that $\mathcal{C}$ is assumed to be of full-dimension (otherwise A_t would be linear when restricted to any affine subspace of direction equal to the subspace orthogonal to the direction of the smallest affine subspace spanned by $\mathcal{C}$), which in turn implies that A_t is strictly convex on $\mathbb{R}^d$. Thus, Proposition 4.1 in Berthet et al. [6] gives the asymptotic normality:

$$\sqrt{N}(\boldsymbol{\theta}^\star_N - \boldsymbol{\theta}_0) \xrightarrow[N \to \infty]{\mathcal{D}} \mathcal{N}\left(\mathbf{0},\, \left(\nabla^2_{\boldsymbol{\theta}} A_t(\boldsymbol{\theta}_0)\right)^{-1} \mathsf{cov}_{\pi_{\boldsymbol{\theta}_0,t}}[Y] \left(\nabla^2_{\boldsymbol{\theta}} A_t(\boldsymbol{\theta}_0)\right)^{-1}\right).$$

Moreover, we already derived $\nabla^2_{\boldsymbol{\theta}} A_t(\boldsymbol{\theta}_0) = \frac{1}{t} \mathsf{cov}_{\pi_{\boldsymbol{\theta}_0,t}}[Y]$ in Appendix E.2, leading to the simplified asymptotic normality given in the proposition.

□

E.6 Proof of Proposition 3.2

Proof. The proof consists in bounding the convergence rate of the Markov chain $\left(\boldsymbol{y}^{(k)}\right)_{k\in\mathbb{N}}$ (which has transition kernel $P_{\boldsymbol{\theta},t}$) for all $\boldsymbol{\theta}$, in order to apply Theorem 4.1 in Younes [52]. It is defined as the smallest constant $\lambda_{\boldsymbol{\theta}}$ such that:

$$\exists A > 0 \,:\, \forall \boldsymbol{y} \in \mathcal{Y},\, |\mathbb{P}(\boldsymbol{y}^{(k)} = \boldsymbol{y}) - \pi_{\boldsymbol{\theta},t}(\boldsymbol{y})| \leq A\lambda_{\boldsymbol{\theta}}^k.$$

More precisely, we must find a constant D such that $\exists B > 0 \,:\, \lambda_{\boldsymbol{\theta}} \leq 1 - Be^{-D||\boldsymbol{\theta}||}$, in order to impose $K_{n+1} > \left\lfloor 1 + a' \exp\left(2D||\hat{\boldsymbol{\theta}}_n||\right)\right\rfloor$.

A known result gives $\lambda_{\boldsymbol{\theta}} \leq \rho(\boldsymbol{\theta})$ with $\rho(\boldsymbol{\theta}) = \max_{\lambda \in S_{\boldsymbol{\theta}} \setminus \{1\}} |\lambda|$ [32], where $S_{\boldsymbol{\theta}}$ is the spectrum of the transition kernel $P_{\boldsymbol{\theta},t}$ (here, $1 - \rho(\boldsymbol{\theta})$ is known as the *spectral gap* of the Markov chain). To bound $\rho(\boldsymbol{\theta})$, we use the results of Ingrassia [23], which study the Markov chain with transition kernel $P'_{\boldsymbol{\theta},t}$, such that $P_{\boldsymbol{\theta},t} = \frac{1}{2}\left(I + P'_{\boldsymbol{\theta},t}\right)$. It corresponds to the same algorithm, but with a proposal distribution q' defined as:

$$q'\left(\boldsymbol{y}, \boldsymbol{y}'\right) = \begin{cases} \frac{1}{d^*} & \text{if } \boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}), \\ 1 - \frac{d(\boldsymbol{y})}{d^*} & \text{if } \boldsymbol{y}' = \boldsymbol{y}, \\ 0 & \text{else.} \end{cases}$$

As $P'_{\boldsymbol{\theta},t}$ is a row-stochastic matrix, Gershgorin's circle theorem gives that its spectrum is included in the complex unit disc. Moreover, one can easily check that the associated Markov chain is also reversible with respect to $\pi_{\boldsymbol{\theta},t}$, and the corresponding detailed balance equation gives:

$$\forall \boldsymbol{y}, \boldsymbol{y}' \in \mathcal{Y},\; \pi_{\boldsymbol{\theta},t}(\boldsymbol{y}) P'_{\boldsymbol{\theta},t}(\boldsymbol{y}, \boldsymbol{y}') = \pi_{\boldsymbol{\theta},t}(\boldsymbol{y}') P'_{\boldsymbol{\theta},t}(\boldsymbol{y}', \boldsymbol{y}),$$

which is equivalent to:

$$\forall \boldsymbol{y}, \boldsymbol{y}' \in \mathcal{Y},\; \sqrt{\frac{\pi_{\boldsymbol{\theta},t}(\boldsymbol{y})}{\pi_{\boldsymbol{\theta},t}(\boldsymbol{y}')}} P'_{\boldsymbol{\theta},t}(\boldsymbol{y}, \boldsymbol{y}') = \sqrt{\frac{\pi_{\boldsymbol{\theta},t}(\boldsymbol{y}')}{\pi_{\boldsymbol{\theta},t}(\boldsymbol{y})}} P'_{\boldsymbol{\theta},t}(\boldsymbol{y}', \boldsymbol{y})$$

as $\pi_{\boldsymbol{\theta},t}$ has full support on $\mathcal{Y}$, which can be further written in matrix form as:

$$\Pi_{\boldsymbol{\theta}}^{1/2} P'_{\boldsymbol{\theta},t} \Pi_{\boldsymbol{\theta}}^{-1/2} = \Pi_{\boldsymbol{\theta}}^{-1/2} P'^{\top}_{\boldsymbol{\theta},t} \Pi_{\boldsymbol{\theta}}^{1/2},$$

where $\Pi_{\boldsymbol{\theta}} = \mathsf{diag}(\pi_{\boldsymbol{\theta};t})$. Thus, the matrix $\Pi_{\boldsymbol{\theta}}^{1/2} P'_{\boldsymbol{\theta},t} \Pi_{\boldsymbol{\theta}}^{-1/2}$ is symmetric, and the spectral theorem ensures its eigenvalues are real. As it is similar to the transition kernel $P'_{\boldsymbol{\theta},t}$ (with change of basis matrix $\Pi_{\boldsymbol{\theta}}^{-1/2}$), they share the same spectrum $S'_{\boldsymbol{\theta}}$, and we have $S'_{\boldsymbol{\theta}} \subset [-1, 1]$. Let us order $S'_{\boldsymbol{\theta}}$ as $-1 \leq \lambda'_{\min} \leq \cdots \leq \lambda'_2 \leq \lambda'_1 = 1$. As $P_{\boldsymbol{\theta},t} = \frac{1}{2}\left(I + P'_{\boldsymbol{\theta},t}\right)$, we clearly have $\rho(\boldsymbol{\theta}) = \frac{1+\lambda'_2}{2}$. Thus, we can use Theorem 4.1 of Ingrassia [23], which gives $\lambda'_2 \leq 1 - G \cdot Z(\boldsymbol{\theta}) \exp(-m\,(\boldsymbol{\theta}))$ (we keep their notations for Z and m, and add the dependency in $\boldsymbol{\theta}$ for clarity), where G is a constant depending only on the graph $G_{\mathcal{N}}$, and with:

$$\begin{aligned} Z(\boldsymbol{\theta}) &= \sum_{\boldsymbol{y}\in\mathcal{Y}} \exp\left(\frac{\langle \boldsymbol{\theta},\, \boldsymbol{y}\rangle + \varphi(\boldsymbol{y})}{t} - \max_{\boldsymbol{y}'\in\mathcal{Y}}\left[\frac{\langle \boldsymbol{\theta},\, \boldsymbol{y}'\rangle + \varphi(\boldsymbol{y}')}{t}\right]\right) \\ &\geq |\mathcal{Y}| \exp\left(\frac{1}{t}\left[\min_{\boldsymbol{y}\in\mathcal{Y}}\langle \boldsymbol{\theta},\, \boldsymbol{y}\rangle + \min_{\boldsymbol{y}\in\mathcal{Y}}\varphi(\boldsymbol{y}) - \max_{\boldsymbol{y}'\in\mathcal{Y}}\langle \boldsymbol{\theta},\, \boldsymbol{y}'\rangle - \max_{\boldsymbol{y}'\in\mathcal{Y}}\varphi(\boldsymbol{y}')\right]\right) \\ &\geq |\mathcal{Y}| \exp\left(-\frac{2R_{\mathcal{C}}}{t}||\boldsymbol{\theta}|| - \frac{2R_{\varphi}}{t}\right), \end{aligned}$$

and:

$$
\begin{aligned}
m(\boldsymbol{\theta}) &\leq \max_{\boldsymbol{y}\in\mathcal{Y}}\left\{\max_{\boldsymbol{y}'\in\mathcal{Y}}\left[\frac{\langle\boldsymbol{\theta},\,\boldsymbol{y}'\rangle+\varphi(\boldsymbol{y}')}{t}\right]-\frac{\langle\boldsymbol{\theta},\boldsymbol{y}\rangle+\varphi(\boldsymbol{y})}{t}\right\}-2\min_{\boldsymbol{y}\in\mathcal{Y}}\left\{\max_{\boldsymbol{y}'\in\mathcal{Y}}\left[\frac{\langle\boldsymbol{\theta},\,\boldsymbol{y}'\rangle+\varphi(\boldsymbol{y}')}{t}\right]-\frac{\langle\boldsymbol{\theta},\boldsymbol{y}\rangle+\varphi(\boldsymbol{y})}{t}\right\}\\
&=\max_{\boldsymbol{y}'\in\mathcal{Y}}\left[\frac{\langle\boldsymbol{\theta},\,\boldsymbol{y}'\rangle+\varphi(\boldsymbol{y}')}{t}\right]-\min_{\boldsymbol{y}\in\mathcal{Y}}\left[\frac{\langle\boldsymbol{\theta},\,\boldsymbol{y}\rangle+\varphi(\boldsymbol{y})}{t}\right]\\
&\leq\frac{1}{t}\left(\max_{\boldsymbol{y}'\in\mathcal{Y}}\langle\boldsymbol{\theta},\,\boldsymbol{y}'\rangle+\max_{\boldsymbol{y}'\in\mathcal{Y}}\varphi(\boldsymbol{y}')-\min_{\boldsymbol{y}\in\mathcal{Y}}\langle\boldsymbol{\theta},\,\boldsymbol{y}\rangle-\min_{\boldsymbol{y}\in\mathcal{Y}}\varphi(\boldsymbol{y})\right)\\
&\leq\frac{2R_{\mathcal{C}}}{t}||\boldsymbol{\theta}||+\frac{2R_{\varphi}}{t},
\end{aligned}
$$

where $R_{\mathcal{C}}=\max_{\boldsymbol{y}\in\mathcal{Y}}||\boldsymbol{y}||$ and $R_{\varphi}=\max_{\boldsymbol{y}\in\mathcal{Y}}|\varphi(\boldsymbol{y})|$. Thus, we have:

$$
\lambda_2'\leq 1-G|\mathcal{Y}|\exp\left(-\frac{4R_{\varphi}}{t}\right)\exp\left(-\frac{4R_{\mathcal{C}}}{t}||\boldsymbol{\theta}||\right),
$$

and finally:

$$
\lambda_{\boldsymbol{\theta}}\leq 1-\frac{G|\mathcal{Y}|\exp\left(-\frac{4R_{\varphi}}{t}\right)}{2}\exp\left(-\frac{4R_{\mathcal{C}}}{t}||\boldsymbol{\theta}||\right),
$$

so taking $D=4R_{\mathcal{C}}/t$ concludes the proof.

□

Remark E.2. The stationary distribution in Ingrassia [23] is defined as proportional to $\exp(-H(\boldsymbol{y}))$, with the assumption that the function H is such that $\min_{\boldsymbol{y}\in\mathcal{Y}}H(\boldsymbol{y})=0$. Thus, we apply their results with

$$
H(\boldsymbol{y})\coloneqq\max_{\boldsymbol{y}'\in\mathcal{Y}}\left[\frac{\langle\boldsymbol{\theta},\,\boldsymbol{y}'\rangle+\varphi(\boldsymbol{y}')}{t}\right]-\frac{\langle\boldsymbol{\theta},\boldsymbol{y}\rangle+\varphi(\boldsymbol{y})}{t}
$$

(which gives correct distribution $\pi_{\boldsymbol{\theta},t}$ and respects this assumption), hence the obtained forms for $Z(\boldsymbol{\theta})$ and the upper bound on $m(\boldsymbol{\theta})$.

E.7 Proofs of Proposition C.1 and Proposition C.2

Proposition C.1. The distribution of the first iterate of the Markov chain with transition kernel defined in Eq. (3) and initialized at the ground-truth structure $\boldsymbol{y}$ is given by:

$$
\begin{aligned}
(\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}})(\boldsymbol{y}')&=P_{\boldsymbol{\theta},t}(\boldsymbol{y},\boldsymbol{y}')\\
&=\begin{cases}
q(\boldsymbol{y},\boldsymbol{y}')\min\left[1,\frac{q(\boldsymbol{y}',\boldsymbol{y})}{q(\boldsymbol{y},\boldsymbol{y}')}\exp\left(\left[\langle\boldsymbol{\theta},\boldsymbol{y}'-\boldsymbol{y}\rangle+\varphi(\boldsymbol{y}')-\varphi(\boldsymbol{y})\right]/t\right)\right] & \text{if } \boldsymbol{y}'\in\mathcal{N}(\boldsymbol{y}),\\
1-\sum_{\boldsymbol{y}''\in\mathcal{N}(\boldsymbol{y})}(\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}})(\boldsymbol{y}'') & \text{if } \boldsymbol{y}'=\boldsymbol{y},\\
0 & \text{else.}
\end{cases}
\end{aligned}
$$

Let $\alpha_{\boldsymbol{y}}(\boldsymbol{\theta},\boldsymbol{y}')\coloneqq\frac{q(\boldsymbol{y}',\boldsymbol{y})}{q(\boldsymbol{y},\boldsymbol{y}')}\exp\left(\left[\langle\boldsymbol{\theta},\boldsymbol{y}'-\boldsymbol{y}\rangle+\varphi(\boldsymbol{y}')-\varphi(\boldsymbol{y})\right]/t\right)$. Define also the following sets:

$$
\mathcal{N}^{-}_{\boldsymbol{y}}(\boldsymbol{\theta})=\left\{\boldsymbol{y}'\in\mathcal{N}(\boldsymbol{y})\mid\alpha_{\boldsymbol{y}}(\boldsymbol{\theta},\boldsymbol{y}')\leq 1\right\},\quad\mathcal{N}^{+}_{\boldsymbol{y}}(\boldsymbol{\theta})=\left\{\boldsymbol{y}'\in\mathcal{N}(\boldsymbol{y})\mid\alpha_{\boldsymbol{y}}(\boldsymbol{\theta},\boldsymbol{y}')>1\right\}.
$$

The expectation of the first iterate is then given by:

$$
\begin{aligned}
\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y]&=\sum_{\boldsymbol{y}'\in\mathcal{N}(\boldsymbol{y})}(\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}})(\boldsymbol{y}')\cdot\boldsymbol{y}'+\left(1-\sum_{\boldsymbol{y}''\in\mathcal{N}(\boldsymbol{y})}(\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}})(\boldsymbol{y}'')\right)\cdot\boldsymbol{y}\\
&=\boldsymbol{y}+\sum_{\boldsymbol{y}'\in\mathcal{N}(\boldsymbol{y})}(\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}})(\boldsymbol{y}')\cdot(\boldsymbol{y}'-\boldsymbol{y})\\
&=\boldsymbol{y}+\sum_{\boldsymbol{y}'\in\mathcal{N}^{-}_{\boldsymbol{y}}(\boldsymbol{\theta})}q(\boldsymbol{y}',\boldsymbol{y})\exp\left(\left[\langle\boldsymbol{\theta},\boldsymbol{y}'-\boldsymbol{y}\rangle+\varphi(\boldsymbol{y}')-\varphi(\boldsymbol{y})\right]/t\right)\cdot(\boldsymbol{y}'-\boldsymbol{y})\;+\sum_{\boldsymbol{y}'\in\mathcal{N}^{+}_{\boldsymbol{y}}(\boldsymbol{\theta})}q(\boldsymbol{y},\boldsymbol{y}')\cdot(\boldsymbol{y}'-\boldsymbol{y}).
\end{aligned}
$$

Let now $f_{\boldsymbol{y}} : \mathbb{R}^d \times \mathcal{N}(\boldsymbol{y}) \to \mathbb{R}$ be defined as:

$$f_{\boldsymbol{y}} : (\boldsymbol{\theta}\,;\boldsymbol{y}') \mapsto \begin{cases} t \cdot q(\boldsymbol{y}', \boldsymbol{y}) \exp\left([\langle \boldsymbol{\theta}, \boldsymbol{y}' - \boldsymbol{y}\rangle + \varphi(\boldsymbol{y}') - \varphi(\boldsymbol{y})]\,/t\right) & \text{if } \alpha_{\boldsymbol{y}}(\boldsymbol{\theta}, \boldsymbol{y}') \leq 1, \\ t \cdot q(\boldsymbol{y}, \boldsymbol{y}') \left([\langle \boldsymbol{\theta}, \boldsymbol{y}' - \boldsymbol{y}\rangle + \varphi(\boldsymbol{y}') - \varphi(\boldsymbol{y})]\,/t + 1 - \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})}\right) & \text{if } \alpha_{\boldsymbol{y}}(\boldsymbol{\theta}, \boldsymbol{y}') > 1. \end{cases}$$

Let $F_{\boldsymbol{y}} : \boldsymbol{\theta} \mapsto \langle \boldsymbol{\theta}, \boldsymbol{y}\rangle + \sum_{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})} f_{\boldsymbol{y}}(\boldsymbol{\theta}\,;\boldsymbol{y}')$. We define the target-dependent regularization function $\Omega_{\boldsymbol{y}}$ and the corresponding Fenchel-Young loss as:

$$\Omega_{\boldsymbol{y}} : \boldsymbol{\mu} \mapsto (F_{\boldsymbol{y}})^*(\boldsymbol{\mu}), \qquad L_{\Omega_{\boldsymbol{y}}}(\boldsymbol{\theta}\,;\boldsymbol{y}) \coloneqq (\Omega_{\boldsymbol{y}})^*(\boldsymbol{\theta}) + \Omega_{\boldsymbol{y}}(\boldsymbol{y}) - \langle \boldsymbol{\theta}, \boldsymbol{y}\rangle.$$

- $\Omega_{\boldsymbol{y}}$ is $t/\mathbb{E}_{q(\boldsymbol{y},\,\cdot\,)}||Y - \boldsymbol{y}||_2^2$-strongly convex:

One can easily check that $f_{\boldsymbol{y}}(\,\cdot\,;\boldsymbol{y}')$ is continuous for all $\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})$, as it is defined piecewise as continuous functions that match on the junction affine hyperplane defined by:

$$\left\{\boldsymbol{\theta} \in \mathbb{R}^d \mid \alpha_{\boldsymbol{y}}(\boldsymbol{\theta}; \boldsymbol{y}') = 1\right\} = \left\{\boldsymbol{\theta} \in \mathbb{R}^d \mid \langle \boldsymbol{\theta}, \boldsymbol{y}' - \boldsymbol{y}\rangle = t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}')\right\}.$$

Moreover, we have that $f_{\boldsymbol{y}}(\,\cdot\,;\boldsymbol{y}')$ is actually differentiable everywhere as its gradient can be continuously extended to the junction affine hyperplane with constant value equal to $q(\boldsymbol{y}, \boldsymbol{y}')(\boldsymbol{y}' - \boldsymbol{y})$. We now show that $f_{\boldsymbol{y}}(\,\cdot\,;\boldsymbol{y}')$ is $\frac{1}{t}q(\boldsymbol{y}, \boldsymbol{y}') \cdot ||\boldsymbol{y}' - \boldsymbol{y}||^2$-smooth. Indeed, it is defined as the composition of the linear form $\boldsymbol{\theta} \mapsto \langle \boldsymbol{\theta}, \boldsymbol{y}' - \boldsymbol{y}\rangle$ and the function $g : \mathbb{R} \to \mathbb{R}$ given by:

$$g : x \mapsto \begin{cases} t \cdot q(\boldsymbol{y}', \boldsymbol{y}) \exp\left([x + \varphi(\boldsymbol{y}') - \varphi(\boldsymbol{y})]\,/t\right) & \text{if } x \leq t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}'), \\ t \cdot q(\boldsymbol{y}, \boldsymbol{y}') \left([x + \varphi(\boldsymbol{y}') - \varphi(\boldsymbol{y})]\,/t + 1 - \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})}\right) & \text{if } x > t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}'). \end{cases}$$

We begin by showing that g is $\frac{1}{t}q(\boldsymbol{y}, \boldsymbol{y}')$-smooth. We have:

$$g' : x \mapsto \begin{cases} q(\boldsymbol{y}', \boldsymbol{y}) \exp\left([x + \varphi(\boldsymbol{y}') - \varphi(\boldsymbol{y})]\,/t\right) & \text{if } x \leq t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}'), \\ q(\boldsymbol{y}, \boldsymbol{y}') & \text{if } x > t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}'). \end{cases}$$

Thus, g' is continuous, and differentiable everywhere except in $x_0 \coloneqq t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}')$. Its derivative is given by:

$$g'' : x \mapsto \begin{cases} \frac{1}{t}q(\boldsymbol{y}', \boldsymbol{y}) \exp\left([x + \varphi(\boldsymbol{y}') - \varphi(\boldsymbol{y})]\,/t\right) & \text{if } x \leq t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}'), \\ 0 & \text{if } x > t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}'). \end{cases}$$

- For $x_1, x_2 \leq x_0$, we have:

$$\begin{aligned} |g'(x_1) - g'(x_2)| &\leq |x_1 - x_2| \sup_{x \in]-\infty, x_0[} |g''(x)| \\ &= |x_1 - x_2| \lim_{\substack{x \to x_0 \\ x < x_0}} |g''(x)| \\ &= \frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}') \cdot |x_1 - x_2|. \end{aligned}$$

- For $x_1, x_2 \geq x_0$, we trivially have $|g'(x_1) - g'(x_2)| = 0$.

- For $x_1 \leq x_0 \leq x_2$, we have:

$$\begin{aligned} |g'(x_1) - g'(x_2)| &= |(g'(x_1) - g'(x_0)) - (g'(x_2) - g'(x_0))| \\ &\leq |g'(x_1) - g'(x_0)| + |g'(x_2) - g'(x_0)| \\ &\leq \frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}') \cdot |x_1 - x_0| \\ &\leq \frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}') \cdot |x_1 - x_2|. \end{aligned}$$

Thus, we have:

$$\forall x_1, x_2 \in \mathbb{R}, |g'(x_1) - g'(x_2)| \leq \frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}') \cdot |x_1 - x_2|,$$

and g is $\frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}')$-smooth. Nevertheless, we have $f_{\boldsymbol{y}}(\,\cdot\,, \boldsymbol{y}') = g(\langle\,\cdot\,, \boldsymbol{y}' - \boldsymbol{y}\rangle)$. Thus, we have, for $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$:

$$\begin{aligned}
||\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{y}}(\boldsymbol{\theta_1}, \boldsymbol{y}') - \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{y}}(\boldsymbol{\theta_2}, \boldsymbol{y}')|| &= ||g'(\langle \boldsymbol{\theta}_1, \boldsymbol{y}' - \boldsymbol{y}\rangle)(\boldsymbol{y}' - \boldsymbol{y}) - g'(\langle \boldsymbol{\theta}_2, \boldsymbol{y}' - \boldsymbol{y}\rangle)(\boldsymbol{y}' - \boldsymbol{y})|| \\
&= |g'(\langle \boldsymbol{\theta}_1, \boldsymbol{y}' - \boldsymbol{y}\rangle) - g'(\langle \boldsymbol{\theta}_2, \boldsymbol{y}' - \boldsymbol{y}\rangle)| \cdot ||\boldsymbol{y}' - \boldsymbol{y}|| \\
&\leq \frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}') \cdot |\langle \boldsymbol{\theta}_1, \boldsymbol{y}' - \boldsymbol{y}\rangle - \langle \boldsymbol{\theta}_2, \boldsymbol{y}' - \boldsymbol{y}\rangle| \cdot ||\boldsymbol{y}' - \boldsymbol{y}|| \\
&\leq \frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}') \cdot ||\boldsymbol{y}' - \boldsymbol{y}||^2 \cdot ||\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2||,
\end{aligned}$$

and $f_{\boldsymbol{y}}(\,\cdot\,, \boldsymbol{y}')$ is $\frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}') \cdot ||\boldsymbol{y}' - \boldsymbol{y}||^2$-smooth. Thus, recalling that $F_{\boldsymbol{y}}$ is defined as

$$F_{\boldsymbol{y}} : \boldsymbol{\theta} \mapsto \langle \boldsymbol{\theta}, \boldsymbol{y}\rangle + \sum_{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})} f_{\boldsymbol{y}}(\boldsymbol{\theta}; \boldsymbol{y}'),$$

we have that $F_{\boldsymbol{y}}$ is $\sum_{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})} \frac{1}{t} q(\boldsymbol{y}, \boldsymbol{y}') \cdot ||\boldsymbol{y}' - \boldsymbol{y}||^2 = \mathbb{E}_{q(\boldsymbol{y}, \,\cdot\,)} ||Y - \boldsymbol{y}||_2^2 / t$-smooth. Finally, as $\Omega_{\boldsymbol{y}} \coloneqq (F_{\boldsymbol{y}})^*$, Fenchel duality theory gives that $\Omega_{\boldsymbol{y}}$ is $t / \mathbb{E}_{q(\boldsymbol{y}, \,\cdot\,)} ||Y - \boldsymbol{y}||_2^2$-strongly convex.

- $\underline{\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta}, \boldsymbol{y}}}[Y] = \mathsf{argmax}_{\boldsymbol{\mu} \in \mathsf{conv}(\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\})} \{\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle - \Omega_{\boldsymbol{y}}(\boldsymbol{\mu})\}}$:

Noticing that g is continuous on $\mathbb{R}$, convex on $\left]-\infty, t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}')\right[$ and on $\left]t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}'), +\infty\right[$, and with matching derivatives on the junction:

$$g'(t) \xrightarrow[t < t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}')]{t \to t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}')} q(\boldsymbol{y}, \boldsymbol{y}'), \quad g'(t) \xrightarrow[t > t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}')]{t \to t \log \frac{q(\boldsymbol{y}, \boldsymbol{y}')}{q(\boldsymbol{y}', \boldsymbol{y})} + \varphi(\boldsymbol{y}) - \varphi(\boldsymbol{y}')} q(\boldsymbol{y}, \boldsymbol{y}'),$$

gives that g is convex on $\mathbb{R}$. Thus, $f_{\boldsymbol{y}}(\,\cdot\,; \boldsymbol{y}')$ is convex on $\mathbb{R}^d$ by composition. Thus,

$$F_{\boldsymbol{y}} : \boldsymbol{\theta} \mapsto \langle \boldsymbol{\theta}, \boldsymbol{y}\rangle + \sum_{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})} f_{\boldsymbol{y}}(\boldsymbol{\theta}; \boldsymbol{y}')$$

is closed proper convex as the sum of such functions. The Fenchel-Moreau theorem then gives that it is equal to its biconjugate. Thus, we have:

$$F_{\boldsymbol{y}}(\boldsymbol{\theta}) = \sup_{\boldsymbol{\mu} \in \mathbb{R}^d} \{\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle - (F_{\boldsymbol{y}})^*(\boldsymbol{\mu})\} = \sup_{\boldsymbol{\mu} \in \mathbb{R}^d} \{\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle - \Omega_{\boldsymbol{y}}(\boldsymbol{\mu})\}.$$

Nonetheless, the gradient of $F_{\boldsymbol{y}}$ is given by:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} F_{\boldsymbol{y}}(\boldsymbol{\theta}) &= \boldsymbol{y} + \sum_{\boldsymbol{y}' \in \mathcal{N}^-_{\boldsymbol{y}}(\boldsymbol{\theta})} q(\boldsymbol{y}', \boldsymbol{y}) \exp\left([\langle \boldsymbol{\theta}, \boldsymbol{y}' - \boldsymbol{y}\rangle + \varphi(\boldsymbol{y}') - \varphi(\boldsymbol{y})] / t\right) \cdot (\boldsymbol{y}' - \boldsymbol{y}) \; + \sum_{\boldsymbol{y}' \in \mathcal{N}^+_{\boldsymbol{y}}(\boldsymbol{\theta})} q(\boldsymbol{y}, \boldsymbol{y}') \cdot (\boldsymbol{y}' - \boldsymbol{y}) \\
&= \mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta}, \boldsymbol{y}}}[Y].
\end{aligned}$$

Thus, we have $\nabla F_{\boldsymbol{y}}(\mathbb{R}^d) \subset \mathsf{conv}\left(\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}\right)$, which gives:

$$\forall \boldsymbol{\theta} \in \mathbb{R}^d, \; ||\nabla F_{\boldsymbol{y}}(\boldsymbol{\theta})|| \leq R_{\mathcal{N}(\boldsymbol{y})} \coloneqq \max_{\boldsymbol{\mu} \in \mathsf{conv}(\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\})} ||\boldsymbol{\mu}||,$$

so that we have $\mathsf{dom}(\Omega_{\boldsymbol{y}}) \subset B(\mathbf{0}, R_{\mathcal{N}(\boldsymbol{y})})$. Thus we can actually write:

$$F_{\boldsymbol{y}}(\boldsymbol{\theta}) = \sup_{\boldsymbol{\mu} \in B(\mathbf{0}, R_{\mathcal{N}(\boldsymbol{y})})} \{\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle - \Omega_{\boldsymbol{y}}(\boldsymbol{\mu})\},$$

and now apply Danksin's theorem as $B(\mathbf{0}, R_{\mathcal{N}(\boldsymbol{y})})$ is compact, which further gives:

$$\partial F_{\boldsymbol{y}}(\boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\mu} \in B(\mathbf{0}, R_{\mathcal{N}(\boldsymbol{y})})} \{\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle - \Omega_{\boldsymbol{y}}(\boldsymbol{\mu})\},$$

and the fact that $F_{\boldsymbol{y}}$ is differentiable gives that both sides are single-valued. Moreover, as $\nabla F_{\boldsymbol{y}}(\mathbb{R}^d) \subset \mathsf{conv}\left(\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}\right)$, we know that the right hand side is maximized in $\mathsf{conv}\left(\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}\right)$, and we can actually write:

$$\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y] = \nabla F_{\boldsymbol{y}}(\boldsymbol{\theta}) = \underset{\boldsymbol{\mu} \in \mathsf{conv}(\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\})}{\mathsf{argmax}} \left\{ \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - \Omega_{\boldsymbol{y}}(\boldsymbol{\mu}) \right\}.$$

- Smoothness of $L_{\Omega_{\boldsymbol{y}}}(\,\cdot\,;\boldsymbol{y})$ and expression of its gradient:

Based on the above, we have:

$$L_{\Omega_{\boldsymbol{y}}}(\boldsymbol{\theta}\,;\boldsymbol{y}) = F_{\boldsymbol{y}}(\boldsymbol{\theta}) + \Omega_{\boldsymbol{y}}(\boldsymbol{y}) - \langle \boldsymbol{\theta}, \boldsymbol{y} \rangle.$$

Thus, the $\mathbb{E}_{q(\boldsymbol{y},\,\cdot\,)}||Y - \boldsymbol{y}||_2^2/t$-smoothness of $L_{\Omega_{\boldsymbol{y}}}(\,\cdot\,;\boldsymbol{y})$ follows directly from the previously established $\mathbb{E}_{q(\boldsymbol{y},\,\cdot\,)}||Y - \boldsymbol{y}||_2^2/t$-smoothness of $F_{\boldsymbol{y}}$. Similarly, the expression of $\nabla_{\boldsymbol{\theta}} L_{\Omega_{\boldsymbol{y}}}(\boldsymbol{\theta}\,;\boldsymbol{y})$ follows from the previously established expression of $\nabla_{\boldsymbol{\theta}} F_{\boldsymbol{y}}(\boldsymbol{\theta})$, and we have:

$$\nabla_{\boldsymbol{\theta}} L_{\Omega_{\boldsymbol{y}}}(\boldsymbol{\theta}\,;\boldsymbol{y}) = \nabla_{\boldsymbol{\theta}} F_{\boldsymbol{y}}(\boldsymbol{\theta}) - \boldsymbol{y} = \mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y] - \boldsymbol{y}.$$

□

Proposition C.2. In the unsupervised setting, given a dataset $(\boldsymbol{y}_i)_{i=1}^N$, the distribution of the first iterate of the Markov chain with transition kernel defined in Eq. (3) and initialized by $\boldsymbol{y}^{(0)} = \boldsymbol{y}_i$, with $i \sim \mathcal{U}([\![1, N]\!])$, is given by:

$$\begin{aligned}
(\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\bar{Y}_N})(\boldsymbol{y}) &= \sum_{\boldsymbol{y}' \in \mathcal{Y}} \left(\sum_{i=1}^N \mathbf{1}_{\{\boldsymbol{y}_i = \boldsymbol{y}'\}} \cdot \frac{1}{N} \right) P_{\boldsymbol{\theta},t}(\boldsymbol{y}', \boldsymbol{y}) \\
&= \sum_{\boldsymbol{y}' \in \mathcal{Y}} \left(\sum_{i=1}^N \mathbf{1}_{\{\boldsymbol{y}_i = \boldsymbol{y}'\}} \cdot \frac{1}{N} \right) \boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}'}(\boldsymbol{y}) \\
&= \frac{1}{N} \sum_{i=1}^N \boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}_i}(\boldsymbol{y}).
\end{aligned}$$

Thus, keeping the same notations as in the previous proof, previous calculations give:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\bar{Y}_N}}[Y] &= \frac{1}{N} \sum_{i=1}^N \mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}_i}}[Y] \\
&= \frac{1}{N} \sum_{i=1}^N \nabla_{\boldsymbol{\theta}} F_{\boldsymbol{y}_i}(\boldsymbol{\theta}) \\
&= \nabla_{\boldsymbol{\theta}} \left(\frac{1}{N} \sum_{i=1}^N F_{\boldsymbol{y}_i} \right)(\boldsymbol{\theta}).
\end{aligned}$$

Let $F_{\bar{Y}_N} \coloneqq \frac{1}{N} \sum_{i=1}^N F_{\boldsymbol{y}_i}$ Then, the exact same arguments as in the supervised case hold, and the results of Proposition C.2 are obtained by replacing $F_{\boldsymbol{y}}$ by $F_{\bar{Y}_N}$ in the proof of Proposition C.1, and noticing that the previously shown $\mathbb{E}_{q(\boldsymbol{y}_i,\,\cdot\,)}||Y - \boldsymbol{y}_i||_2^2/t$-smoothness of $F_{\boldsymbol{y}_i}$ gives that $F_{\bar{Y}_N}$ is $\frac{1}{N} \sum_{i=1}^N \mathbb{E}_{q(\boldsymbol{y}_i,\,\cdot\,)}||Y - \boldsymbol{y}_i||_2^2/t$-smooth. Similar arguments also hold for the regularized optimization formulation, by noting that this time we have $\nabla F_{\bar{Y}_N}(\mathbb{R}^d) \subset \mathsf{conv}\left(\bigcup_{i=1}^N \{\mathcal{N}(\boldsymbol{y}_i) \cup \{\boldsymbol{y}_i\}\} \right)$. □

E.8 Proof of Proposition C.3

Proof. The first point is directly given by the fact that $\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y]$ is the expectation of a distribution over $\mathcal{N}(\boldsymbol{y}) \cup \{\boldsymbol{y}\}$. For the second and third points, as derived in Appendix E.7, we have:

$$\mathbb{E}_{\boldsymbol{p}^{(1)}_{\boldsymbol{\theta},\boldsymbol{y}}}[Y] = \boldsymbol{y} + \sum_{\boldsymbol{y}' \in \mathcal{N}^-_{\boldsymbol{y}}(\boldsymbol{\theta})} q(\boldsymbol{y}', \boldsymbol{y}) \exp\left(\left[\langle \boldsymbol{\theta}, \boldsymbol{y}' - \boldsymbol{y} \rangle + \varphi(\boldsymbol{y}') - \varphi(\boldsymbol{y}) \right] / t \right) \cdot (\boldsymbol{y}' - \boldsymbol{y}) \; + \sum_{\boldsymbol{y}' \in \mathcal{N}^+_{\boldsymbol{y}}(\boldsymbol{\theta})} q(\boldsymbol{y}, \boldsymbol{y}') \cdot (\boldsymbol{y}' - \boldsymbol{y}).$$

Define then:

$$\mathcal{N}_{\text{better}}(\boldsymbol{y}) \coloneqq \{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}) \mid \langle \boldsymbol{\theta}, \boldsymbol{y}' \rangle + \varphi(\boldsymbol{y}') > \langle \boldsymbol{\theta}, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y})\},$$
$$\mathcal{N}_{\text{worse}}(\boldsymbol{y}) \coloneqq \{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}) \mid \langle \boldsymbol{\theta}, \boldsymbol{y}' \rangle + \varphi(\boldsymbol{y}') < \langle \boldsymbol{\theta}, \boldsymbol{y} \rangle + \varphi(\boldsymbol{y})\}$$

as the sets of improving and worsening neighbors of $\boldsymbol{y}$ respectively (assuming no neighbor of $\boldsymbol{y}$ has exactly equal objective value for simplicity, which is true almost everywhere w.r.t. $\boldsymbol{\theta} \in \mathbb{R}^d$).

Low temperature limit. We have:

$$\mathcal{N}_{\boldsymbol{y}}^{+}(\boldsymbol{\theta}) \xrightarrow[t \to 0^+]{} \mathcal{N}_{\text{better}}(\boldsymbol{y}), \quad \text{and} \quad \mathcal{N}_{\boldsymbol{y}}^{-}(\boldsymbol{\theta}) \xrightarrow[t \to 0^+]{} \mathcal{N}_{\text{worse}}(\boldsymbol{y}).$$

Then, as $x < 0 \implies \exp(x/t) \xrightarrow[t \to 0^+]{} 0$, we have effectively

$$\mathbb{E}_{\boldsymbol{p}_{\boldsymbol{\theta},\boldsymbol{y}}^{(1)}}[Y] \xrightarrow[t \to 0^+]{} \boldsymbol{y} + \sum_{\boldsymbol{y}' \in \mathcal{N}_{\text{better}}(\boldsymbol{y})} q(\boldsymbol{y}, \boldsymbol{y}') \cdot (\boldsymbol{y}' - \boldsymbol{y}).$$

High temperature limit. As $\forall x \in \mathbb{R},\ \exp(x/t) \xrightarrow[t \to +\infty]{} 1$, we have:

$$\mathcal{N}_{\boldsymbol{y}}^{+}(\boldsymbol{\theta}) \xrightarrow[t \to +\infty]{} \{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}) \mid q(\boldsymbol{y}', \boldsymbol{y}) > q(\boldsymbol{y}, \boldsymbol{y}')\}, \quad \text{and} \quad \mathcal{N}_{\boldsymbol{y}}^{-}(\boldsymbol{\theta}) \xrightarrow[t \to +\infty]{} \{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y}) \mid q(\boldsymbol{y}', \boldsymbol{y}) \leq (\boldsymbol{y}, \boldsymbol{y}')\}.$$

Thus, we have:

$$\mathbb{E}_{\boldsymbol{p}_{\boldsymbol{\theta},\boldsymbol{y}}^{(1)}}[Y] \xrightarrow[t \to +\infty]{} \boldsymbol{y} + \sum_{\boldsymbol{y}' \mid q(\boldsymbol{y}', \boldsymbol{y}) \leq (\boldsymbol{y}, \boldsymbol{y}')} q(\boldsymbol{y}', \boldsymbol{y}) \cdot (\boldsymbol{y}' - \boldsymbol{y}) + \sum_{\boldsymbol{y}' \mid q(\boldsymbol{y}', \boldsymbol{y}) > (\boldsymbol{y}, \boldsymbol{y}')} q(\boldsymbol{y}, \boldsymbol{y}') \cdot (\boldsymbol{y}' - \boldsymbol{y}),$$

which gives effectively:

$$\mathbb{E}_{\boldsymbol{p}_{\boldsymbol{\theta},\boldsymbol{y}}^{(1)}}[Y] \xrightarrow[t \to +\infty]{} \boldsymbol{y} + \sum_{\boldsymbol{y}' \in \mathcal{N}(\boldsymbol{y})} \min\left[q(\boldsymbol{y}, \boldsymbol{y}'), q(\boldsymbol{y}', \boldsymbol{y})\right] \cdot (\boldsymbol{y}' - \boldsymbol{y}).$$

□

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the abstract and introduction, we clearly outline the specific settings addressed in this paper and the corresponding contributions.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include a discussion on the limitations of this work in the conclusion.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All proofs are included in the appendix, and provide the full set of needed assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The main contributions of the paper are high-level training algorithms, which are described in detail. Additionally, the numerical experiments are carefully documented to ensure they are as reproducible as possible.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The data of the *EURO Meets NeurIPS 2022 Vehicle Routing Competition* is accessible online. Apart from this, we only use synthetic data, for which the generation process is detailed. We will release the code upon acceptance of the paper.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We present the experimental setting with sufficient level of detail to fully appreciate the results in the core of the paper, and give full details in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We average experiments over multiple runs with varying seeds and report error bars and statistical significance statements in consequence.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We include sufficient relevant information on the compute resources.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have carefully reviewed the NeurIPS Code of Ethics and ensured that all aspects of the research comply with its guidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: As this is a foundational research paper, we do not foresee direct negative societal impacts in its current form. Potential negative impacts would depend on downstream applications, which are beyond the scope of this work.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: this work does not pose such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly credit original papers for the code we use (Vidal [49] and Baty et al. [4]), and proper credit will be also given in the released code.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No assets are released.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.