OPT2024: 16th Annual Workshop on Optimization for Machine Learning

Optimal Transport for Probabilistic Circuits

Adrian Ciotinga ACIOTING@ASU.EDU
YooJung Choi YJ.CHOI@ASU.EDU
School of Computing and Augmented Intelligence, Arizona State University

Abstract

We introduce a novel optimal transport framework for probabilistic circuits (PCs). While it has been shown recently that divergences between distributions represented as certain classes of PCs can be computed tractably, to the best of our knowledge, there is no existing approach to compute the Wasserstein distance between probability distributions given by PCs. We consider a Wasserstein-type distance that restricts the coupling measure of the associated optimal transport problem to be a probabilistic circuit. We then develop an algorithm for computing this distance by solving a series of small linear programs and derive the circuit conditions under which this is tractable. Furthermore, we show that we can also retrieve the optimal transport plan between the PCs from the solutions to these linear programming problems. We then consider the empirical Wasserstein distance between a PC and a dataset, and show that we can estimate the PC parameters to minimize this distance through an efficient iterative algorithm.

Code: https://github.com/aciotinga/pc-optimal-transport

1. Introduction

Modeling probability distributions in a way that enables tractable computation of certain probabilistic queries is of great interest to the machine learning community. Probabilistic circuits (PCs) [3] provide a unifying framework for representing many classes of tractable probabilistic models as computational graphs. They have received attention lately for the ability to guarantee tractable inference of certain query classes through imposing structural properties on the computational graph of the circuit. This includes tractable marginal and conditional inference, as well as pairwise queries that compare two circuits such as Kullback-Leibler Divergence and cross-entropy [9, 16].

However, to the best of our knowledge, there is no existing algorithm to compute the Wasserstein distance between two probabilistic circuits.

Definition 1 (Wasserstein distance) *Let P and Q be two probability measures on $\mathbb{R}^n$. For $p \geq 1$, the* p-Wasserstein distance *between P and Q is $\mathsf{W}_p^p(P,Q) \triangleq \inf_{\gamma \in \Gamma(P,Q)} \mathbb{E}_{\gamma(\mathbf{x},\mathbf{y})}[\|\mathbf{x}-\mathbf{y}\|_p^p]$ where $\Gamma(P,Q)$ denotes the set of all* couplings *which are joint distributions whose marginal distributions coincide exactly with P and Q. That is, for all $\gamma \in \Gamma(P,Q)$, $P(\mathbf{x}) = \int_{\mathbb{R}^n} \gamma(\mathbf{x},\mathbf{y}) d\mathbf{y}$ and $Q(\mathbf{y}) = \int_{\mathbb{R}^n} \gamma(\mathbf{x},\mathbf{y}) d\mathbf{x}$.*

Here, the *Wasserstein objective* of some (not necessarily optimal) coupling refers to the expectation inside the infimum taken over that coupling, and the *Wasserstein distance* between two distributions refers to the value taken by the Wasserstein objective for the optimal coupling.

This paper focuses on computing (or bounding) the Wasserstein distance and optimal transport plan between (i) two probabilistic circuits and (ii) a probabilistic circuit and an empirical distribution. For (i) we propose a Wasserstein-type distance that upper-bounds the true Wasserstein distance

and provide an efficient and exact algorithm for computing it between two circuits. For (ii) we propose a parameter estimation algorithm for PCs that seeks to minimize the Wasserstein distance between a circuit and an empirical distribution and provide experimental results comparing it to existing approaches.

2. Optimal Transport between Circuits

We now consider the problem of computing Wasserstein distances and optimal transport plans between distributions represented by probabilistic circuits P and Q with scopes $\mathbf{X}$ and $\mathbf{Y}$.

Definition 2 (Probabilistic circuit) *A probabilistic circuit (PC) C over a set of discrete or continuous random variables $\mathbf{X}$ is a parameterized, rooted directed acyclic graph (DAG) with three types of nodes: sum, product and input nodes. Each sum node n has normalized parameters $\theta_{n,c}$ for each child node c, and each input node n is associated with function f_n that encodes a univariate probability distribution over one of the random variables $X_i \in \mathbf{X}$, also called its* scope $\mathsf{sc}(n)$. *The set of child nodes for an internal node (sum or product) n is denoted $\mathsf{ch}(n)$, and the sub-circuit rooted at any node n parameterizes a probability distribution $p_n(x)$ over its scope $\mathsf{sc}(n) = \bigcup_{c \in \mathsf{ch}(n)} \mathsf{sc}(c)$ defined as follows:*[1]

$$p_n(\mathbf{x}) = \begin{cases} f_n(\mathbf{x}) & \text{if } n \text{ is an input node,} \\ \prod_{c \in \mathsf{ch}(n)} p_c(\mathbf{x}) & \text{if } n \text{ is a product node,} \\ \sum_{c \in \mathsf{ch}(n)} \theta_{n,c} p_c(\mathbf{x}) & \text{if } n \text{ is a sum node.} \end{cases}$$

Structural properties of a PC's computational graph enable tractable computation of certain queries. In particular, as is common in the PC literature, we assume that the circuit structure satisfies two properties, namely *smoothness* and *decomposability*. A PC C is *smooth* if the children of every sum node $n \in C$ have the same scope: $\forall n_i \in \mathsf{ch}(n), \mathsf{sc}(n_i) = \mathsf{sc}(n)$. C is *decomposable* if the children of every product node $n \in C$ have disjoint scopes: $\forall n_i, n_j \in \mathsf{ch}(n), \mathsf{sc}(n_i) \bigcap \mathsf{sc}(n_j) = \emptyset$. Such circuits admit linear-time computation of marginal and conditional probabilities for arbitrary subsets of variables [3].

Furthermore, we assume that we can compute the Wasserstein distance between circuit input distributions in constant time—which is the case for the 2-Wasserstein distance between Gaussian distributions and categorical distributions associated with a metric space—and that there is a bijective mapping $\leftrightarrow$ between random variables in $\mathbf{X}$ and random variables in $\mathbf{Y}$. Unfortunately, even with the above assumptions, computing the Wasserstein distance between probabilistic circuits is computationally hard, including for circuits satisfying restrictive structural properties that enable tractable computation of hard queries such maximum-a-posteriori (MAP) [3]. Complete proofs of all theorems and propositions can be found in the Appendix.

Theorem 1 *Suppose P and Q are probabilistic circuits over n Boolean variables. Then computing the ∞-Wasserstein distance between P and Q is coNP-hard.*

Theorem 1 shows that computing the ∞-Wasserstein distance between two PCs is computationally hard. Whether computing W_p for some other fixed p (such as $p = 1$ or 2) is NP-hard is still

1. Below, we implicitly project $\mathbf{x}$ onto $\mathsf{sc}(n)$ by only considering the dimensions that correspond to random variables in the node's scope.

an open question—although there only exist efficient algorithms that bound this quantity between GMMs, rather than compute it exactly [2, 5]. In this work, however, we are interested in efficiently computing or upper-bounding W_p between PCs for *arbitrary* p, including W_∞. Thus, to address this computational challenge, we consider a Wasserstein-type distance between PCs by restricting the set of coupling measures to be PCs of a particular structure. Furthermore, we derive the structural conditions on the input PCs required to construct such structure and find the parameters that minimize the Wasserstein objective in time quadratic in the size of the input circuits.

2.1. CW_p: A Distance based on Coupling Circuits

We propose the notion of a *coupling circuit* between two compatible (see Definition 3 below) PCs, and introduce a Wasserstein-type distance CW_p which restricts the coupling set in Definition 1 to be circuits of this form. We then exploit the structural properties guaranteed by coupling circuits, namely smoothness and decomposability, to derive efficient algorithms for computing CW_p.

Definition 3 (Circuit compatibility [16]) *Two smooth and decomposable PCs P and Q over RVs $\mathbf{X}$ and $\mathbf{Y}$, respectively, are* compatible *if the following two conditions hold: (i) there is a bijective mapping $\leftrightarrow$ between RVs X_i and Y_i, and (ii) any pair of product nodes $n \in P$ and $m \in Q$ with the same scope up to the bijective mapping are mutually compatible and decompose the scope the same way—that is, if n and m have scopes $\mathbf{X}$ and $\mathbf{Y}$ and $\mathbf{X} \leftrightarrow \mathbf{Y}$, then n and m have the same number of children, and for each child of n with scope $\mathbf{X}_i$ there is a corresponding child of m with scope $\mathbf{Y}_i$ such that $\mathbf{X}_i \leftrightarrow \mathbf{Y}_i$. Such pair of nodes are called* corresponding *nodes.*

Definition 4 (Coupling circuit) *A* coupling circuit *C between two compatible PCs P and Q with scopes $\mathbf{X}$ and $\mathbf{Y}$, respectively, is a PC with the following properties. (i) Each node $r \in C$ is recursively a coupling of a pair of nodes $n \in P$ and $m \in Q$.[2] (ii) Each node $r \in C$ that is a coupling of sum nodes $n \in P, m \in Q$ with edge weights $\{\theta_i\}$ and $\{\theta_j\}$ has edge weights $\{\theta_{i,j}\}$ such that $\sum_i \theta_{i,j} = \theta_j$ and $\sum_j \theta_{i,j} = \theta_i$ for all i and j.*

The second property described above ensures that such coupling circuit C matches marginal distributions to P and Q as described in Def. 1 (see Appx. B.5). Thus, valid parameterizations of the coupling circuit structure form a subset of couplings in Def. 1.

Definition 5 (Circuit Wasserstein distance CW_p) *The p-th Circuit Wasserstein distance CW_p between PCs P and Q is the value of the p-th Wasserstein objective computed for an objective-minimizing coupling measure that is restricted to be a coupling circuit of P and Q.*

Proposition 2 *For any set $\mathcal{C}$ of compatible circuits, CW_p defines a metric on $\mathcal{C}$.*

By definition, we have that $\mathsf{W}_p(P, Q) \leq \mathsf{CW}_p(P, Q)$ because both are infima of the same Wasserstein objective, while the feasible set of couplings for $\mathsf{CW}_p(P, Q)$ is more restrictive. Thus, the circuit Wasserstein distance is a metric that upper-bounds the true Wasserstein distance between PCs.

2. The coupling circuit has the same structure as the product circuit [16] of P and Q. Informally, this is done by constructing a *cross product* of children at every pair of sum nodes, and the product of corresponding children at every pair of product nodes. Algorithm A.1 shows this construction.

2.2. Exact and Efficient Computation of CW_p

In this section, we first identify the recursive properties of the Wasserstein objective for a given parameterization of the coupling circuit that enable its linear-time computation in the size of the coupling circuit. Then, we propose a simple algorithm to compute the exact parameters for the coupling circuit that minimize the Wasserstein objective, giving us, again, a linear-time algorithm to compute CW_p as well as a transport plan between PCs.

Recursive Computation of the Wasserstein Objective Below equation shows the recursive computation of the CW_p-objective function $g(n)$ at each node n in the coupling circuit C (see Appx. B.2 for correctness proof). We denote the ith child of node n to be c_i.

$$g(n) = \begin{cases} \sum_i \theta_i g(c_i) & \text{if } n \text{ is a sum} \\ \sum_i g(c_i) & \text{if } n \text{ is a product with sum or product node children} \\ \mathsf{W}_p(c_1, c_2) & \text{if } n \text{ is a product with input node children} \end{cases} \tag{1}$$

Thus, we can push computation of the Wasserstein objective down to the leaf nodes of a coupling circuit, and our algorithm only requires a closed-form solution for W_p between univariate input distributions. Note that the objective function at a product node is the *sum* of the objective functions at its children; this is because the L_p^p-norm decomposes into the sum of norm in each dimension.

Recursive Computation of the Optimal Coupling Circuit Parameters for CW_p Leveraging the recursive properties of the Wasserstein objective, we can compute the optimal sum edge parameters in the coupling circuit by solving a small linear program at each sum node. This is done by using the optimal CW_p values computed at each child as the coefficients for the sum of corresponding weight parameters in the linear objective, which comes from the decomposition of the Wasserstein objective at sum nodes in the previous section. These linear programs are constrained to enforce the marginal-matching constraints defined in Def. 4. Since the time to solve the linear program at each sum node depends only on the number of children of the sum node, which is bounded, we consider this time constant when calculating the runtime of the full algorithm. Thus, we can compute CW_p and the corresponding transport plan between two circuits in time linear in the number of nodes in the coupling circuit, or equivalently quadratic in the number of nodes in the original circuits. Appendix B.4 presents the recursive algorithm in detail along with correctness proof.

2.3. Experimental Results

To determine the feasibility of computing CW_p for large circuits, we implement and evaluate our algorithm on randomly-generated compatible circuits of varying sizes. As a baseline, we consider the naive application of an existing algorithm to compute a similar Wasserstein-type distance called "Mixture Wasserstein" MW_p between Gaussian mixture models (GMMs) [5], leveraging

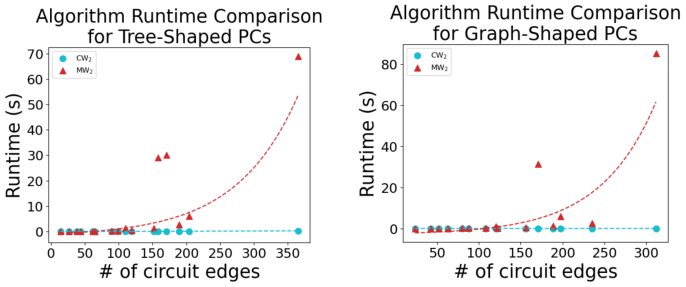

Figure 1: Runtime of Wasserstein-type distance computation using our approach (blue dots) and the baseline (red triangles). For circuits larger than those depicted, the baseline approach runs out of memory. See Appendix C.1 for more detailed experiments.

the fact that PCs with Gaussian input units can be "unrolled" into GMMs. However, we quickly observe the impracticality of this baseline approach for circuits larger than even a few hundred edges due to the GMM representation of a PC potentially being *exponentially larger* than the original circuit. Figure 1 illustrates how the direct application of GMM-based algorithms is intractable, while our approach runs in quadratic time in the size of the original circuits just as predicted by the theory.

3. Parameter Learning of PCs using the Empirical Wasserstein Distance

Motivated by past works that minimize the Wasserstein distance between a generative model and the empirical distribution, parameterized by a dataset, to train model parameters [1, 13–15], we investigate the applicability of minimizing the Wasserstein distance between a PC and data as a means of learning the parameters of a given PC structure. Formally, suppose we have a dataset $\mathcal{D} = \{\mathbf{y}^{(k)}\}_{k=1}^n$ that induces the empirical probability measure $\hat{Q}$. Then for a given PC structure, we find its parameters θ to optimize the following:

$$\begin{aligned}\min_\theta \mathsf{W}_p^p(P_\theta, \hat{Q}) &= \min_\theta \inf_{\gamma \in \Gamma(P_\theta, \hat{Q})} \mathbb{E}_{\gamma(\mathbf{x},\mathbf{y})}[\|\mathbf{x}-\mathbf{y}\|_p^p] \\ &= \min_\theta \inf_{\gamma \in \Gamma(P_\theta, \hat{Q})} \frac{1}{n}\sum_{k=1}^n \mathbb{E}_{\gamma(\mathbf{x}|\mathbf{y}^{(k)})}\left[\left\|\mathbf{x}-\mathbf{y}^{(k)}\right\|_p^p\right] \end{aligned} \tag{2}$$

Unfortunately, the above optimization problem is NP-hard (see Appendix B.6). We tackle this computational hardness by again imposing a circuit structure on the coupling measure, allowing us compute the Wasserstein objective and optimize it efficiently.

Definition 6 (Empirical Circuit Wasserstein distance) *Let P be a PC distribution and $\hat{Q}$ an empirical distribution induced by a dataset $\mathcal{D} = \{\mathbf{y}^{(k)}\}_{k=1}^n$. The p-Empirical Circuit Wasserstein distance between P and $\hat{Q}$ is*

$$\mathsf{ECW}_p^p(P, \hat{Q}) = \min_\gamma \frac{1}{n}\sum_{k=1}^n \mathbb{E}_{\gamma(\mathbf{x}|\mathbf{y}^{(k)})}\left[\left\|\mathbf{x}-\mathbf{y}^{(k)}\right\|_p^p\right],$$

where $\gamma(\mathbf{x}, \mathbf{y} = \mathbf{y}^{(k)})$ satisfies the following: (i) for each $k \in \{1, \ldots, n\}$, $\gamma(\cdot, \mathbf{y} = \mathbf{y}^{(k)})$ is a PC with the same structure as P (but not necessarily the same parameters) that normalizes to $1/n$, and (ii) $\sum_{k=1}^n \gamma(\mathbf{x}, \mathbf{y} = \mathbf{y}^{(k)}) = P(\mathbf{x})$.

Since a coupling satisfying the above structure also satisfies the marginal constraints and is in $\Gamma(P, \hat{Q})$, we learn the parameters of PCs by minimizing this upper bound $\mathsf{W}_p(P, \hat{Q}) \leq \mathsf{ECW}_p(P, \hat{Q})$, alternating between (i) optimizing the coupling given the current circuit parameters and (ii) updating the circuit parameters given the current coupling.

Step (i) computes $\mathsf{ECW}_p^p(P_\theta, \hat{Q})$ for a given θ and in the process finds the corresponding coupling γ. Rather than materializing k PCs to represent each $\gamma(\cdot, \mathbf{y} = \mathbf{y}^{(k)})$, we equivalently model a single coupling circuit γ as having the same structure as P and a set of parameters $\{w_{r,c,k}\}_{k=1}^n$ for each parameter $\theta_{r,c}$ in P. Then optimizing the coupling circuit parameters amounts to minimizing the Wasserstein objective according to the coupling distribution—similar to computing CW—and can be done efficiently by solving a small linear program at each sum node. Here, we have the following marginal-matching constraints: $\sum_{k=1}^n w_{r,c,k} = \theta_{r,c}$ for each sum node r and child c and

$\sum_c w_{r,c,k} = 1/n$ for each k. Step (ii) simply updates the parameters θ of PC P from a given coupling γ as $\theta_{r,c} = \sum_{k=1}^{n} w_{r,c,k}$ since γ has the same structure as P.

Interestingly, the above linear program at each sum node is a variation of the continuous knapsack problem [12] and has a closed-form solution where each weight $w_{r,c,k}$ is either $\frac{1}{n}$ or zero (details in Appendix B.7); intuitively, the coupling circuit parameters w describe how each data point is routed through the circuit. Due to the closed-form solution of the LP, the time complexity of one iteration of our algorithm is linear in both the size of the circuit and the size of the dataset, and our algorithm is also guaranteed to converge (potentially to a local minimum) as every iteration only decreases or preserves the empirical Wasserstein objective (Appendix B.8). Nevertheless, finding the global optimum parameters minimizing the Wasserstein distance is still NP-hard, and our proposed efficient algorithm may get stuck at a local minimum, similar to existing maximum-likelihood parameter learning approaches.

3.1. Experimental Results

To determine the performance of our proposed Wasserstein minimization algorithm, we consider learning the parameters of circuits of various sizes from the MNIST benchmark dataset [8]. When compared to mini-batch Expectation Maximization (EM) for estimating maximum-likelihood parameters, our Wasserstein Minimization (WM) approach is nearly competitive for small circuits but falls behind for larger circuits. We attribute this to WM's inability to make use of the parameter space of larger models. Detailed experimental results are provided in Appendix C.4.

4. Conclusion

This paper studied the optimal transport problem for probabilistic circuits. We introduced a Wasserstein-type distance CW_p between two PCs an proposed an efficient algorithm that computes the distance and corresponding optimal transport plan in quadratic time in the size of the input circuits, provided that their circuit structures are compatible. We show that CW_p always upper-bounds the true Wasserstein distance, and that—when compared to the naive application of an existing algorithm for computing a Wasserstein-type distance between GMMs to PCs—the former is exponentially faster to compute between circuits. Lastly, we propose an iterative algorithm to minimize the empirical Wasserstein distance between a circuit and data, suggesting an alternative, viable approach to parameter estimation for PCs which is mainly done using maximum-likelihood estimation. While performance was competitive with the EM algorithm for small circuits, and we leave as future work to get Wasserstein Minimization to fully exploit the increased expressiveness of larger models.

We consider this work an initial stepping stone towards a deeper understanding of optimal transport theory for probabilistic circuits. Future work includes exploring more expressive formulations of coupling circuits to close the gap between CW_p and MW_p, extending the marginal-preserving properties of coupling circuits to the multimarginal setting, and computing Wasserstein barycenters for PCs.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*,

pages 214–223. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/arjovsky17a.html.

[2] Yongxin Chen, Tryphon T Georgiou, and Allen Tannenbaum. Optimal transport for gaussian mixture models. *IEEE Access*, 7:6269–6278, 2018.

[3] YooJung Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. oct 2020. URL http://starai.cs.ucla.edu/papers/ProbCirc20.pdf.

[4] Sanjoy Dasgupta. The hardness of k-means clustering. UCSD Technical Report, 2008. URL https://www.cs.columbia.edu/~verma/classes/uml/ref/clustering_kmeans_NPhard_dasgupta.pdf.

[5] Julie Delon and Agnes Desolneux. A wasserstein-type distance in the space of gaussian mixture models. *SIAM Journal on Imaging Sciences*, 13(2):936–970, 2020.

[6] Mattia Desana and Christoph Schnörr. Expectation maximization for sum-product networks as exponential family mixture models. 04 2016. doi: 10.48550/arXiv.1604.07243.

[7] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024. URL https://www.gurobi.com.

[8] Yann LeCun, Léon Bottou, Yoshua Bengio, , and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[9] Yitao Liang and Guy Van den Broeck. Towards compact interpretable models : Shrinking of learned probabilistic sentential decision diagrams. In *IJCAI 2017 Workshop on Explainable Artificial Intelligence (XAI)*, 2017. URL http://starai.cs.ucla.edu/papers/LiangXAI17.pdf.

[10] Anji Liu and Guy Van den Broeck. Tractable regularization of probabilistic circuits. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 3558–3570. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/1d0832c4969f6a4cc8e8a8fffe083efb-Paper.pdf.

[11] Anji Liu, Kareem Ahmed, and Guy Van den Broeck. Scaling tractable probabilistic circuits: A systems perspective, 2024. URL https://arxiv.org/abs/2406.00766.

[12] Roberto Tamassia Michael Goodrich. *Algorithm Design: Foundations, Analysis, and Internet Examples*. John Wiley & Sons, 2002.

[13] Litu Rout, Alexander Korotin, and Evgeny Burnaev. Generative modeling with optimal transport maps. In *International Conference on Learning Representations*, 2022.

[14] Tim Salimans, Dimitris Metaxas, Han Zhang, and Alec Radford. Improving gans using optimal transport. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[15] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.

[16] Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso, and Guy den Broeck. A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference. In *Advances in Neural Information Processing Systems*, volume 34, pages 13189–13201, 2021.

Algorithm 1 COUPLE(n, m): coupling circuit that optimizes $\mathsf{CW}_p^p(n, m)$ of compatible PCs rooted at nodes n, m

Note: We omit calls to a cache storing previously-computed coupling circuits COUPLE(n, m) for simplicity.

if n, m *are input nodes* **then**
  $r \leftarrow$ new product(n, m) ▷ Product node with children n,m
end
if n, m *are sum nodes* **then**
  $r \leftarrow$ new sum node with parameters $\theta_{i,j}$
  foreach $c_i \in n.children$, $c_j \in m.children$ **do**
    r.children$[i, j] \leftarrow$ COUPLE(c_i, c_j)
  end
  $$\mathsf{LP} \leftarrow \begin{cases} \text{minimize} & \sum_i \sum_j \mathsf{CW}_p(r.\text{children}[i,j]) * \theta_{i,j} \\ \text{subject to} & \forall i, \sum_j \theta_{i,j} = \theta_i \\ & \forall j, \sum_i \theta_{i,j} = \theta_j \\ & \theta_{i,j} \in [0, 1] \end{cases}$$
  solve LP ▷ Solve for optimal parameters $\theta_{i,j}$
end
if n, m *are product nodes* **then**
  $r \leftarrow$ new product node
  foreach $c_1 \in n.children$, $c_2 \in m.children$ ***where*** $\mathsf{sc}(c_1) = \mathsf{sc}(c_2)$ **do**
    add COUPLE(c_1, c_2) to r.children ▷ Child is the coupling of corresponding children
  end
end
return n

Appendix A. Algorithms

A.1. Algorithm for Computing the Coupling Circuit between PCs

Algorithm 1 details the construction of a coupling circuit and the computation of the optimal parameters for sum nodes. LP represents a linear program and we assume that sum nodes in n_1 have i children and sum nodes in n_2 have j children. With caching of both $\mathsf{CW}_p(n)$ and COUPLE(n_1, n_2) calls, this algorithm runs in quadratic time.

A.2. Algorithm for Computing the Wasserstein Objective for a Coupling Circuit

Given a coupling circuit rooted at n, Algorithm 2 computes the value of the Wasserstein objective (see Definition 1) for the coupling. With caching, this algorithm runs in linear time.

A.3. Algorithm for Minimum Wasserstein Parameter Estimation

Our proposed algorithm is broadly divided into two steps: an inference step and a minimization step. These steps are performed iteratively until model convergence. The inference step populates

Algorithm 2 $\mathsf{CW}_p(n)$: p-Wasserstein objective for a coupling circuit rooted at n with i children.

if *n is a product node with input node children* **then**
  return $\mathsf{W}_p(n.\text{children}[0], n.\text{children}[1])$
end
if *n is a product node without input node children* **then**
  return $\sum_i \mathsf{CW}_p(n.\text{children}[i])$
end
if *n is a sum node* **then**
  return $\sum_i \theta_i \mathsf{CW}_p(n.\text{children}[i])$
end

a cache, which stores the expected distance of each data point at each node in the circuit. This inference step is done in linear time in a bottom-up recursive fashion, making use of the cache for already-computed results. This is provided in algorithm 3.

The minimization step is done top-down recursively, and seeks to route the data at a node to its children in a way that minimizes the total expected distance between the routed data at each child and the sub-circuit. The root node is initialized with all data routed to it. At a sum node, each data point is routed to the child that has the smallest expected distance to it (making use of the cache from the inference step), and the edge weight corresponding to a child is equal to the proportion of data routed to that child; at a product node, the data point is routed to both children. Input node parameters are updated to reflect the empirical distribution of the data routed to that node. The minimization step is thus also done in linear time, and we note that this algorithm guarantees a non-decreasing objective function (see Appendix B.8 for a proof). The algorithm for this is provided in algorithm 4.

Appendix B. Proofs

B.1. Hardness Proof of the ∞-Wasserstein Distance Between Circuits

Theorem 1 *Suppose P and Q are probabilistic circuits over n Boolean variables. Then computing the ∞-Wasserstein distance between P and Q is coNP-hard, even when P and Q are deterministic and structured-decomposable.*

Proof We will prove hardness by reducing the problem of deciding equivalence of two DNF formulas, which is known to be coNP-hard, to Wasserstein distance computation between two compatible PCs.

Consider a DNF α containing m terms $\{\alpha_1, \ldots, \alpha_m\}$ over Boolean variables $\mathbf{X}$. We will construct a PC P_α associated with this DNF as follows. For each term α_i, we construct a product of input nodes—one for each $X \in \mathbf{X}$ whose literal appears in term α_i, $\mathbb{1}[X = 1]$ for a positive literal and $\mathbb{1}[X = 0]$ for negative. Then we construct a sum unit with uniform parameters over these products as the root of our PC: $P_\alpha = \sum_{i=1}^{m} \frac{1}{m} P_{\alpha_i}$. We can easily smooth this PC by additionally multiplying P_{α_i} with a sum node $\frac{1}{2}\mathbb{1}[X = 0] + \frac{1}{2}\mathbb{1}[X = 1]$ for each variable X that does not appear in α_i. Furthermore, note that every product node in this circuit fully factorizes the variables $\mathbf{X}$, and thus the PC is trivially compatible with any decomposable circuit over $\mathbf{X}$ and in particular with any other PC for a DNF over $\mathbf{X}$, constructed as above.

Algorithm 3 INFERENCE(n, D): returns a cache storing the distance between each datapoint $d_j \in D$ and each sub-circuit rooted at n, where n has children c_i. For conciseness, we omit checking for cache hits

for $c_i \in n$.*children* **do**
  INFERENCE(c_i, D) ▷ recursively build cache
end
if n *is a product node* **then**
  for $d_j \in D$ **do**
    cache$[n, d_j] \leftarrow \sum_i$cache$[c_i, d_j]$
  end
end
if n *is a sum node* **then**
  for $d_j \in D$ **do**
    cache$[n, d_j] \leftarrow \sum_i \theta_i$cache$[c_i, d_j]$
  end
end
if n *is an input node* **then**
  for $d_j \in D$ **do**
    cache$[n, d_j] \leftarrow dist(n, d_j)$ ▷ here, $dist(n, d_j)$ is the expected distance between n and d_j
  end
end
return cache

Algorithm 4 LEARN$(n, D,$ cache$)$: learns the parameters of circuit rooted at n on datapoints $d_j \in D$

if *not all parents of* n *have been learned* **then**
  return ▷ We only call this method on nodes who's parents have all been learned
end
if n *is a product node* **then**
  for $c_i \in n$.*children* **do**
    routing$[c_i] \leftarrow$ routing$[n]$ ▷ products route their data to their children
  end
end
if n *is a sum node* **then**
  $\forall \theta_i, \theta_i \leftarrow 0$ ▷ zero out parameters
  for $d_j \in$ *routing*$[n]$ **do**
    ▷ route data points at current node to children
    $c_i \leftarrow \arg\min_{c_i}$ cache$[c_i, d_j]$ ▷ c_i is the child node of n for which d_j has the lowest distance
    routing$[c_i] \leftarrow d_j$ ▷ route d_j to c_i
    $\theta_i \leftarrow \theta_i + \frac{1}{|\text{routing}[n]|}$ ▷ update parameter weight
  end
end
if n *is an input node* **then**
  n.parameters $\leftarrow$ parameters matching empirical distribution of routing$[n]$
end

Clearly, the above PC P_α assigns probability mass only to the models of α. In other words, for any $\mathbf{x} \in \{0,1\}^n$, $P_\alpha(\mathbf{x}) > 0$ iff $\mathbf{x} \models \alpha$ (i.e. there is a term α_i that is satisfied by $\mathbf{x}$). ■

B.2. Recursive Computation of the Wasserstein Objective

Referring to Definition 1, the Wasserstein objective for a given coupling circuit $C(\mathbf{x}, \mathbf{y})$ is the expected distance between $\mathbf{x}$ and $\mathbf{y}$. Below, we demonstrate that the Wasserstein objective at a sum node that decomposes into $C(\mathbf{x}, \mathbf{y}) = \sum_i \theta_i C_i(\mathbf{x}, \mathbf{y})$ is simply the weighted sum of the Wasserstein objectives at its children:

$$\mathbb{E}_{C(\mathbf{x},\mathbf{y})}[\|\mathbf{x}-\mathbf{y}\|_p^p] = \int \|\mathbf{x}-\mathbf{y}\|_p^p \, C(\mathbf{x},\mathbf{y}) d\mathbf{x} d\mathbf{y} = \int \|\mathbf{x}-\mathbf{y}\|_p^p \sum_i \theta_i C_i(\mathbf{x},\mathbf{y}) d\mathbf{x} d\mathbf{y}$$
$$= \sum_i \theta_i \int \|\mathbf{x}-\mathbf{y}\|_p^p \, C_i(\mathbf{x},\mathbf{y}) d\mathbf{x} d\mathbf{y} = \sum_i \theta_i \, \mathbb{E}_{C_i(\mathbf{x},\mathbf{y})}[\|\mathbf{x}-\mathbf{y}\|_p^p] \tag{3}$$

Now, consider a decomposable product node, where $C(\mathbf{x}, \mathbf{y}) = C_1(\mathbf{x}_1, \mathbf{y}_1) C_2(\mathbf{x}_2, \mathbf{y}_2)$ [3]. Below, we see that the Wasserstein objective at the parent is simply the *sum* of the Wasserstein objectives at its children:

$$\mathbb{E}_{C(\mathbf{x},\mathbf{y})}[\|\mathbf{x}-\mathbf{y}\|_p^p] = \int \|\mathbf{x}-\mathbf{y}\|_p^p \, C(\mathbf{x},\mathbf{y}) d\mathbf{x} d\mathbf{y} = \int \|\mathbf{x}-\mathbf{y}\|_p^p \, C_1(\mathbf{x}_1,\mathbf{y}_1) C_2(\mathbf{x}_2,\mathbf{y}_2) d\mathbf{x} d\mathbf{y}$$
$$= \int (\|\mathbf{x}_1-\mathbf{y}_1\|_p^p + \|\mathbf{x}_2-\mathbf{y}_2\|_p^p) \times C_1(\mathbf{x}_1,\mathbf{y}_1) C_2(\mathbf{x}_2,\mathbf{y}_2) d\mathbf{x}_1 d\mathbf{x}_2 d\mathbf{y}_1 d\mathbf{y}_2$$
$$= \left(\int \|\mathbf{x}_1-\mathbf{y}_1\|_p^p \, C_1(\mathbf{x}_1,\mathbf{y}_1) d\mathbf{x}_1 d\mathbf{y}_1 \right) + \left(\int \|\mathbf{x}_2-\mathbf{y}_2\|_p^p) C_2(\mathbf{x}_2,\mathbf{y}_2) d\mathbf{x}_2 d\mathbf{y}_2 \right)$$
$$= \mathbb{E}_{C_1(\mathbf{x}_1,\mathbf{y}_1)}[\|\mathbf{x}_1-\mathbf{y}_1\|_p^p] + \mathbb{E}_{C_2(\mathbf{x}_2,\mathbf{y}_2)}[\|\mathbf{x}_2-\mathbf{y}_2\|_p^p] \tag{4}$$

Thus, we can push computation of Wasserstein objective down to the leaf nodes of a coupling circuit.

B.3. Proof of the Metric Properties of CW_p

Proposition 1 (Metric Properties of CW_p) *For any set $\mathcal{C}$ of compatible circuits, CW_p defines a metric on $\mathcal{C}$*

Proof It is clear that CW_p is symmetric since construction of the coupling circuit is symmetric. Furthermore, since CW_p upper-bounds W_p, it must also be non-negative.

If $\mathsf{CW}_p(P, Q) = 0$, then $\mathsf{W}_p(P, Q) = 0$ so $P = Q$. Any constraint-satisfying assignment of the parameters of a coupling circuit between P and P would also result in the Wasserstein objective at the root node being 0, since the base-case computation of W_p at the leaf nodes would always be zero.

Now, we show that CW_p satisfies the triangle inequality. Let $P, Q, R \in \mathcal{C}$ be compatible PCs over random variables $\mathbf{X}, \mathbf{Y}$, and $\mathbf{Z}$, and let $d_1 = \mathsf{CW}_p(P, Q)$, $d_2 = \mathsf{CW}_p(P, R)$, and $d_3 = \mathsf{CW}_p(R, Q)$ with optimal coupling circuits C_1, C_2, and C_3. We can construct circuits $C_2(\mathbf{x}|\mathbf{z})$ and

3. We assume for notational simplicity that product nodes have two children, but it is straightforward to rewrite a product node with more than two children as a chain of product nodes with two children each and see that our result still holds.

$C_3(\mathbf{y}|\mathbf{z})$ that are still compatible with C_2 and C_3, since conditioning a circuit preserves the structure. Because all of these are compatible, we can then construct circuit $C(\mathbf{X}, \mathbf{Y}, \mathbf{Z}) = C_2(\mathbf{X}|\mathbf{Z})C_3(\mathbf{Y}|\mathbf{Z})R(\mathbf{Z})$. Thus, C is a coupling circuit of P, Q, and R such that $C_2(\mathbf{x}, \mathbf{y}) = \int C(\mathbf{x}, \mathbf{y}, \mathbf{z})d\mathbf{z}$ and $C_3(\mathbf{y}, \mathbf{z}) = \int C(\mathbf{x}, \mathbf{y}, \mathbf{z})d\mathbf{x}$. Then we have:

$$\begin{aligned}\mathsf{CW}_p(P, Q) &= \int \|\mathbf{x} - \mathbf{y}\|_p^p \, C_1(\mathbf{x}, \mathbf{y})d\mathbf{x}d\mathbf{y} = \int \|(\mathbf{x} - \mathbf{z}) - (\mathbf{y} - \mathbf{z})\|_p^p \, C(\mathbf{x}, \mathbf{y}, \mathbf{z})d\mathbf{x}d\mathbf{y}d\mathbf{z} \\ &\leq \int \|\mathbf{x} - \mathbf{z}\|_p^p \, C_2(\mathbf{x}, \mathbf{z})d\mathbf{x}d\mathbf{z} + \int \|\mathbf{z} - \mathbf{y}\|_p^p \, C_3(\mathbf{y}, \mathbf{z})d\mathbf{y}d\mathbf{z} \\ &= \mathsf{CW}_p(P, R) + \mathsf{CW}_p(R, Q)\end{aligned}$$

Thus, CW_p satisfies the triangle inequality, which concludes the proof. ■

B.4. Proof of the Optimality of Coupling Circuit Parameter Learning in A.1

Theorem 2 *Suppose P and Q are compatible probabilistic circuits with coupling circuit C. Then the parameters of C - and thus CW_p - can be computed exactly in a bottom-up recursive fashion.*

Proof We will construct a recursive argument showing that the optimal parameters of C can be computed exactly. Let $n \in C$ be some non-input node in the coupling circuit C that is the product of nodes n_1 and n_2 in P and Q respectively. Then we have three cases:

Case 1: n is a product node with input node children Due to the construction of the coupling circuit, n must have two children that are input nodes with scopes $\mathbf{X}_k$ and $\mathbf{Y}_k$. Thus, $\mathsf{CW}_p(n)$ is simply computed in closed-form as the p-Wasserstein distance between the input distributions.

Case 2: n is a product node with non-input node children By recursion, $\mathsf{CW}_p(n) = \sum_i \mathsf{CW}_p(c_i)$ for each child c_i of n (see 4).

Case 3: n is a sum node Let $\theta_{i,j}$ be the parameter corresponding to the product of the i-th child of n_1 and j-th child of n_2. We want to solve the following optimization problem $\inf \mathbb{E}_{P_n(\mathbf{X},\mathbf{Y})}[\|\mathbf{X} - \mathbf{Y}\|_p^p]$, which can be rewritten as follows:

$$\inf \mathbb{E}_{P_n(\mathbf{X},\mathbf{Y})}[\|\mathbf{X} - \mathbf{Y}\|_p^p] = \inf \int_{\mathbb{R}^d \times \mathbb{R}^d} \|\mathbf{X} - \mathbf{Y}\|_p^p \, P_n(\mathbf{X}, \mathbf{Y})d\mathbf{X}d\mathbf{Y} \tag{5}$$

Rewriting the distribution of n as a mixture of its child distributions $c_{i,j}$, we get:

$$= \inf_{\theta, P_{i,j}} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|\mathbf{X} - \mathbf{Y}\|_p^p \sum_{i,j} \theta_{i,j} P_{c_{i,j}}(\mathbf{X}, \mathbf{Y})d\mathbf{X}d\mathbf{Y} \tag{6}$$

Due to linearity of integrals, we can bring out the sum:

$$= \inf_{\theta, P_{i,j}} \sum_{i,j} \theta_{i,j} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|\mathbf{X} - \mathbf{Y}\|_p^p \, P_{c_{i,j}}(\mathbf{X}, \mathbf{Y})d\mathbf{X}d\mathbf{Y} \tag{7}$$

Lastly, due to the acyclicity of PCs, we can separate out $\inf_{\theta_i, P_{i,j}}$ into $\inf_{\theta_i} \inf_{P_{i,j}}$ and push the latter infimum inside the sum.

$$= \inf_{\theta} \sum_{i,j} \theta_{i,j} (\inf_{P_{i,j}} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|\mathbf{X} - \mathbf{Y}\|_p^p \, P_{c_{i,j}}(\mathbf{X}, \mathbf{Y})d\mathbf{X}d\mathbf{Y}) \tag{8}$$

Thus, we can solve the inner optimization problem first (corresponding to the optimization problems at the children), and then the outer problem (the optimization problem at the current node). Therefore, a bottom-up recursive algorithm is exact. ■

B.5. Proof of the Marginal-Matching Properties of Coupling Circuits

Theorem 3 *Let P and Q be compatible PCs. Then any feasible coupling circuit C as defined in Def. 4 matches marginals to P and Q.*

Proof We will prove this by induction. Our base case is two corresponding input nodes $n_1, n_2 \in P, Q$. The sub-circuit in C rooted at the product of n_1 and n_2 is a product node with copies of n_1 and n_2 as children, which clearly matches marginals to n_1 and n_2.

Now, let n_1 and n_2 be arbitrary corresponding nodes in P and Q, and assume that the product circuits for all children of the two nodes match marginals. We then have two cases:

Case 1: n_1, n_2 are product nodes Since the circuits are compatible, we know that n_1 and n_2 have the same number of children - let the number of children be k. Thus, let $c_{1,i}$ represent the i'th child of n_1, and let $c_{2,i}$ represent the i'th child of n_2. The coupling circuit of n_1 and n_2 (denoted n) is a product node with k children, where the i'th child is the coupling circuit of $c_{1,i}$ and $c_{2,i}$ (denoted c_i).

By induction, the distribution $P_{c_i}(\mathbf{X}, \mathbf{Y})$ at each child coupling sub-circuit matches marginals to the original sub-circuits: $P_{c_i}(\mathbf{X}) = P_{c_{1,i}}(\mathbf{X})$, and $P_{c_i}(\mathbf{Y}) = P_{c_{2,i}}(\mathbf{Y})$. n_1 and n_2 being product nodes means that $P_{n_1}(\mathbf{X}) = \prod_i P_{c_{1,i}}(\mathbf{X})$ and $P_{n_2}(\mathbf{Y}) = \prod_i P_{c_{2,i}}(\mathbf{Y})$, so thus $P_n(\mathbf{X}) = \prod_i P_{c_i}(\mathbf{X}) = \prod_i P_{c_{1,i}}(\mathbf{X})$ and $P_n(\mathbf{Y}) = \prod_i P_{c_i}(\mathbf{Y}) = \prod_i P_{c_{2,i}}(\mathbf{Y})$. Therefore, n matches marginals to n_1 and n_2.

Case 2: n_1, n_2 are sum nodes Let the number of children of n_1 be k_1 and the number of children of n_2 be k_2. Let $c_{1,i}$ represent the i'th child of n_1, and let $c_{2,j}$ represent the j'th child of n_2. The coupling circuit of n_1 and n_2 (denoted n) is a sum node with $k_1 * k_2$ children, where the (i, j)'th child is the coupling circuit of $c_{1,i}$ and $c_{2,j}$ (denoted $c_{i,j}$).

By induction, the distribution $P_{c_{i,j}}(\mathbf{X}, \mathbf{Y})$ at each child coupling sub-circuit matches marginals to the original sub-circuits: $P_{c_{i,j}}(\mathbf{X}) = P_{c_{1,i}}(\mathbf{X})$, and $P_{c_{i,j}}(\mathbf{Y}) = P_{c_{2,j}}(\mathbf{Y})$. n_1 and n_2 being sum nodes means that $P_{n_1}(\mathbf{X}) = \sum_i \theta_i P_{c_{1,i}}(\mathbf{X})$ and $P_{n_2}(\mathbf{Y}) = \sum_j \theta_j P_{c_{2,j}}(\mathbf{Y})$, so thus

$$P_n(\mathbf{X}) = \sum_i \sum_j \theta_{i,j} P_{c_{i,j}}(\mathbf{X}) = \sum_i \sum_j \theta_{i,j} P_{c_{1,i}}(\mathbf{X}) = \sum_i \theta_i P_{c_{1,i}}(\mathbf{X}) = P_{n_1}(\mathbf{X})$$
$$P_n(\mathbf{Y}) = \sum_i \sum_j \theta_{i,j} P_{c_{i,j}}(\mathbf{Y}) = \sum_i \sum_j \theta_{i,j} P_{c_{2,j}}(\mathbf{Y}) = \sum_j \theta_j P_{c_{2,j}}(\mathbf{Y}) = P_{n_2}(\mathbf{Y}) \quad (9)$$

Note that we rewrite $\sum_i \theta_{i,j} = \theta_j$ and $\sum_j \theta_{i,j} = \theta_i$ by the constraints on coupling circuits. Therefore, n satisfies marginal constraints. ■

B.6. Proving that Computing Minimum Wasserstein Parameters is NP-Hard

Theorem 4 *Computing the parameters of probabilistic circuit C is NP-hard.*

Proof We will prove this hardness result by reducing k-means clustering - which is known to be NP-hard [4] - to learning the minimum Wasserstein parameters of a circuit. Consider a set of points $x_1 ... x_n \in \mathbb{R}^d$ and a number of clusters k. We will construct a Gaussian PC C associated with this problem as follows: the root of C is a sum node with k children; each child is a product node with d univariate Gaussian input node children (so each product node is a multivariate Gaussian comprised of independent univariate Gaussians). Minimizing the parameters of C over x_i corresponds to finding a routing of data points x_i that minimizes the total distance between all x_i's and the mean of the multivariate Gaussian child each x_i is routed to. A solution to k-means can be retrieved by taking the mean of each child of the root sum node to be the center of each of k clusters. ■

B.7. Deriving a Closed-Form Solution to the Linear Programs for Parameter Updates

For a sum node s with m children $s_1 ... s_m$ and a dataset with n datapoints $d_1 ... d_n$ each with weight w_j, we construct a linear program with $m * n$ variables $x_{i,j}$ as follows:

$$\min \quad \sum_{i=1}^{m}\sum_{j=1}^{n} \mathbb{E}_{s_i}[\|\mathbf{X} - d_j\|_2^2] x_{i,j} \qquad \text{s.t.} \quad \sum_{i=1}^{m} x_{i,j} = w_j \; \forall j$$

Note that the constraints do not overlap for differing values of j. Thus, we can break this problem up into n smaller linear programs, each with the following form:

$$\min \quad \sum_{i=1}^{m} \mathbb{E}_{s_i}[\|\mathbf{X} - d_j\|_2^2] x_{i,j} \qquad \text{s.t.} \quad \sum_{i=1}^{m} x_{i,j} = w_j$$

The only constraint here requires that the sum of objective variables is equal to w_j. Thus, the objective is minimized when $x_{i,j}$ corresponding to the smallest coefficient takes value w_j and all other variables take value 0. Thus, the solution to the original linear program can be thought of as assigning each data point to the sub-circuit that has the smallest expected distance to it.

B.8. Proof that the Wasserstein Minimization Algorithm has a Monotonically Decreasing Objective

Theorem 5 *For a circuit rooted at n and dataset D routed to it, the Wasserstein distance between the empirical distribution of D and sub-circuit rooted at n will not increase after an iteration of algorithm A.3*

Proof Let $\mathbb{E}_n[D]$ denote the Wasserstein distance between the empirical distribution of D and sub-circuit rooted at n before an iteration of algorithm A.3, and let $\mathbb{E}_{n'}[D]$ denote the distance after an iteration. We will show by induction that $\mathbb{E}_{n'}[D] \leq \mathbb{E}_n[D]$. Our base case is when n is an input node. By setting the parameters of n to as closely match the empirical distribution of D as possible, there is no parameter assignment with a lower Wasserstein distance to D so thus one iteration of algorithm A.3 does not increase the objective value.

Recursively, we have two cases:

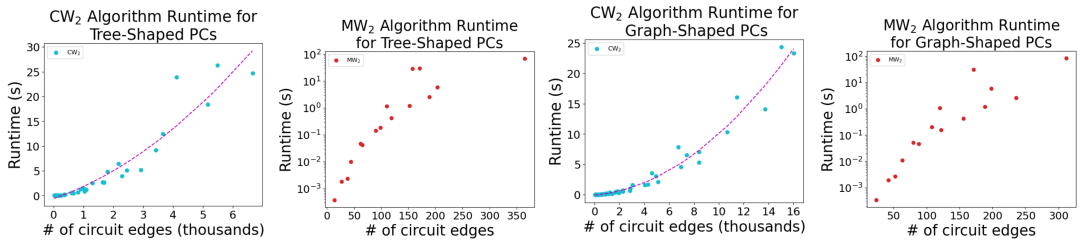

Figure 2: Runtime for algorithms computing CW_2 and MW_2. The first pair of graphs considers only tree-shaped PCs, whereas the second pair considers graph-shaped PCs as well. Note that the right-side graphs use logarithmic scaling. Number of circuit edges represents the number of edges in both circuits combined, and each data point represents an average over 100 runs.

Case 1: n is a product node By the decomposition of the Wasserstein objective, we have that $\mathbb{E}_n[D] = \sum_i \mathbb{E}_{c_i}[D]$, which is $\geq \sum_i \mathbb{E}_{c'_i}[D] = \mathbb{E}_{n'}[D]$ by induction.

Case 2: n is a sum node By the decomposition of the Wasserstein objective, we have that $\mathbb{E}_n[D] = \sum_i \theta_i \mathbb{E}_{c_i}[D_i]$ (where $D_i \subseteq D$ is the data routed to n_i), which is $\geq \sum_i \theta_i \mathbb{E}_{c'_i}[D_i] = \mathbb{E}_{n'}[D]$ by induction. Our parameter updates also update each $D_i \to D'_i$, but that also guarantees that $\mathbb{E}_{c'_i}[D_i] \geq \mathbb{E}_{c'_i}[D'_i]$ since $D_i = D'_i$ is within the feasible set of updates for D_i. Thus, $\mathbb{E}_n[D] \geq \mathbb{E}_{n'}[D]$, so therefore the Wasserstein objective is monotonically decreasing. ■

Appendix C. Additional Experimental Results

C.1. Additional CW_p Runtime Results

To evaluate the runtime of computing CW_2, we consider a fixed variable scope and randomly construct a balanced region tree for the scope. Then, we randomly construct two PCs for this region tree; the PCs are constructed with a fixed sum node branching factor and fixed rejoin probability - which is the chance that a graph connection to an existing node in the PC will be made to add a child rather than creating a new node for the child, and is 0% in the case of trees and 50% in the case of graphs. We implement our algorithm as detailed in appendix A.1 to compute the optimal transport map and value for CW_2, as well as also implement a PC-to-GMM unrolling algorithm and the algorithm proposed by Chen et al. [2] to compute MW_2 [5]. The value obtained for each circuit size is averaged over 100 runs, and we omit data points for experiments that ran out of memory. See Figure 2 for the graphs.

The experiments were conducted on a machine with an Intel Core i9-10980XE CPU and 256Gb of RAM (these experiments made no use of GPUs); linear programs were solved using Gurobi [7]. Each experiment was conducted with a fixed random seed, and the parameter values for sum nodes were clamped to be greater than 0.01 for numeric stability.

C.2. Comparing CW_2 and MW_2

To evaluate the proximity of CW_2 to MW_2, we adopt the same framework as we did for runtime experiments to randomly construct compatible PCs and compute CW_2 and MW_2 between them. Due to the exponential blowup of computing MW_2 it quickly becomes impractical to compute (see

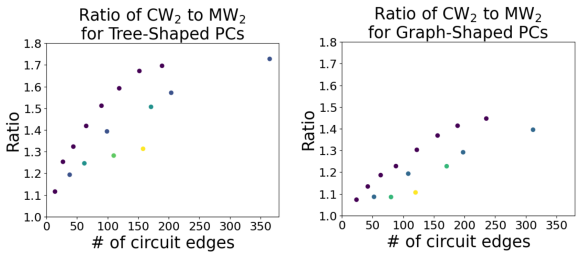

Figure 3: Ratio of $\frac{\mathsf{CW}_2}{\mathsf{MW}_2}$, lower is better. Empirically, the gap between CW_2 and MW_2 grows roughly linearly in the size of the circuit. The hue of each point represents the circuit depth, with lighter points being a higher depth. Circuit depth does not seem to affect the ratio significantly.

runtime experiments in Appendix C.1); however, we still attempt to provide some empirical insight into the difference between CW_p and MW_p.

We note that the ratio $\frac{\mathsf{CW}_2}{\mathsf{MW}_2}$ appears to grow linearly in the size of the circuit; furthermore, for graph-shaped circuits, the ratio is closer to 1 than for tree-shaped circuits. Figure 3 provides an in-depth look at these observations.

C.3. Visualizing Transport Maps between Circuits

Since our algorithm does not only return CW_p between two circuits but also the corresponding transport plan, we can visualize the transport of point densities between the two distributions by conditioning the coupling circuit on an assignment of random variables in one circuit. We can similarly visualize the transport map for an arbitrary region in one PC to another by conditioning on the random variable assignments being within said region. See Figure 4 for an example.

Since the transport plan for a single point (or region of points) is itself a PC, we can query it like we would any other circuit; for example, computing *maximum a posteriori* - which is tractable if the original two circuits are marginal-deterministic [3] - for the transport plan of a point corresponds to the most likely corresponding point in the second distribution for the given point. Because a coupling circuit inherits the structural properties of the original circuit, it is straightforward to understand what queries are and are not tractable for a point transport map.

C.4. Empirical Wasserstein Parameter Estimation Experimental Results

To understand the effectiveness of parameter estimation via minimizing the empirical Wasserstein distance, we evaluated the performance of PCs trained using the HCLT [10] structure with categorical input distributions on the MNIST [8] dataset. The baseline for this experiment is the EM algorithm for circuits [6].

We first generated the structure of the circuits using the HCLT implementation provided in PyJuice [11], varying the size of each block to increase or decrease the number of parameters. We then learned two sets of circuit parameters per structure per block size: one set of parameters was learned using mini-batch EM with a batch size of 1000, and the other set was learned using an implementation of the algorithm detailed in Appendix A.3. We perform early stopping for the EM algorithm that stops training once the point of diminishing returns has been surpassed. All experiments were ran on a single NVIDIA L40s GPU.

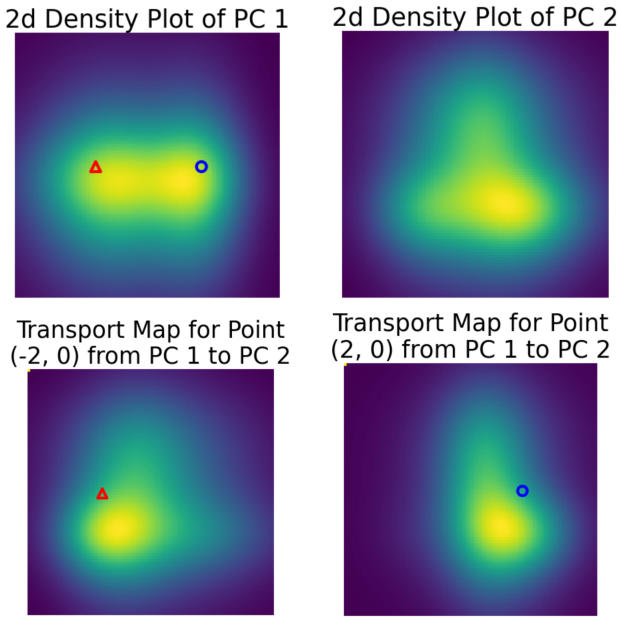

Figure 4: Visualization of transporting the indicated points from the distribution parameterized by PC 1 to the distribution parameterized by PC 2. The top two figures visualize the distributions, while the bottom two figures visualize where the point density indicated is transported to from the first to the second distribution.

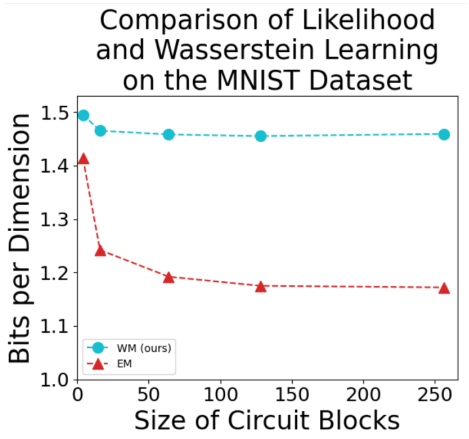

Figure 5: Visualization of the performance of PCs learned using Expectation Maximization (red triangles) and Wasserstein Minimization (our approach, blue dots). The bits-per-dimension (bpd) of the learned circuits does not decrease significantly with an increase in circuit size for circuits learned using the empirical Wasserstein distance.

When using bits-per-dimension as a benchmark, we observe that our algorithm performs nearly as well as EM for small circuits (block size of 4). However, as the size of the circuit increases, the performance of our algorithm hardly improves; empirically, our approach to Wasserstein minimization does not make good use of the larger parameter space of larger models, with models that are orders of magnitude larger having better but still comparable performance to their smaller counterparts. We refer to Figure 5 for more details.