OT-Flow: Fast and Accurate Continuous Normalizing Flows via Optimal Transport

Derek Onken[§], Samy Wu Fung[†], Xingjian Li[§], Lars Ruthotto[§]

[§]Departments of Mathematics and Computer Science at Emory University, [†]Department of Mathematics at UCLA

Motivation

Continuous Normalizing Flows (CNFs)

A normalizing flow [1] is an invertible mapping $f\colon \mathbb{R}^d \to \mathbb{R}^d$ between an arbitrary probability distribution and a standard normal distribution whose densities we denote by ρ_0 and ρ_1, respectively.

By change of variables,

$$\log \rho_0(\boldsymbol{x}) = \log \rho_1(f(\boldsymbol{x})) + \log|\det \nabla f(\boldsymbol{x})| \quad \text{for all} \quad \boldsymbol{x} \in \mathbb{R}^d. \qquad (1)$$

In CNFs, f solves the neural ordinary differential equation (ODE) [2, 3]

$$\partial_t \begin{bmatrix} \boldsymbol{z}(\boldsymbol{x},t) \\ \ell(\boldsymbol{x},t) \end{bmatrix} = \begin{bmatrix} \mathbf{v}(\boldsymbol{z}(\boldsymbol{x},t),t;\boldsymbol{\theta}) \\ \operatorname{tr}\left(\nabla \mathbf{v}(\boldsymbol{z}(\boldsymbol{x},t),t;\boldsymbol{\theta})\right) \end{bmatrix}, \quad \begin{bmatrix} \boldsymbol{z}(\boldsymbol{x},0) \\ \ell(\boldsymbol{x},0) \end{bmatrix} = \begin{bmatrix} \boldsymbol{x} \\ 0 \end{bmatrix}, \qquad (2)$$

where, for artificial time $t \in [0,T]$,

- $\boldsymbol{x}$ maps to $f(\boldsymbol{x}) = \boldsymbol{z}(\boldsymbol{x},T)$ following trajectory $\boldsymbol{z}\colon \mathbb{R}^d \times [0,T] \to \mathbb{R}^d$
- $\mathbf{v}\colon \mathbb{R}^d \times [0,T] \to \mathbb{R}^d$ is a neural network layer parameterized by $\boldsymbol{\theta}$
- $\ell(\boldsymbol{x},T) = \log\det \nabla f(\boldsymbol{x})$, derived from Jacobi's Formula [2]

Microscale: an arbitrary sample $\boldsymbol{x}$ maps to a normally distributed $f(\boldsymbol{x})$
Macroscale: ρ_0 maps to ρ_1

CNFs are trained by solving the optimization problem [1, 3]

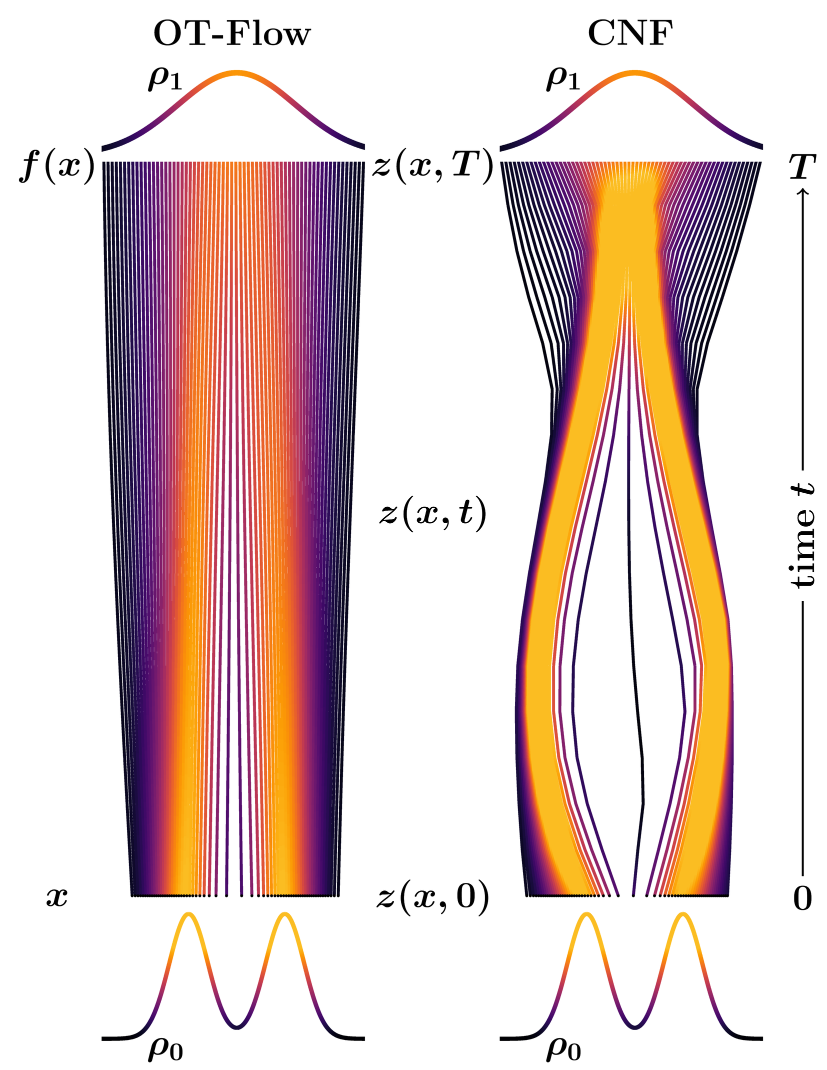


Figure 1: While a CNF can have curved trajectories, OT-Flow's are straight (modification of Fig. 1 in [3, 4]).

$$\min_{\boldsymbol{\theta}} \; \mathbb{E}_{\rho_0(\boldsymbol{x})} \left\{ C(\boldsymbol{x},T) := \frac{1}{2}\|\boldsymbol{z}(\boldsymbol{x},T)\|^2 - \ell(\boldsymbol{x},T) + \frac{d}{2}\log(2\pi) \right\} \quad \text{s.t.} \quad (2). \qquad (3)$$

High Training Costs

- Many functions evaluations are needed to solve (2)
- Computing the trace with automatic differentiation (AD) requires vector-Jacobian products with all d standard basis vectors, costing $\mathcal{O}(d^2)$ FLOPs total

Our Contributions

Optimal Transport (OT) Incorporating OT, we regularize the CNF so it has a unique solution (Figure 1).

Analytic Exact Trace We derive formulae for an exact trace computation with complexity $\mathcal{O}(d)$ FLOPs.

Optimal Transport

L_2 Transport Costs Add transport costs

$$L(\boldsymbol{x},T) = \int_0^T \frac{1}{2}\|\mathbf{v}(\boldsymbol{z}(\boldsymbol{x},t),t)\|^2 \,\mathrm{d}t.$$

to (3) to penalize the arc-length of the trajectories.

Potential Function By the Pontryagin maximum principle [5], there exists a scalar potential function $\Phi\colon \mathbb{R}^d \times [0,T] \to \mathbb{R}$ such that

$$\mathbf{v}(\boldsymbol{x},t;\boldsymbol{\theta}) = -\nabla\Phi(\boldsymbol{x},t;\boldsymbol{\theta}).$$

Idea: Analogous to classical physics, samples move to minimize their potential
$\Rightarrow$ We parameterize potential Φ instead of $\mathbf{v}$.

HJB Regularizer At optimality, Φ satisfies the Hamilton-Jacobi-Bellman (HJB) equation [6]

$$-\partial_t\Phi(\boldsymbol{x},t) = -\frac{1}{2}\|\nabla\Phi(\boldsymbol{z}(\boldsymbol{x},t),t)\|^2,$$

$$\Phi(\boldsymbol{x},T) = 1 + \log\left(\rho_0(\boldsymbol{x})\right) - \log\left(\rho_1(\boldsymbol{z}(\boldsymbol{x},T))\right) - \ell(\boldsymbol{x},T)$$

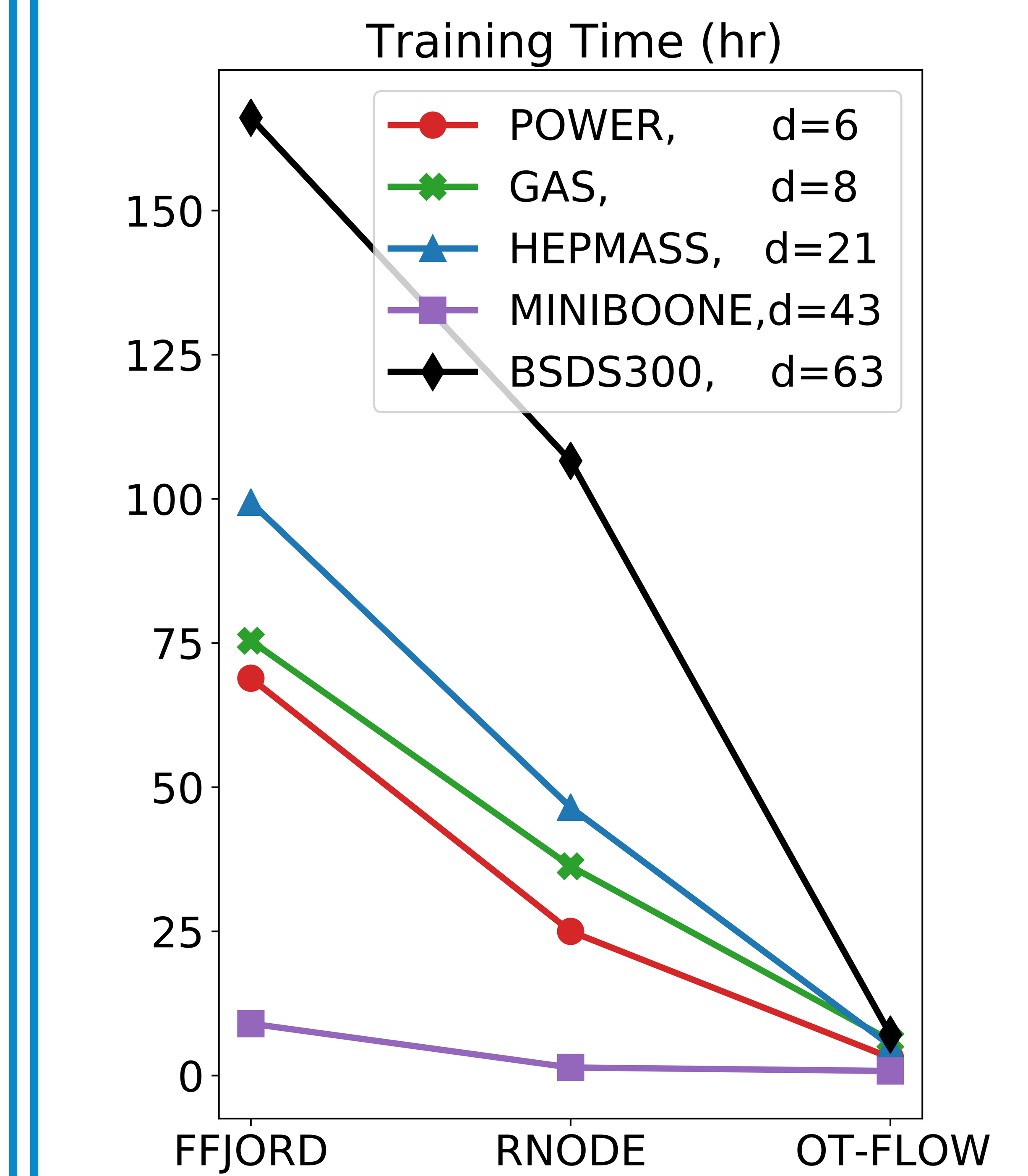
To penalize sub-optimality, use HJB regularizer

$$R(\boldsymbol{x},T) = \int_0^T \left| \partial_t\Phi(\boldsymbol{z}(\boldsymbol{x},t),t) - \frac{1}{2}\|\nabla\Phi(\boldsymbol{z}(\boldsymbol{x},t),t)\|^2 \right| \,\mathrm{d}t$$

OT-Flow Optimization Problem

$$\min_{\boldsymbol{\theta}} \; \mathbb{E}_{\rho_0(\boldsymbol{x})} \left\{ C(\boldsymbol{x},T) + L(\boldsymbol{x},T) + R(\boldsymbol{x},T) \right\} \text{ s.t. } (2)$$

Results



8x speedup in training and 24x speedup in inference relative to the state-of-the-art RNODE [4] on five real datasets of dimensionality $d = 6, 8, 21, 43, 63$

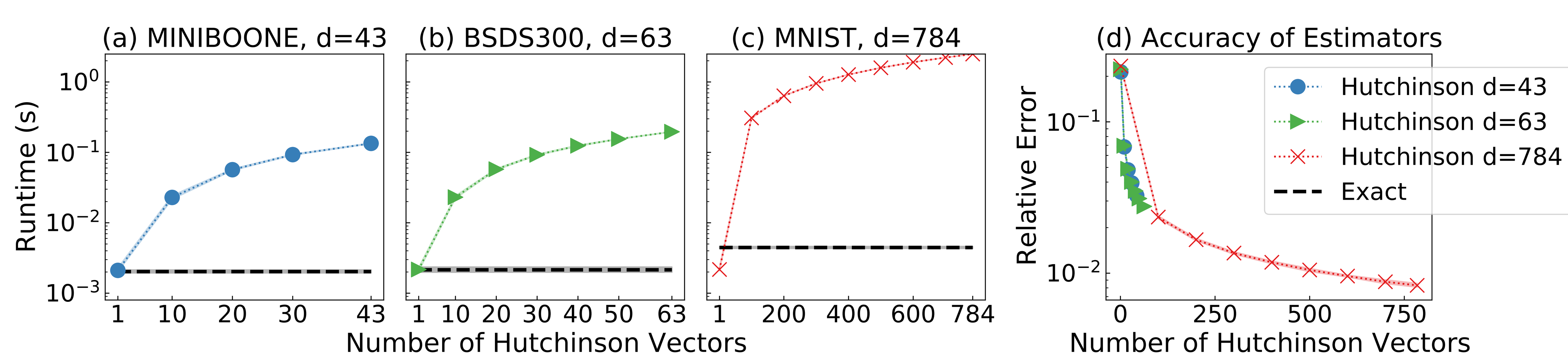
Exact Trace

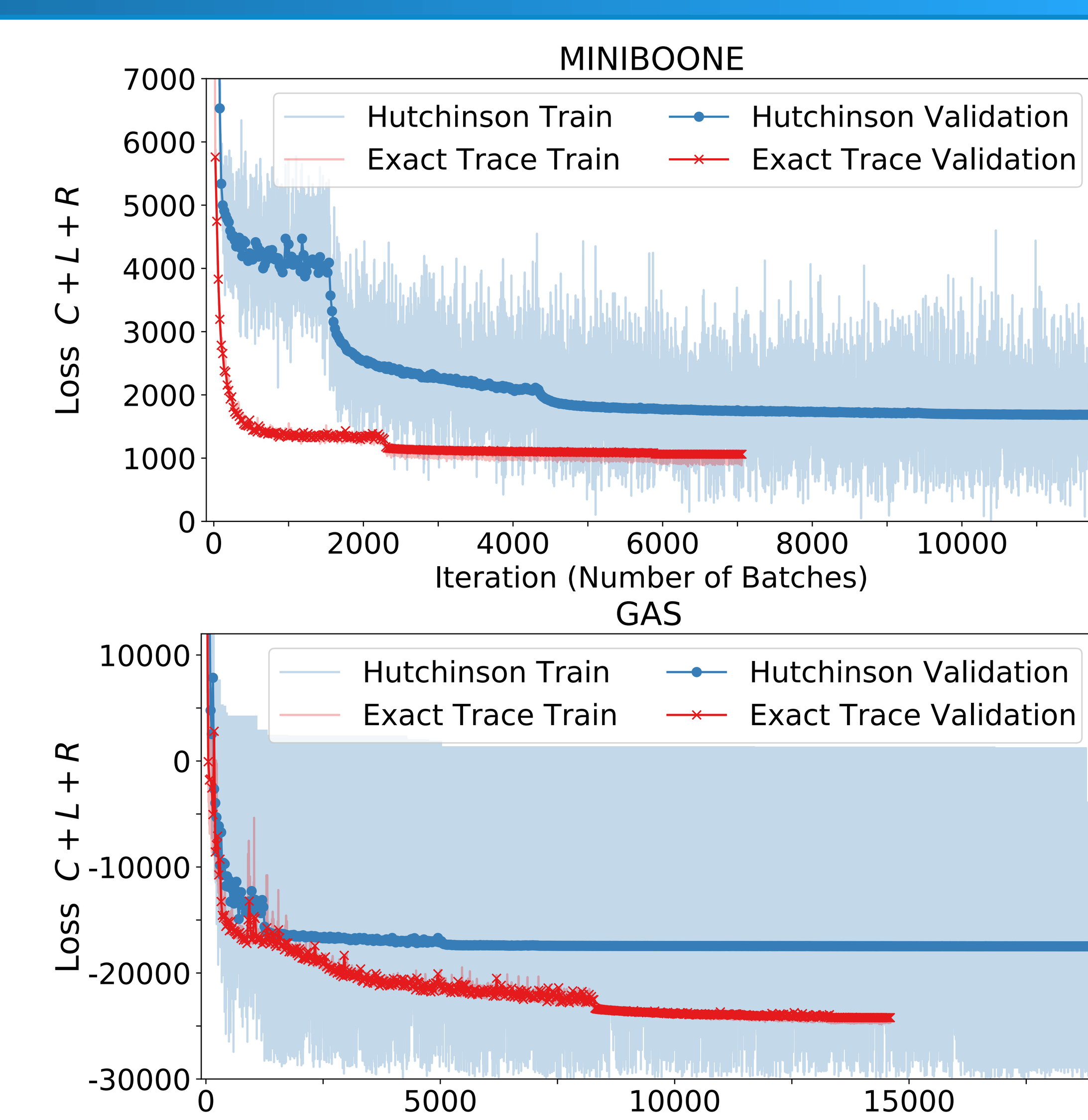
We devise an efficient analytic method for computing the trace by exploiting

$$\operatorname{tr}\left(\nabla^2\Phi(\boldsymbol{s};\boldsymbol{\theta})\right) = \operatorname{tr}\left(\boldsymbol{E}^\top \nabla^2_{\boldsymbol{s}}\Phi(\boldsymbol{s};\boldsymbol{\theta})\,\boldsymbol{E}\right) \quad \text{for} \quad \boldsymbol{E} = \texttt{eye(d+1,d)}.$$

In runtime, the exact trace is competitive with estimators used in other CNFs.

In convergence, the exact trace during training achieves 1) quicker validation convergence than using an estimator and 2) less variance in training loss than using an estimator





References

[1] D Rezende and S Mohamed *Variational Inference with Normalizing Flows.* ICML, 2015.

[2] Chen et al. *Neural Ordinary Differential Equations.* NeurIPS, 2018.

[3] W Grathwohl et al. *FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models.* ICLR, 2019.

[4] C Finlay et al. *How to Train your Neural ODE: the World of Jacobian and Kinetic Regularization.* ICML, 2020.

[5] L Evans *An Introduction to Mathematical Optimal Control Theory Version 0.2.* 2013.

[6] L Evans *Partial differential equations and Monge-Kantorovich mass transfer.* Current Developments in Mathematics, 1997.

Acknowledgments

National Science Foundation award DMS 1751636, Binational Science Foundation Grant 2018209, AFOSR Grants 20RT0237 and FA9550-18-1-0167, AFOSR MURI FA9550-18-1-0502, ONR Grant No. N00014-18-1-2527, a gift from UnitedHealth Group R&D, and a GPU donation by NVIDIA Corporation.

Special Thanks: IPAM Long Program MLP 2019 organizers & staff

Code

github.com/EmoryMLIP/OT-Flow