Rare Events: Analysis, Numerics, and Applications
Brin Mathematics Center, UMD, Feb 27 - March 3, 2023

Courant NYU

Generative Models, Normalizing Flows, and Monte Carlo Samplers

Eric Vanden-Eijnden

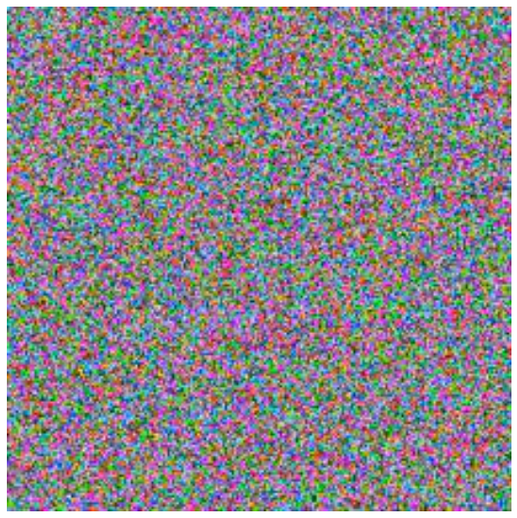
with Michael Albergo, Marylou Gabrié, and Grant Rotskoff

Density Estimation with Transport Maps

Aim: estimate the unknown ***probability density function*** $\rho_* \in \mathscr{D}(\Omega)$ from sample data $\{x_i\}_{i=1}^n$

- Take a simple ***base density*** $\rho_b \in \mathscr{D}(\Omega)$ (e.g. Gaussian) and;
- Build a (reversible) map $T : \Omega \to \Omega$ such that the ***pushforward of*** ρ_b ***by*** T ***is*** ρ_***:** $\quad T\sharp\rho_b = \rho_*$

Well-suited for ***generative modeling*** *and* ***sampling:*** if $x_b \sim \rho_b$ then $x = T(x_b) \sim \rho_*$



$\overset{T}{\longmapsto}$

Song *et al.*, ICLR (2021)

…. DALL-E (Open AI)

Allows for ***likelihood estimation****, etc. :* $\quad \rho_*(x) = \rho_b(T^{-1}(x)) \det[\nabla T^{-1}(x)]$

Density Estimation with Transport Maps

- Take a simple ***base density*** $\rho_b \in \mathscr{D}(\Omega)$ (e.g. Gaussian) and;
- Build a (reversible) map $T : \Omega \to \Omega$ such that the ***pushforward of*** ρ_b ***by*** T ***is*** ρ_***:** $\quad T\sharp\rho_b = \rho_*$

Link with transportation theory (without the need for optimality) - Monge, Ampère, Kantorovich, Brenier, Villani, …

How to estimate the map T ***in a computationally tractable way?***

Build T as the *composition of simpler maps* estimated sequentially via *max entropy method.*

Chen & Gopinath, NeurIPS 13 (2000);
Tabak & V.-E., Commun. Math. Sci. 8: 217-233 (2010);
Tabak & Turner, Comm. Pure App. Math LXVI, 145-164 (2013).

Approximate T by a *neural net* and use *invertible neural architectures* giving T^{-1}.

Dinh *et al.* arXiv:1410.8516 (2014);
Rezende *et al.*, arXiv:1505.05770 (2015);
Papamakarios *et al.* arXiv:1912.02762 (2019); …

NICE: Dinh *et al.* arXiv:1410.8516 (2014);
Real NVP: Dinh *et al.* arXiv:1605.08803 (2016)

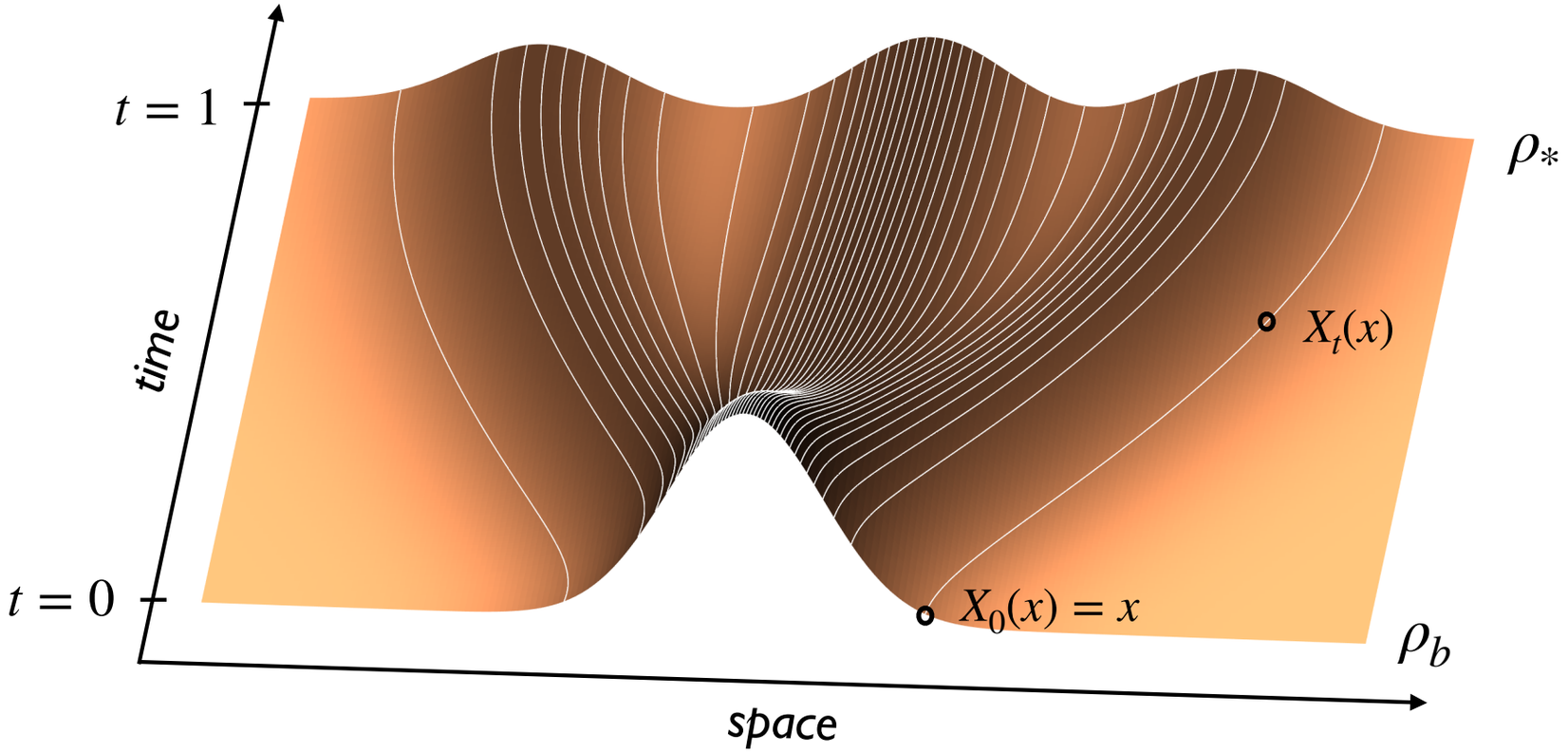
View T as solution of a *continuous-time flow* with a velocity approximated by a neural net.

FFJORD: Grathwohl *et al.* arXiv:1810.01367 (2018)

Continuous Time Flow

Set $T = X_{t=1}$ where X_t = **flow map** associated with a **time-dependent velocity field** $v_t(x)$:

$$\frac{d}{dt}X_t(x) = v_t(X_t(x)) \qquad X_0(x) = x$$



Continuous Time Flow

Set $T = X_{t=1}$ where X_t = **flow map** associated with a **time-dependent velocity field** $v_t(x)$:

$$\frac{d}{dt}X_t(x) = v_t(X_t(x)) \qquad X_0(x) = x$$

Lagrangian frame

Equivalently:

If $\rho_t(x)$ solves $\partial_t \rho_t + \nabla \cdot (v_t \rho_t) = 0, \quad \rho_{t=0} = \rho_b$ then $\rho_{t=1} = \rho_* =$ target PDF

Eulerian frame

- Solution by **method of characteristics**: given initial $\rho_{t=0}(x) = \rho_b(x)$, we have

$$\rho_t(X_t(x)) = \rho_b(x)\exp\left(-\int_0^t \nabla \cdot v_s(X_s(x))ds \right) \qquad \forall t$$

Pointwise evaluation of $\rho_t(x)$;
Calculation of cross-entropy; …

- **Benamou-Brenier theory** guarantees that $v_t(x)$ exists such that $\rho_{t=1}(x) = \rho_*(x)$

How do we get the right $v_t(x)$?

Maximum Entropy Formulation

Basic idea

- *Use the Kullback-Leibler divergence of ρ_* to $\rho_{t=1}$ as objective;*
- *Notice that unknown $\int_\Omega \log \rho_*(x)\rho_*(x)dx$ is a constant that plays no role.*

Proposition: Consider the optimization problem

$$\min_v \int_\Omega \log\left(\frac{\rho_*(x)}{\rho_{t=1}(x)}\right)\rho_*(x)dx = -\max_v \int_\Omega \log \rho_{t=1}(x)\rho_*(x)dx + C$$

subject to:
$$\partial_t \rho_t = -\nabla\cdot\left(v_t\rho_t\right), \quad \rho_{t=0} = \rho_b$$

Then all optimizers $v_t(x)$ are such that $\rho_{t=1} = \rho_*$.

Eulerian $\Rightarrow$ Lagrangian

Maximum Entropy Formulation

Proposition: Consider the minimization problem

$$\min_{v} \int_{\Omega} \left[\int_{0}^{1} \nabla \cdot v_t(\bar{X}_t(x))dt - \log \rho_b(\bar{X}_{t=0}(x)) \right] \rho_*(x)dx$$

subject to:

$$\frac{d}{dt}\bar{X}_t(x) = v_t(\bar{X}_t(x)), \qquad \bar{X}_{t=1} = x$$

Then all minimizers $v_t(x)$ are such that $\bar{X}_{t=1}^{-1} \sharp \rho_b = \rho_*$ i.e. $x_b \sim \rho_b \;\Rightarrow\; \bar{X}_{t=1}^{-1}(x_b) \sim \rho_*$.

Tractable in principle:

- Objective and its gradient can be evaluated empirically using samples from ρ_* ;
- Velocity $v_t(x)$ can be approximated by deep neural network (DNN);
- Constrained optimization can be performed by SGD + adjoint method (= neural ODE framework)

FFJORD: Grathwohl *et al.* arXiv:1810.01367 (2018)

Maximum Entropy Formulation

Proposition: Consider the minimization problem

$$\min_{v} \int_{\Omega} \left[\int_{0}^{1} \nabla \cdot v_t(\bar{X}_t(x))dt - \log \rho_b(\bar{X}_{t=0}(x)) \right] \rho_*(x)dx$$

subject to:

$$\frac{d}{dt}\bar{X}_t(x) = v_t(\bar{X}_t(x)), \qquad \bar{X}_{t=1} = x$$

Then all minimizers $v_t(x)$ are such that $\bar{X}_{t=1}^{-1} \sharp \rho_b = \rho_*$ i.e. $x_b \sim \rho_b \Rightarrow \bar{X}_{t=1}^{-1}(x_b) \sim \rho_*$.

Training is costly as it requires many passes through ODE solver.

Optimization is only weakly constrained — many $v_t(x)$ do the job, most are unnecessarily complicated.

Can we separate the task of building a path from ρ_b to ρ_* from that of learning $v_t(x)$?

Score-Based Diffusion Models

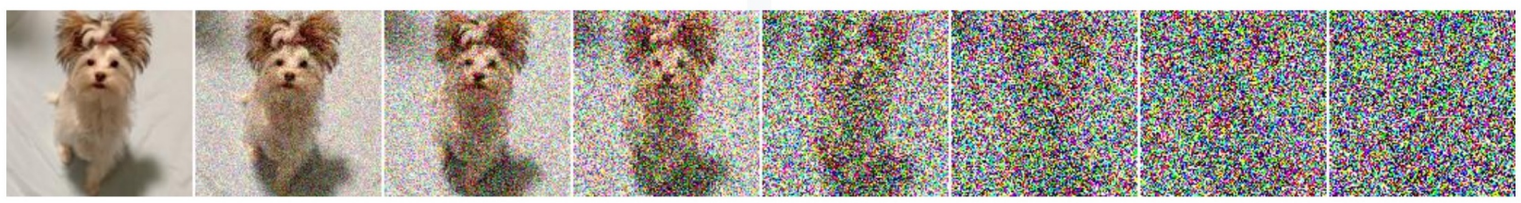
Song et al. arXiv:2011.13456 (2021);
*Hyvärinen JMLR **6** (2005);*
*Vincent, Neural Comp. **23**, 1661 (2011)*

Given data from the target ρ_*:

- Devolve it into the Gaussian base ρ_b using e.g. an Ornstein-Ulhenbeck process;
- Time-reverse the SDE to generate new samples from ρ_* from samples from ρ_b;

Forward SDE (data → noise)

$\mathbf{x}(0)$ — $\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$ → $\mathbf{x}(T)$



score function

$\mathbf{x}(0)$ ← $\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$ — $\mathbf{x}(T)$

Reverse SDE (noise → data)

From Song's blog on SBDM

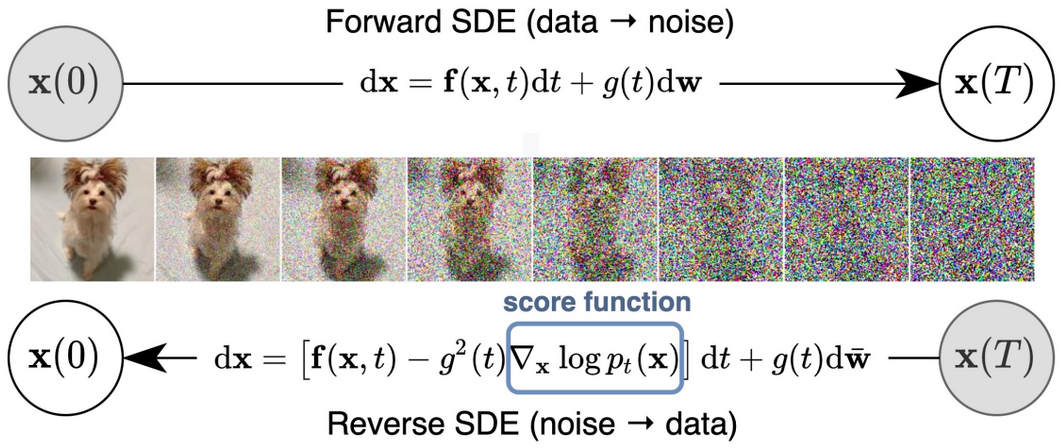
Builds a connection = path in density space between ρ_b and ρ

Score-Based Diffusion Models

Song et al. arXiv:2011.13456 (2021);
*Hyvärinen JMLR **6** (2005);*
*Vincent, Neural Comp. **23**, 1661 (2011)*

- Data from $\rho_t(x)$ easy to generate:
  add noise to data from $\rho_*(x)$.

- Reverse SDE needs the Fischer score $\nabla \log \rho_t(x)$



- Learn score via minimization of Fisher divergence: Given $\rho_t(x)$, we have:

$$
\begin{aligned}
s_t(x) &= \underset{s(x)}{\operatorname{argmin}} \int_\Omega |\, s(x) - \nabla \log \rho_t(x) \,|^2 \rho_t(x) dx \\
&= \underset{s(x)}{\operatorname{argmin}} \int_\Omega \Big(|\, s(x) \,|^2 + 2\nabla \cdot s(x) \Big) \rho_t(x) dx
\end{aligned}
$$

Tractable in practice:

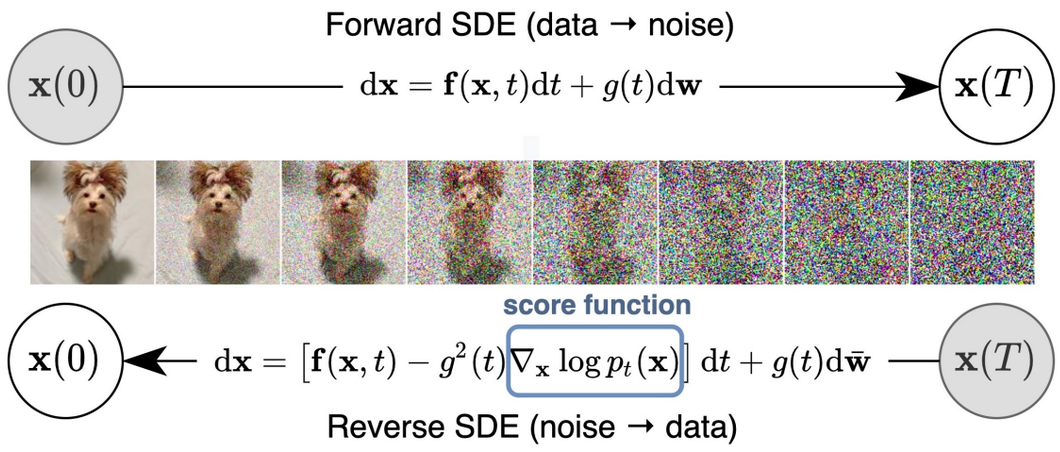
- Objective and its gradient can be evaluated empirically by sampling ρ_t;
- Score $s_t(x)$ can be approximated by deep neural network (DNN);
- Minimization can be performed by **direct SGD** *(no adjoint needed)*.

Score-Based Diffusion Models

Song et al. arXiv:2011.13456 (2021);
*Hyvärinen JMLR **6** (2005);*
*Vincent, Neural Comp. **23**, 1661 (2011)*

- Data from $\rho_t(x)$ easy to generate:
  add noise to data from $\rho(x)$.

- Reverse SDE needs the Fischer score $\nabla \log \rho_t(x)$



Requires taking $T \gg 1$ *and limits choice of base density* ρ_b *since* $\lim_{t\to\infty} \rho_t = \rho_b = N(0,1)$.

Gives reversed SDE and probability flow ODE —the latter is ***needed for likelihood calculation.***

Can we avoid the SDE, work on $t \in [0,1]$ *with arbitrary* ρ_b *and* ρ_**, build a connection between them, and get the velocity* $v_t(x)$ *directly?*

Building Flows with Stochastic Interpolants

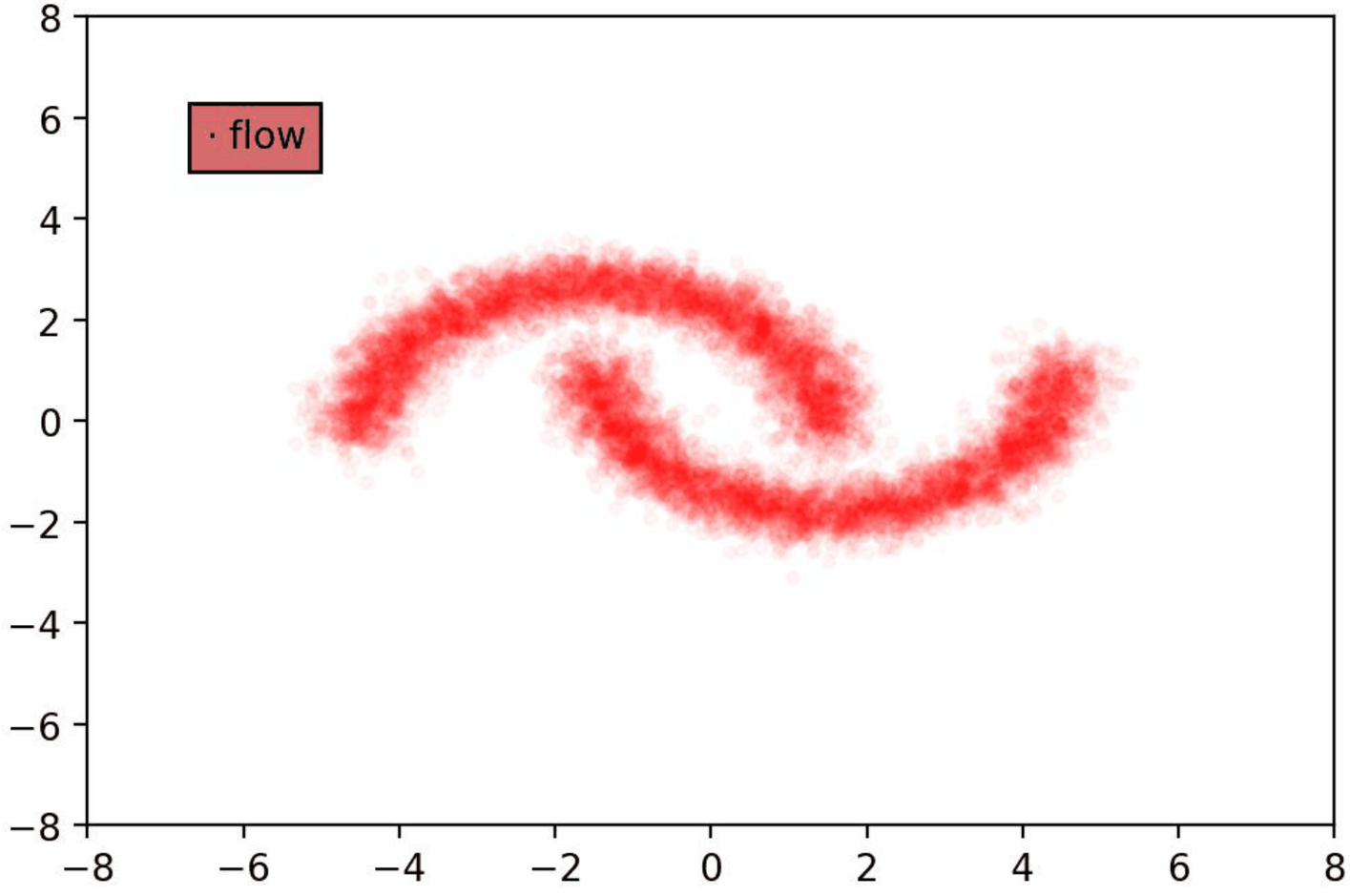
with Michael Albergo

Define the ***interpolant density*** ρ_t as the PDF of the ***stochastic interpolant:***

$$x_t = I_t(x_b, x_*) \quad \text{with} \quad x_b \sim \rho_b, \quad x_* \sim \rho_*,$$

where $I_t(x_b, x_*)$ is differentiable and satisfies $I_{t=0}(x_b, x_*) = x_b, \quad I_{t=1}(x_b, x_*) = x_*.$

For example: $$x_t = \cos\left(\tfrac{1}{2}\pi t\right) x_b + \sin\left(\tfrac{1}{2}\pi t\right) x_*$$



*Builds a path ρ_t between **any** ρ_b and ρ_* that is easy to sample.*

Building Flows with Stochastic Interpolants

with Michael Albergo

S*tochastic interpolant* : $x_t = I_t(x_b, x_*)$ with $x_b \sim \rho_b$, $x_* \sim \rho_*$, and $I_{t=0}(x_b, x_*) = x_b, \quad I_{t=1}(x_b, x_*) = x_*.$

Proposition: We have

$$\partial_t \rho_t + \nabla \cdot j_t = 0, \qquad \rho_{t=0} = \rho_b, \quad \rho_{t=1} = \rho_*$$

with the current $j_t(x)$ defined by: for all test functions $\phi : \Omega \to \mathbb{R}$

$$\int_\Omega \nabla \phi(x) \cdot j_t(x) dx = \int_{\Omega \times \Omega} \partial_t I_t(x_b, x_*) \cdot \nabla \phi(I_t(x_0, x_1)) \rho_b(x_b) \rho_*(x_*) dx_b dx_*$$

Consequence of chain rule: $\rho_t(x) = \mathbb{E}_{\rho_b, \rho_*}[\delta(x - I_t)]$ $j_t(x) = \mathbb{E}_{\rho_b, \rho_*}[\partial_t I_t \, \delta(x - I_t)]$. .

Define $v_t(x) = \nabla U_t(x)$ where the time-dependent potential $U_t(x)$ solves the *Poisson equation*

$$\nabla \cdot \left(\rho_t \nabla U_t \right) = \nabla \cdot j_t = - \partial_t \rho_t$$

Use variational formulation of this equation to get a tractable objective

Building Flows with Stochastic Interpolants

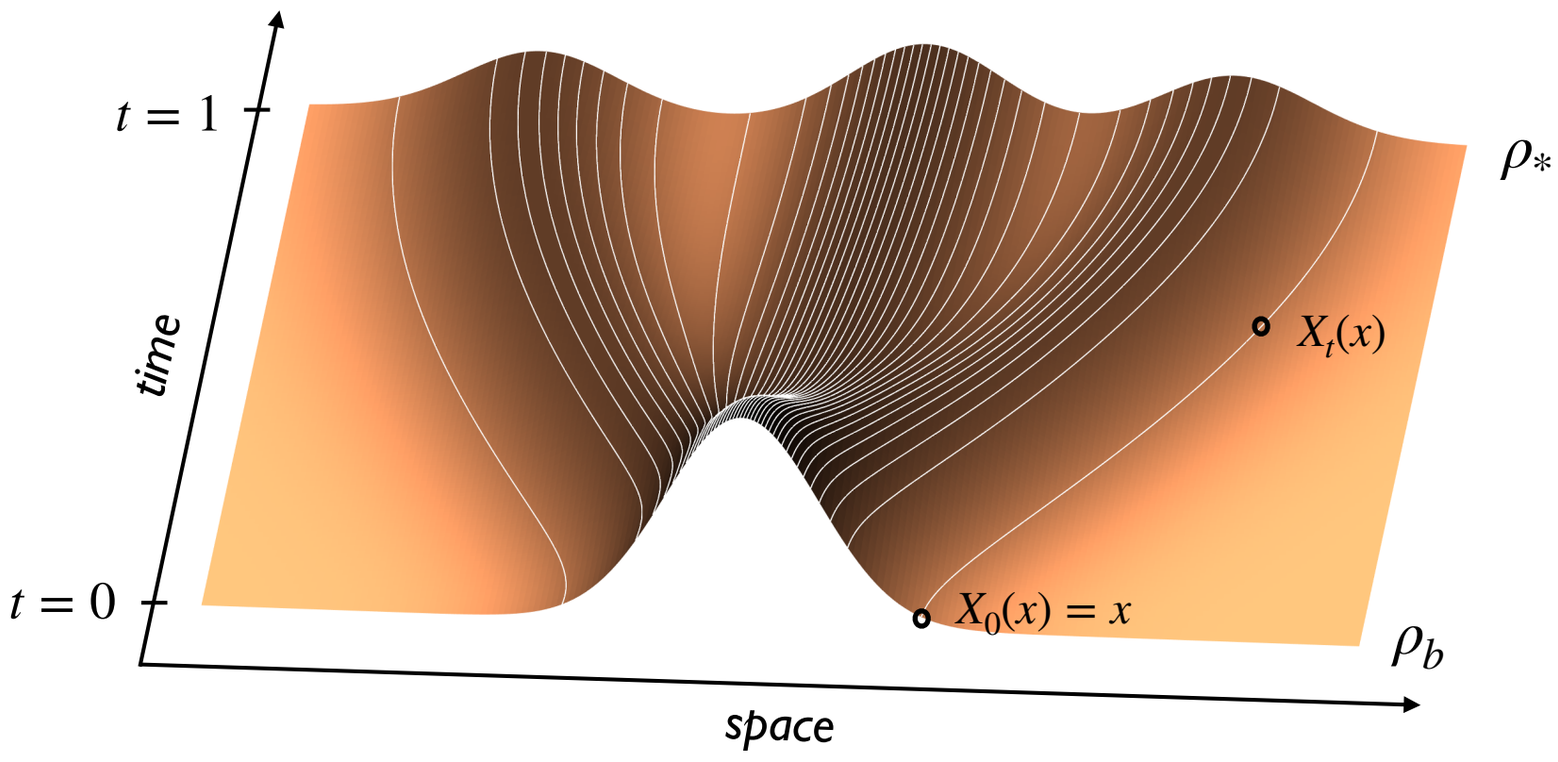
with Michael Albergo

Proposition: The PDF $\rho_t(x)$ of x_t satisfies

$$\partial_t \rho_t + \nabla \cdot (v_t \rho_t) = 0, \qquad \rho_{t=0} = \rho_b, \quad \rho_{t=1} = \rho_*$$

with a velocity $v_t(x) = \nabla U_t(x)$ with $U_t(x)$ the unique minimizer of

$$\mathbb{E}_{\substack{t \sim U(0,1) \\ x_b \sim \rho_b \\ x_* \sim \rho_*}} \left(|\nabla U_t(I_t(x_b, x_*))|^2 - 2\partial_t I_t(x_b, x_*) \cdot \nabla U_t(I_t(x_b, x_*)) \right)$$



Albergo & V.-E. arXiv:2209.15571 (2022);
Liu et al. arXiv:2209.03003 (2022);
Lipman et al. arXiv:2210.02747 (2022)

Building Flows with Stochastic Interpolants

with Michael Albergo

Proposition: The PDF $\rho_t(x)$ of x_t satisfies

$$\partial_t \rho_t + \nabla \cdot (v_t \rho_t) = 0, \qquad \rho_{t=0} = \rho_b, \quad \rho_{t=1} = \rho_*$$

with a velocity $v_t(x) = \nabla U_t(x)$ with $U_t(x)$ the unique minimizer of

$$\mathbb{E}_{\substack{t \sim U(0,1) \\ x_b \sim \rho_b \\ x_* \sim \rho_*}} \left(|\nabla U_t(I_t(x_b, x_*))|^2 - 2\partial_t I_t(x_b, x_*) \cdot \nabla U_t(I_t(x_b, x_*)) \right)$$

Tractable in practice:

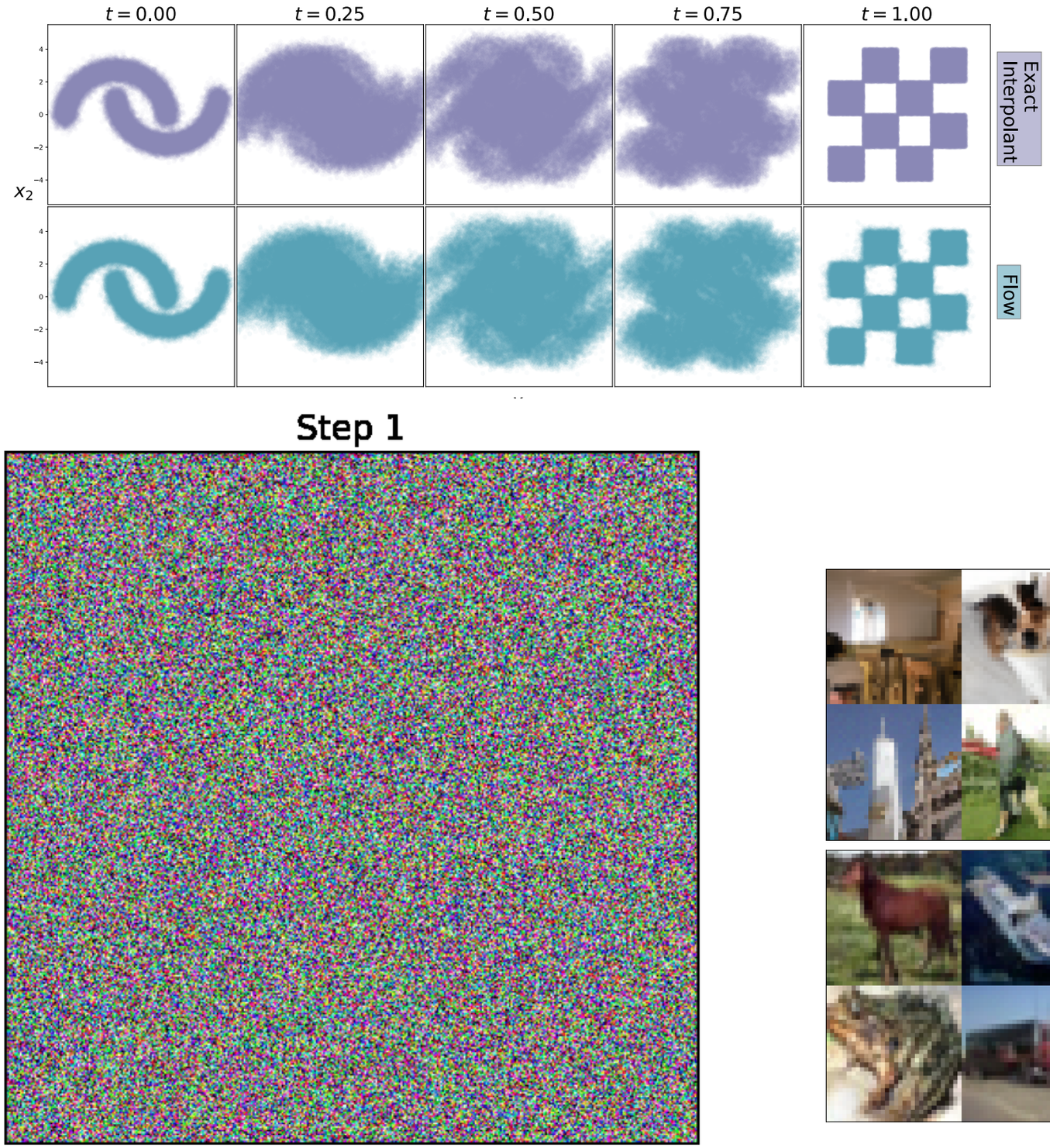
- Objective and its gradient can be evaluated empirically by sampling ρ_b and ρ_*;
- Potential $U_t(x)$ (or velocity $v_t(x) = \nabla U_t(x)$) can be approximated by DNN;
- Minimization can be performed by direct SGD;

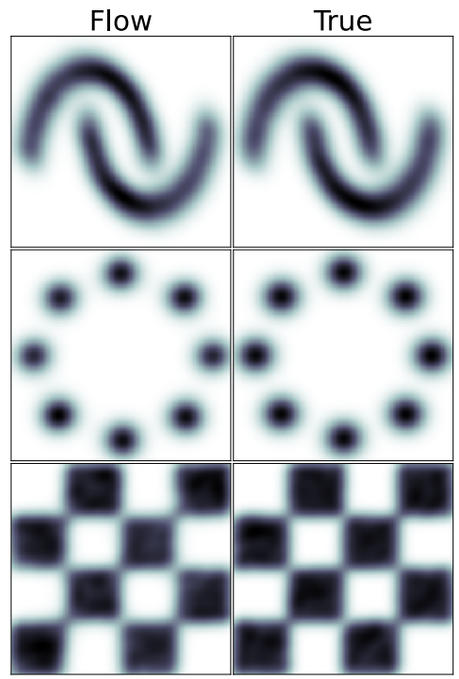
- Loss controls the Wasserstein 2 distance between $\rho_{t=1}$ and ρ_*
  — constant involves Lipschitz constant of estimated $v_t(x)$ (generalization to control the KL)

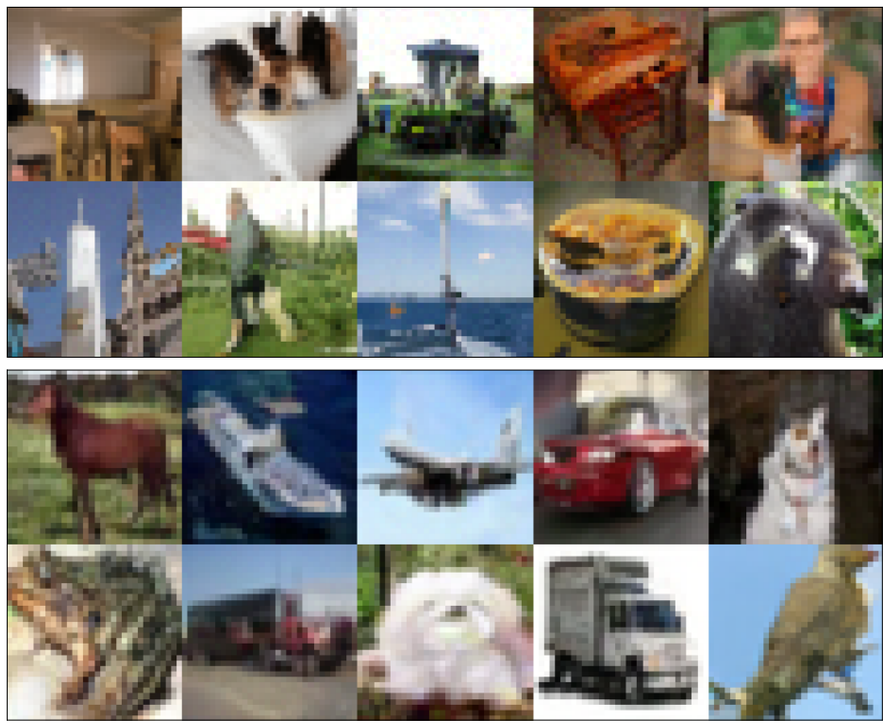
- *Optional:* Maximizing the objective over the interpolant $I_t(x_b, x_*)$ gives optimal transport plan.

Building Flows with Stochastic Interpolants

with Michael Albergo







Flower 129x128
ImageNet 32x32
CIFAR-10

Building Flows with Stochastic Interpolants

with Michael Albergo

	POWER	GAS	HEPMASS	MINI-BOONE	BSDS300
MADE	3.08	-3.56	20.98	15.59	-148.85
Real NVP	-0.17	-8.33	18.71	13.55	-153.28
Glow	-0.17	-8.15	18.92	11.35	-155.07
CPF	-0.52	-10.36	16.93	10.58	-154.99
NSP	-0.64	-13.09	14.75	9.67	-157.54
FFJORD	-0.46	-8.59	14.92	10.43	-157.40
OT-Flow	-0.30	-9.20	17.32	10.55	-154.20
Ours	-0.57	-12.35	14.85	10.42	-156.22

Method	CIFAR-10		ImageNet-32x32	
	NLL	**FID**	**NLL**	**FID**
FFJORD	3.40			
Glow	3.35		4.09	
DDPM	$\leq$ 3.75	3.17		
DDPM++	$\leq$ 3.37	2.90		
ScoreSDE	2.99	3.17		
VDM	$\leq$ 2.65	7.41	$\leq$3.72	
Soft Truncation	2.88	3.45	3.85	8.42
ScoreFlow	2.81	5.40	3.76	10.18
Ours	2.99	10.07	3.45	8.44

Table 2: *Left:* Negative log likelihoods (NLL) computed on test data unseen during training (lower is better). Values of MADE, Real NVP, and Glow quoted from the FFJORD paper. Values of OT-Flow, CPF, and NSP quoted from their respective publications. *Right:* NLL and FID scores on unconditional image generation tasks for recent advanced models that emit a likelihood.

What if we have no prior data from the target ρ_* but some structural info about it ?

$\Rightarrow$ Monte-Carlo sampling

Monte-Carlo Sampling

Fermi, Ulam, Metropolis, Rosenbluth, …

Given the probability density $\rho_* \in \mathscr{P}(\Omega)$ only ***known up to a normalization factor***, i.e.

$$\rho_*(x) = Z_*^{-1} e^{-U_*(x)}$$

with $U_* : \Omega \to \mathbb{R}_+$ given, but $Z_* = \int_\Omega e^{-U_*(x)} dx < \infty$ unknown:

Compute Z_* and/or ***expectation of the observable*** $f : \Omega \to \mathbb{R}$

$$\mathbb{E}_* f := \int_\Omega f(x) \rho_*(x) dx$$

- *Generic problem in* **Statistical Mechanics**, **Bayesian Inference**, **Uncertainty Quantification**, *etc.*
- *Analytical evaluation intractable, standard numerical quadrature methods inapplicable.*

⇒ *use* **Monte-Carlo sampling** *(i.e. approximate expectation by empirical average)*

Monte-Carlo Sampling

Fermi, Ulam, Metropolis, Rosenbluth, …

Two main approaches (since sampling directly from the target ρ_* is hard):

- ***Importance sampling:*** Generate data $\{x_i\}_{i\in\mathbb{N}}$ from *simpler density* $\rho_b(x) = Z_b^{-1} e^{-U_b(x)}$ and use

$$\mathbb{E}_* f = \frac{\mathbb{E}_b(fw)}{\mathbb{E}_b(w)} = \lim_{n\to\infty} \frac{\sum_{i=1}^n f(x_i) w(x_i)}{\sum_{i=1}^n w(x_i)} \quad \text{with} \quad w(x) = e^{-U(x)+U_b(x)}$$

- ***Markov chain MC:*** Generate *Markov sequence* $\{x_i\}_{i\in\mathbb{N}}$ with kernel $P^x(A) = \mathbb{P}(x_{i+1} \in A \,|\, x_i = x)$ such that

$$\mathbb{E}_*(f) = \lim_{n\to\infty} S_n(f) \quad \text{with} \quad S_n(f) = \frac{1}{n}\sum_{i=1}^n f(x_i)$$

Monte-Carlo Sampling

Fermi, Ulam, Metropolis, Rosenbluth, …

Main difficulties:

- ***Importance sampling:*** independent samples **but** beware of high variance of the weights:

$$\mathbb{E}_b(w^2) = \infty \quad \text{in general}$$

Agapiou *et al.* arXiv:1511.06196 (2017)

- ***Markov chain MC:*** no weights **but** beware of slow time-decorrelation*:*

$$\mathbb{E}\,|\,S_n(f) - \mathbb{E}_*(f)\,|^2 \sim \frac{1}{n}\mu\left(u(1-P)u\right) \geq \frac{1}{n}\mathrm{var}(f) \quad \text{with} \quad u - Pu = f - \mathbb{E}_*(f)$$

$$\approx \frac{\tau}{n}\mathrm{var}(f) \quad \text{with} \quad \tau = \text{decorrelation time} \gg 1$$ Kipnis & Varadhan CLT

Efficiency requires to tailor the base distribution ρ_b or the kernel $P^x(dy)$ to the target ρ_*.

Variational Formulations

Basic idea:

- *Use the Kullback-Leibler divergence of $\rho_{t=1} = X_{t=1}\#\rho_b$ from the target $\rho_* = Z^{-1}e^{-U_*}$ as objective;*
- *Notice that unknown Z is a constant that play no role.*

Proposition: Given $\rho_* = Z^{-1}e^{-U_*}$ and ρ_b consider the minimization problem

$$\min \int_\Omega \log\left(\frac{\rho_{t=1}(x)}{\rho_*(x)}\right)\rho_{t=1}(x)dx = \min \int_\Omega [U_*(x) + \log \rho_{t=1}(x)]\rho_{t=1}(x)dx + \log Z_*$$

subject to: $$\partial_t \rho_t = -\nabla \cdot (v_t \rho_t), \quad \rho_{t=0} = \rho_b$$

Then all minimizers satisfy $\rho_{t=1} = \rho_*$.

Eulerian $\Rightarrow$ Lagrangian

Variational Formulations

Proposition: Given $\rho_* = Z^{-1} e^{-U_*}$ and ρ_b consider the minimization problem

$$\min \int_\Omega \left[U_*(X_{t=1}(x)) - \int_0^1 \nabla \cdot v_t(X_t(x)) dt \right] \rho_b(x) dx$$

subject to:

$$\dot{X}_t(x) = v_t(X_t(x)), \qquad X_{t=0} = x$$

Then all minimizers satisfy $X_{t=1} \sharp \rho_b = \rho_*$ i.e. $x_b \sim \rho_b \;\Rightarrow\; X_{t=1}(x_b) \sim \rho_*$.

Tractable in principle:

- Objective and its gradient can be evaluated empirically by sampling ρ_b ;
- Velocity $v_t(x)$ can be approximated by deep neural network (DNN);
- Constrained optimization can be performed by SGD + adjoint method.

FFJORD: Grathwohl *et al.* arXiv:1810.01367 (2018)

Importance Sampling and Transport

In practice:

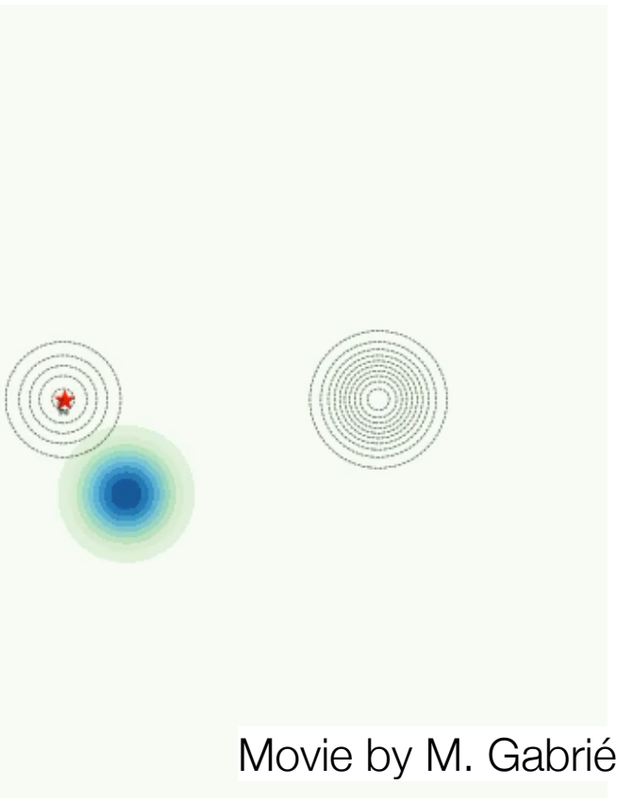
Rezende *et al.*, arXiv:1505.05770; ….
Noé *et al.*, Science 365 eaaw1147 (2019)

- *Use data from* ρ_b *to learn the velocity* $v_t(x)$.
- *Solve* $\dot{X}_t = v_t(X_t)$ *to push forward data* $x_b \sim \rho_b$ *onto* $X_{t=1}(x_b) \sim \rho_{t=1}$
- *Use the (imperfect) samples* $X_{t=1}(x_b) \sim \rho_{t=1}$ *to do IS, i.e. re-weight and use*

$$\mathbb{E}_*(f) = \frac{\mathbb{E}_b(f(X_{t=1})w_b)}{\mathbb{E}_b(w_b)} \quad \textit{with} \quad w_b(x) = e^{-U_*(X_{t=1}(x))+U_b(x)+\int_0^1 \nabla\cdot v_t(X_t(x))dt}$$

Main practical issues:

1. *Hard to train because of constraint*
   - requires adjoint method
2. *Limited capacity for exploration;*
3. *Prone to mode-collapse.*



Movie by M. Gabrié

Assisting MCMC Sampling with Normalizing Flows

with Marylou Gabrié & Grant Rotskoff

Key observation: Any imperfect map $X_{t=1} = T$ can be used to do Metropolis-Hastings MCMC:

- Given x_i, propose a new $\hat{x} = T(x_b)$ with $x_b \sim \rho_b$;
- Set $x_{i+1} = \hat{x}$ instead of keeping $x_{i+1} = x_i$ with probability

$$a(\hat{x}, x_i) = \min\left\{ \frac{\rho_*(\hat{x})\, T\sharp\rho_b(x_i)}{\rho_*(x_i)\, T\sharp\rho_b(\hat{x})}, 1 \right\}$$

Albergo, Kanwar, Shanahan, Phys. Rev. D **100**, 034515 (2019)

Generates a Markov sequence $\{x_i\}_{i\in\mathbb{N}}$ such that

$$\mathbb{E}_*(f) = \lim_{n\to\infty} S_n(f) \quad \text{with} \quad S_n(f) = \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

- no need to reweigh;
- independent samples if $T\sharp\rho_b = \rho_*$ (*perfect map — not needed*).

This strategy can be combined with a standard MH-MCMC (e.g. MALA) by alternating proposal moves.

Assisting MCMC Sampling with Normalizing Flows

with Marylou Gabrié & Grant Rotskoff

Key observation: Any imperfect map $X_{t=1} = T$ can be used to do Metropolis-Hastings MCMC:

- Given x_i, propose a new $\hat{x} = T(x_b)$ with $x_b \sim \rho_b$;
- Set $x_{i+1} = \hat{x}$ instead of keeping $x_{i+1} = x_i$ with probability

$$a(\hat{x}, x_i) = \min\left\{ \frac{\rho_*(\hat{x})\, T\sharp\rho_b(x_i)}{\rho_*(x_i)\, T\sharp\rho_b(\hat{x})}, 1 \right\}$$

In practice:

- *Perform MH-MCMC that alternates between:*
  - *local sampling (e.g. with MALA), and*
  - *resampling step by NF.*

Gabrié, Rotskoff & V.-E. arXiv:2105.12603 (2021)
Gabrié, Rotskoff & V.-E. arXiv:2107.08001 (2021)

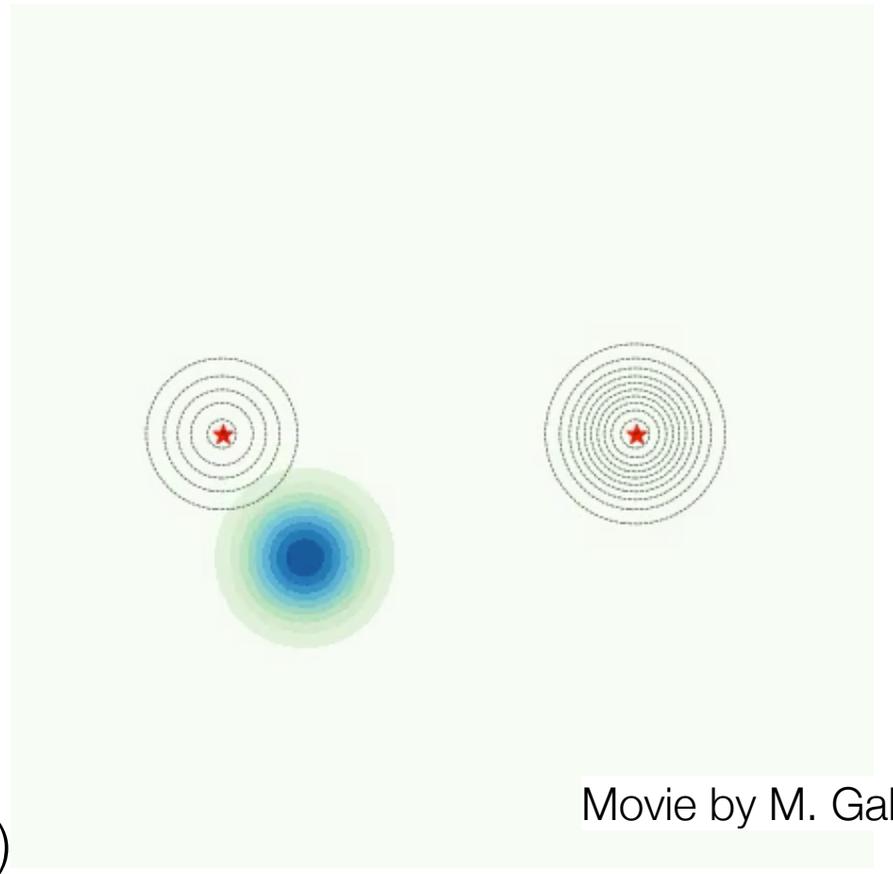
- *Use the generated data from ρ_* to train the flow using interpolant method.*

Assisting MCMC Sampling with Normalizing Flows

with Marylou Gabrié & Grant Rotskoff

Need rough location of modes to start sampling,
but not their relative weights.

Enable global moves
— no need to sample the transition state.



Movie by M. Gabrié

Nonlinear MCMC method (i.e. kernel depends on the law)

Andrieu *et al.* Bernoulli **17**(3), 987 (2011)

Convergence rate can be analyzed in some settings, in particular if:

- the trained map tracks perfectly the evolving distribution of the chain;

Gabrié, Rotskoff & V.-E. arXiv:2105.12603

- the training eventually stops (= *diminishing adaptivity*).

Brofos, Gabrié, Brubaker & Lederman arXiv:2110.13216

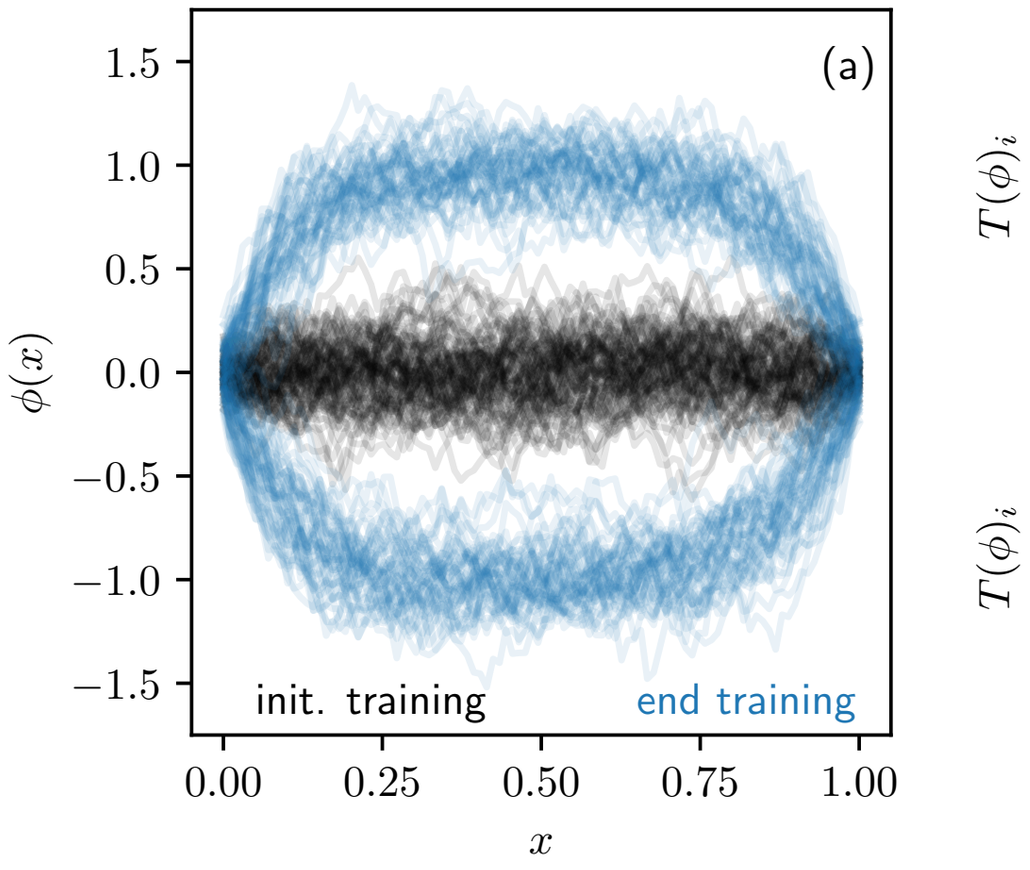
MCMC with NF for Sampling of Random Fields

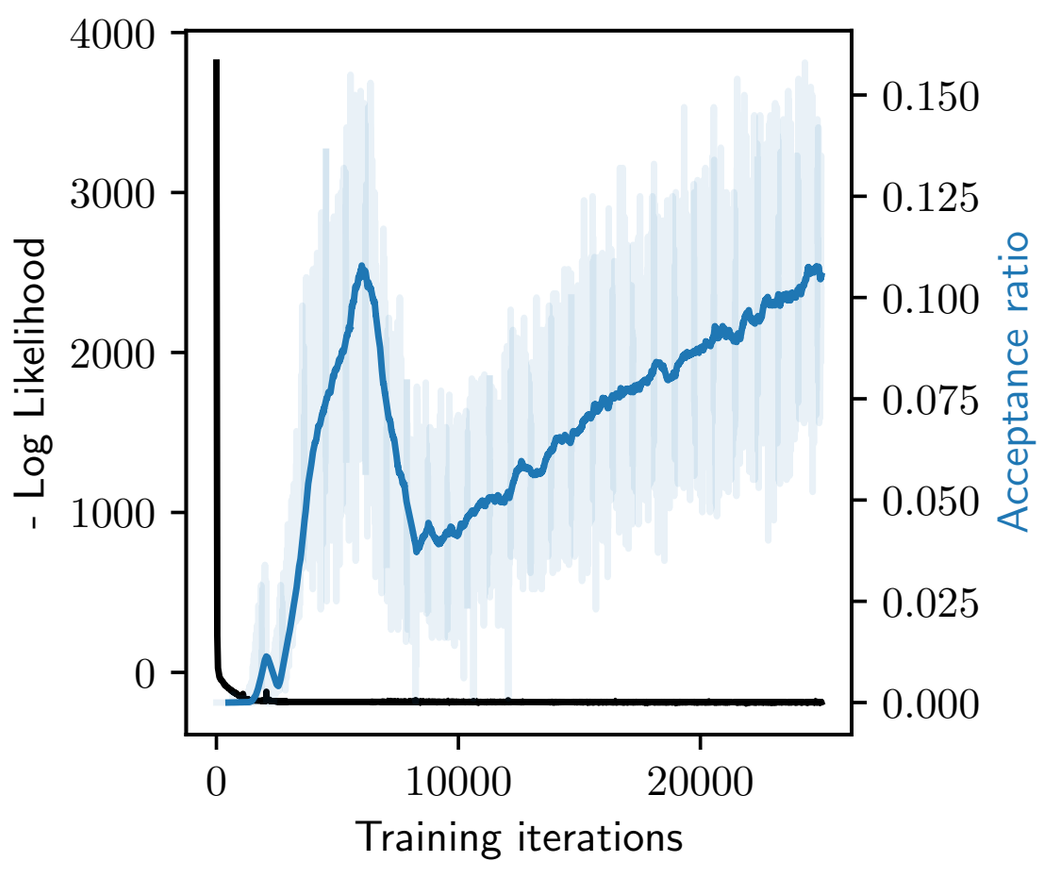
with Marylou Gabrié & Grant Rotskoff

- Target distribution is Gibbs measure associated with ϕ^4 energy:

$$E(\phi) = \int_0^1 \left(\frac{\alpha}{2} |\, \partial_x \phi \,|^2 + \frac{1}{4\alpha}(1-\phi^2)^2 \right) dx \qquad \text{subject to:} \quad \phi(0) = \phi(1) = 0 \quad (\alpha > 0)$$

- Base distribution = scaled Brownian bridge on $[0,1]$





MCMC with NF to Detect Phase Transition

with Marylou Gabrié & Grant Rotskoff

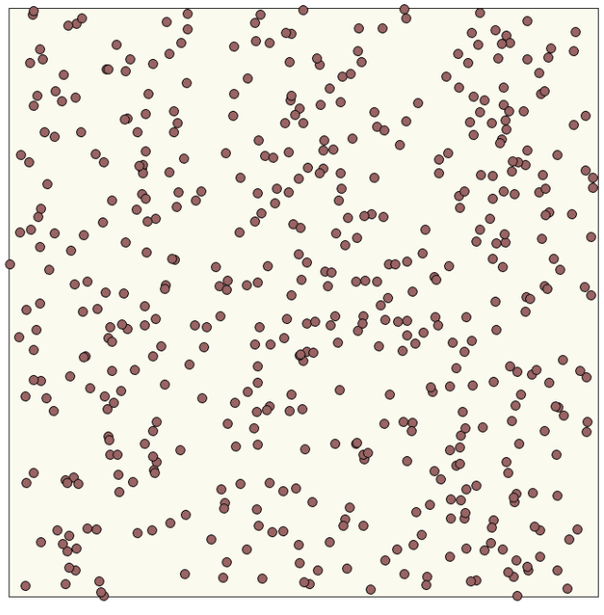
- System of particles in a box $B = [0,1]^2$ interacting via short-range attracting potential $W(x)$:

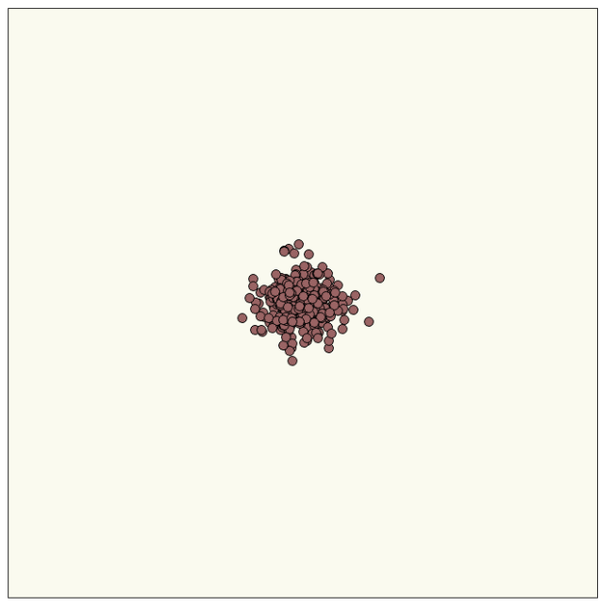
$$U(x_1, \ldots, x_N) = \frac{1}{2N} \sum_{i,j=1}^{N} W(x_i - x_j)$$

- Display a first-order phase transition that can be analyzed at MF level via the free energy:

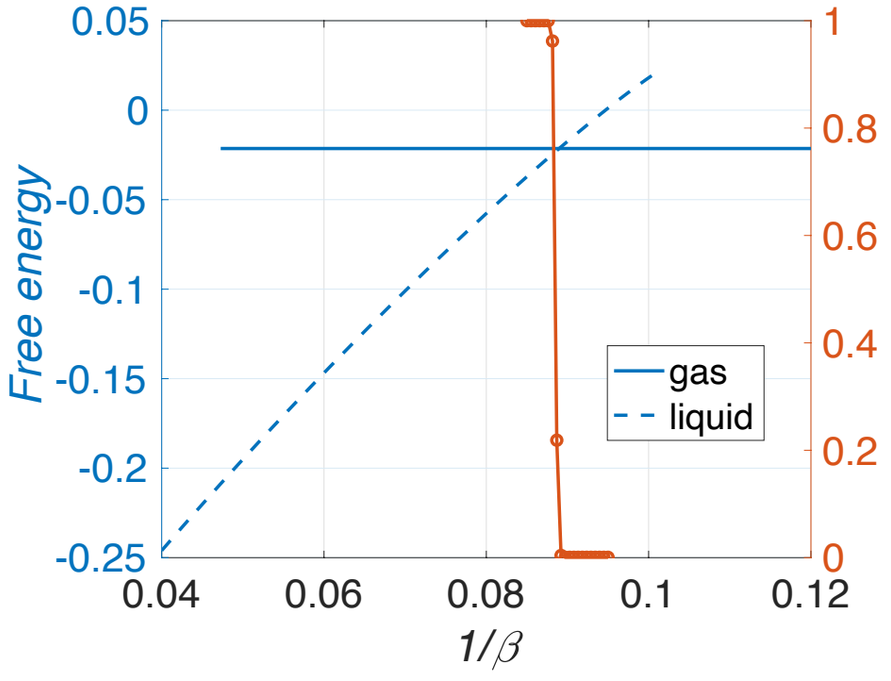
$$E(\rho) = \frac{1}{2} \int_{B^2} W(x-y)\rho(x)\rho(y)dxdy + k_B T \int_B \rho(x) \log \rho(x) dx$$

Gas and liquid-like phases $N = 512$





Free energy & transition correctly detected by training the NF



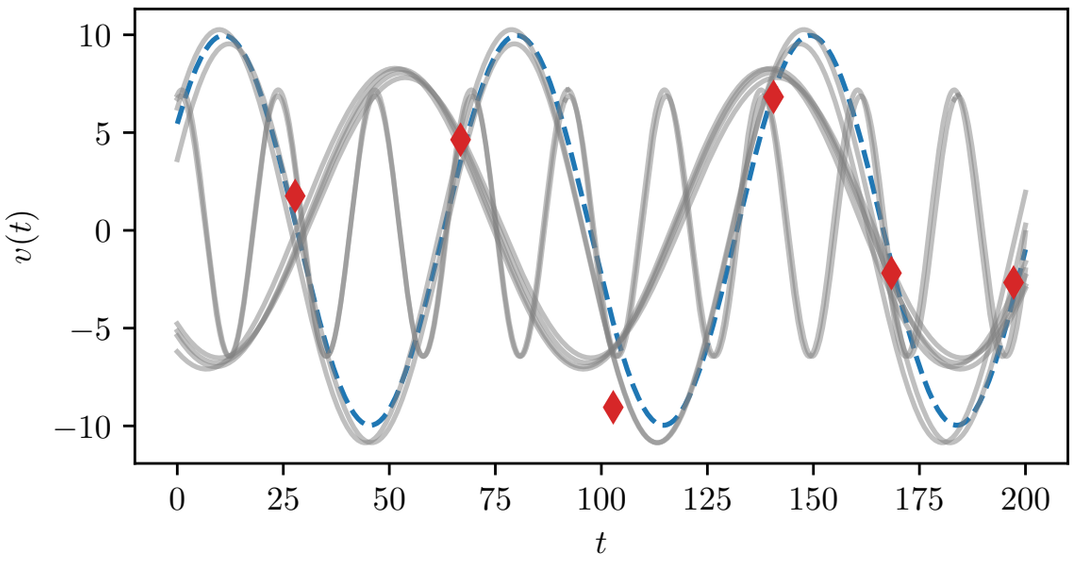
MCMC with NF for Bayesian Inference

with Marylou Gabrié & Grant Rotskoff

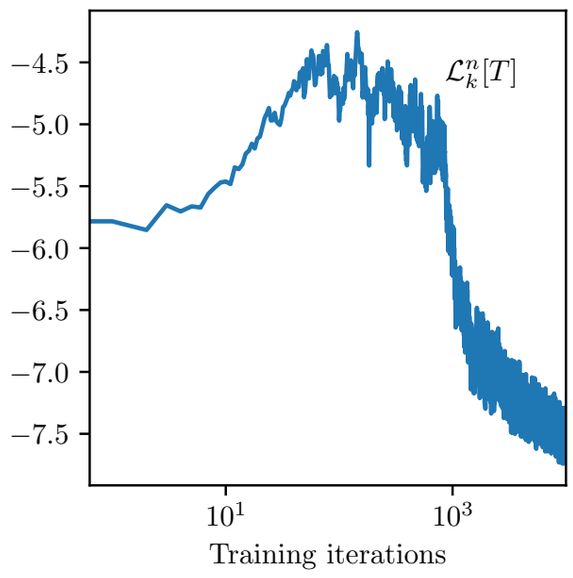
- Sampling of challenging (e.g. multimodal) posterior distributions;
- Allows estimation of the evidence = partition function used for model validation/selection

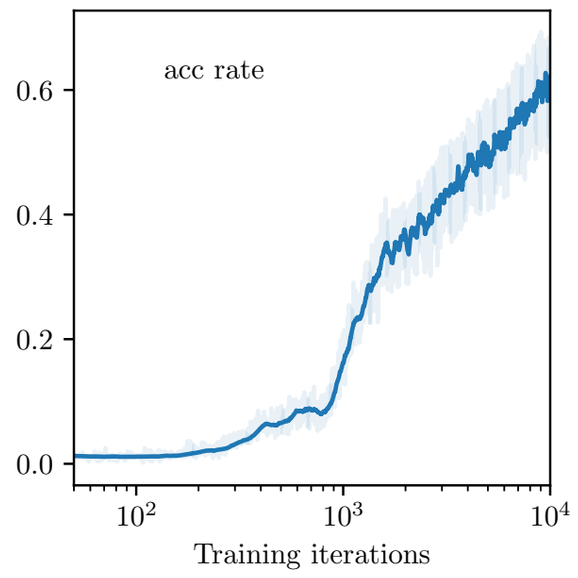
Application to inference of exoplanet radial velocity

Price-Whelan et al. The Joker: A Custom Monte-Carlo Sampler for Binary-Star and Exoplanet Radial Velocity Data. Astro. J., 837, 2017.



Data (♦) from signal (dashed blue) and samples from MCMCM (grey)





Loss (right), acceptance rate (left)

The Unreasonable Effectiveness of Mathematics in the Natural Sciences

Richard Courant Lecture in Mathematical Sciences delivered at New York University, May 11, 1959

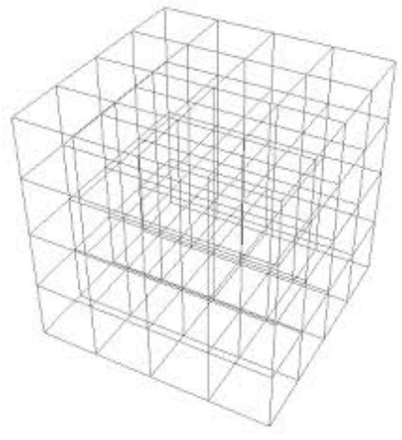
EUGENE P. WIGNER

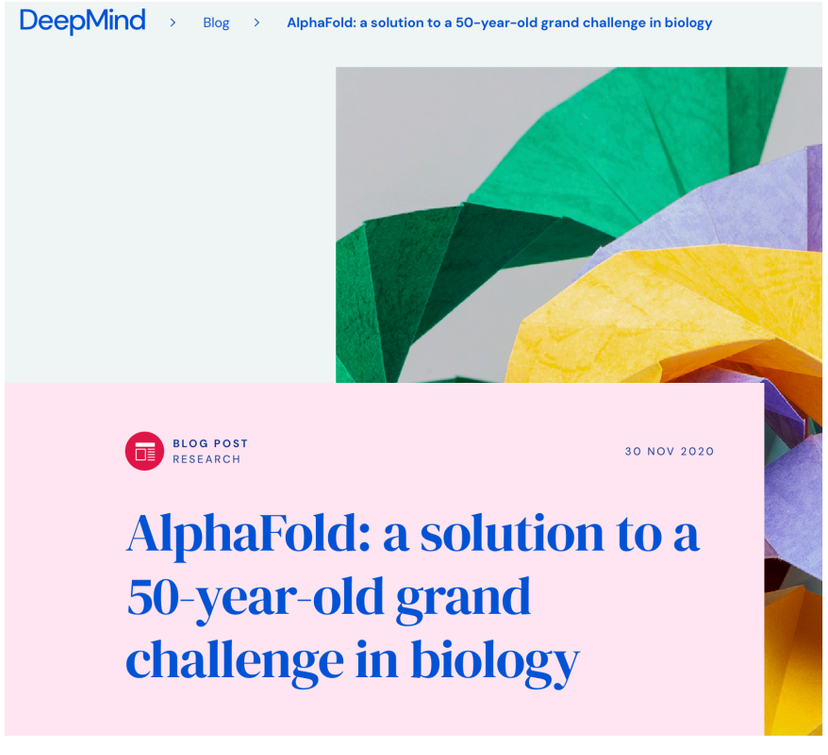
The Unreasonable Effectiveness of Machine Learning

Curses of Dimensionality (CoD):

The number of operations/parameters needed to optimize/integrate/approximate Lipschitz functions to precision δ depends exponentially on the input dimension d, $O(\delta^{-d})$.

[Bellman, 61]



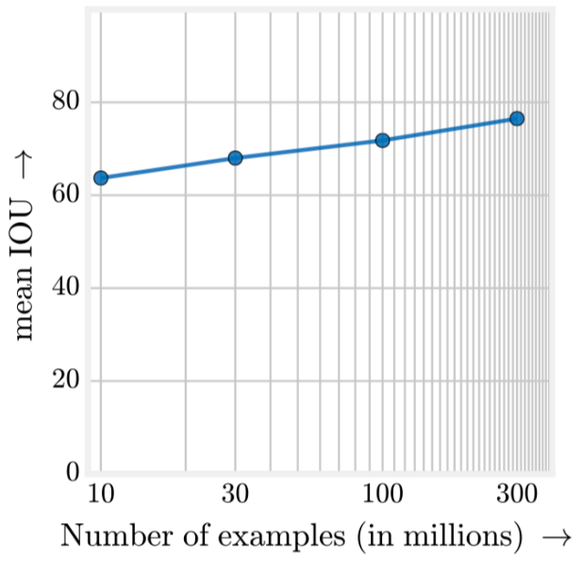


When, how, and why can neural networks approximate high dimensional functions?

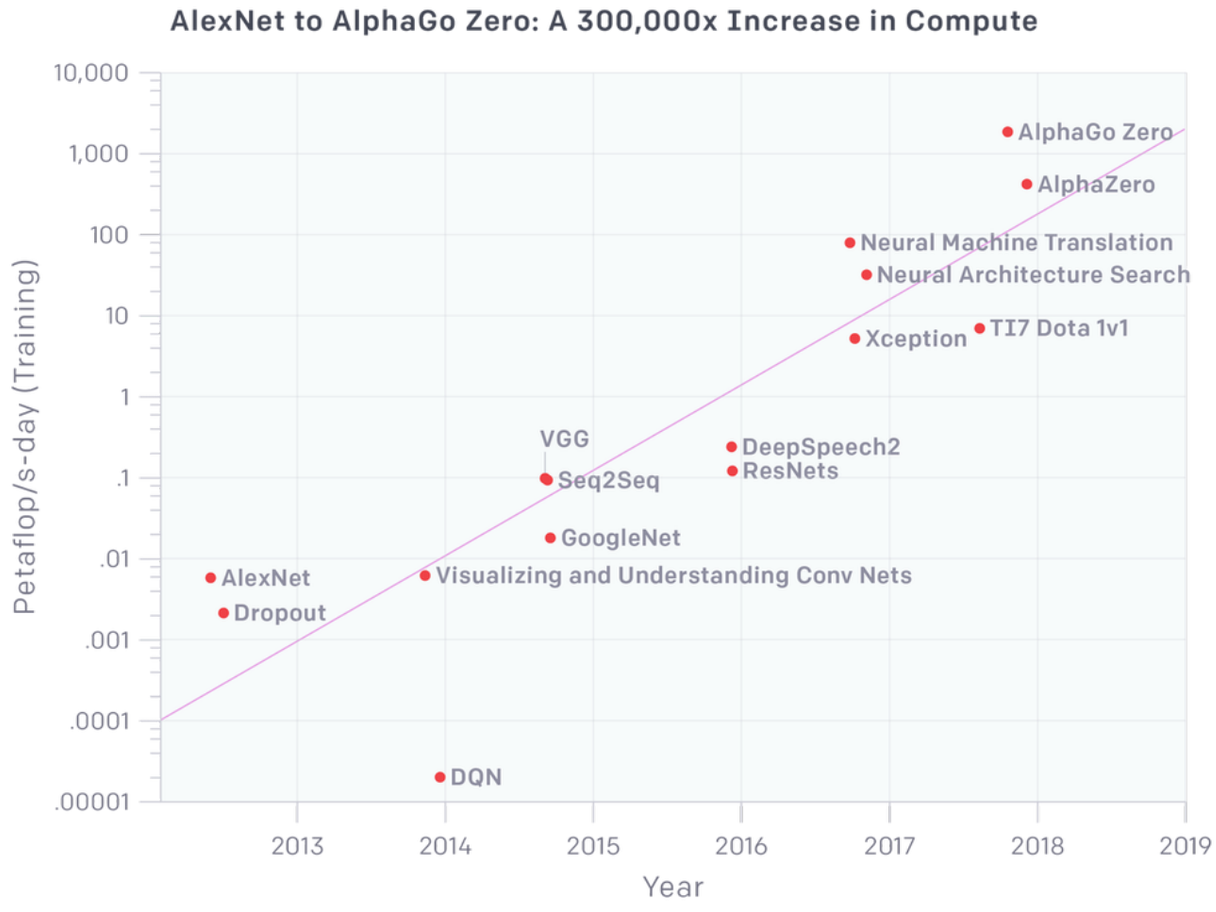
Need for Theory

DL is very costly in terms of compute and data.
Brute-force approach not sustainable.

Initialization	mIOU
ImageNet	73.6
300M	75.3
ImageNet+300M	**76.5**



[Sun et al ICCV 2017]



[Amodei & Hernandez, blog post, 2018]

Challenge & opportunity of MCMC: ***we often have a model and no prior data***

(i.e. we must be data-savvy since we must generate it,
but we can benchmark against the model ground truth.)

Performance increases logarithmically with data volume

AI doubling its compute every 3.5 months

Computer Science > Machine Learning

[Submitted on 30 Sep 2022 (v1), last revised 20 Oct 2022 (this version, v2)]

Building Normalizing Flows with Stochastic Interpolants

Michael S. Albergo, Eric Vanden-Eijnden

A simple generative model based on a continuous-time normalizing flow between any pair of base and target probability densities is proposed. The velocity field of this flow is inferred from the probability current of a time-dependent density that interpolates between the base and the target in finite time. Unlike conventional normalizing flow inference methods based the maximum likelihood principle, which require costly backpropagation through ODE solvers, our interpolant approach leads to a simple quadratic loss for the velocity itself which is expressed in terms of expectations that are readily amenable to empirical estimation. The flow can be used to generate samples from either the base or target, and to estimate the likelihood at any time along the interpolant. In addition, the flow can be optimized to minimize the path length of the interpolant density, thereby paving the way for building optimal transport maps. The approach is also contextualized in its relation to diffusions. In particular, in situations where the base is a Gaussian density, we show that the velocity of our normalizing flow can also be used to construct a diffusion model to sample the target as well as estimating its score. This allows one to map methods based on stochastic differential equations to those using ordinary differential equations, simplifying the mechanics of the model, but capturing equivalent dynamics. Benchmarking on density estimation tasks illustrates that the learned flow can match and surpass maximum likelihood continuous flows at a fraction of the conventional ODE training costs.

Subjects: **Machine Learning (cs.LG)**; Machine Learning (stat.ML)
Cite as: arXiv:2209.15571 **[cs.LG]**
(or arXiv:2209.15571v2 **[cs.LG]** for this version)
https://doi.org/10.48550/arXiv.2209.15571

Submission history

From: Michael Albergo [view email]
[v1] Fri, 30 Sep 2022 16:30:31 UTC (3,383 KB)
[v2] Thu, 20 Oct 2022 14:57:06 UTC (4,361 KB)

Physics > Data Analysis, Statistics and Probability

[Submitted on 26 May 2021]

Adaptive Monte Carlo augmented with normalizing flows

Marylou Gabrié, Grant M. Rotskoff, Eric Vanden-Eijnden

Many problems in the physical sciences, machine learning, and statistical inference necessitate sampling from a high-dimensional, multi-modal probability distribution. Markov Chain Monte Carlo (MCMC) algorithms, the ubiquitous tool for this task, typically rely on random, reversible, and local updates to propagate configurations of a given system in a way that ensures that generated configurations will be distributed according to a target probability distribution asymptotically. In high-dimensional settings with multiple relevant metastable basins, local approaches require either immense computational effort or intricately designed importance sampling strategies to capture information about, for example, the relative populations of such basins. Here we analyze a framework for augmenting MCMC sampling with nonlocal transition kernels parameterized with generative models known as normalizing flows. We focus on a setting where there is no preexisting data, as is commonly the case for problems in which MCMC is used. Our results emphasize that the implementation of the normalizing flow must be adapted to the structure of the target distribution in order to preserve the statistics of the target at all scales. Furthermore, we analyze the propensity of our algorithm to discover new states and demonstrate the importance of initializing the training with some \emph{a priori} knowledge of the relevant modes. We show that our algorithm can sample effectively across large free energy barriers, providing dramatic accelerations relative to traditional MCMC algorithms.

Subjects: **Data Analysis, Statistics and Probability (physics.data-an)**; Disordered Systems and Neural Networks (cond-mat.dis-nn); Statistical Mechanics (cond-mat.stat-mech)

Cite as: arXiv:2105.12603 **[physics.data-an]**

(or arXiv:2105.12603v1 **[physics.data-an]** for this version)

Submission history

From: Grant Rotskoff [view email]

[v1] Wed, 26 May 2021 15:03:07 UTC (6,252 KB)

Efficient Bayesian Sampling Using Normalizing Flows to Assist Markov Chain Monte Carlo Methods

Marylou Gabrié, Grant M. Rotskoff, Eric Vanden-Eijnden

02 Jun 2021 (modified: 08 Jul 2021) INNF+ 2021 contributedtalk Readers: Everyone Show Bibtex Show Revisions

Keywords: MCMC, normalizing flows, Bayesian inference

TL;DR: We present a concurrent scheme where a normalizing flow is used to speed-up a MCMC scheme and the data from the MCMC is used to train the flow, with applications to Bayesian posterior distribution sampling.

Abstract: Normalizing flows can generate complex target distributions and thus show promise in many applications in Bayesian statistics as an alternative or complement to MCMC for sampling posteriors.

Since no data set from the target posterior distribution is available beforehand, the flow is typically trained using the reverse Kullback-Leibler (KL) divergence that only requires samples from a base distribution. This strategy may perform poorly when the posterior is complicated and hard to sample with an untrained normalizing flow.

Here we explore a distinct training strategy, using the direct KL divergence as loss, in which samples from the posterior are generated by (i) assisting a local MCMC algorithm on the posterior with a normalizing flow to accelerate its mixing rate and (ii) using the data generated this way to train the flow.

The method only requires a limited amount of \textit{a~priori} input about the posterior, and can be used to estimate the evidence required for model validation, as we illustrate on examples.