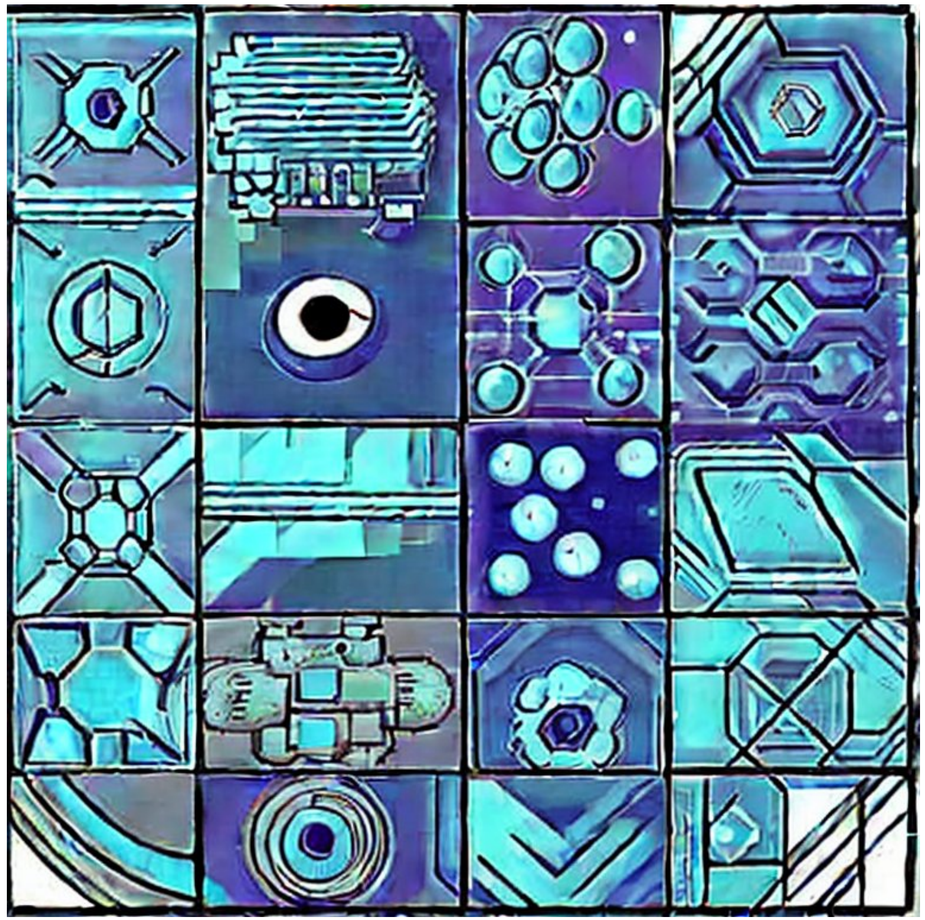


Machine learning for physicists

https://github.com/wangleiphy/ml4p

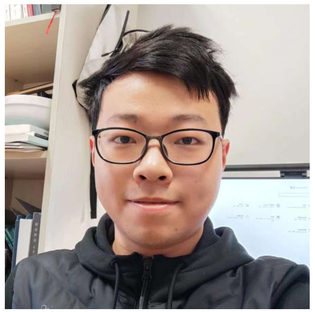
每周四上午10点

2.23	Overview
3.2	Machine learning practices
3.9	A hitchhiker’s guide to deep learning
3.16	Research projects hands-on
3.23	Symmetries in machine learning
3.30	Differentiable programming
4.6	Generative models-I
4.13	Generative models-II
4.20	Research projects presentation
4.27	AI for science: why now ?

课程微信群

助教

李子航

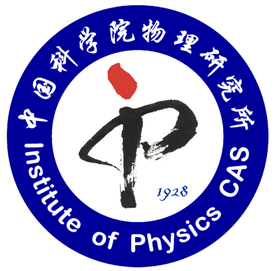


李扬帆

考核方式： project + presentation （1学分）

Generative AI for Science

Lei Wang (王磊)
Institute of Physics, CAS
https://wangleiphy.github.io



Generative models and their physics genes

Goodfellow,
NIPS tutorial, 1701.00160

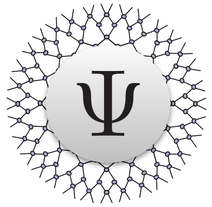
$p(\boldsymbol{x})$

Direct

GAN

Explicit density

Implicit density



Tensor Networks

Tractable density

-Fully visible belief nets

-NADE

-MADE

-PixelRNN

-Change of variables models (nonlinear ICA)

Approximate density

Variational

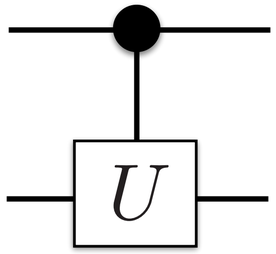
Variational autoencoder

Markov Chain

Boltzmann machine

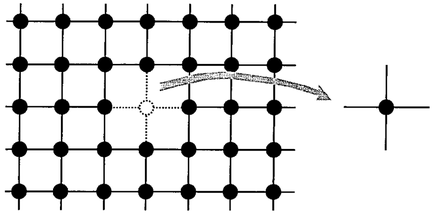
Markov Chain

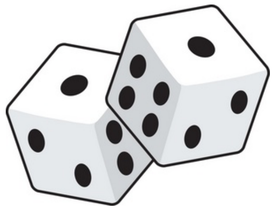
GSN

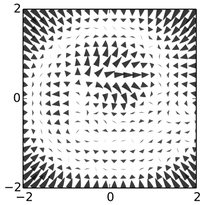


Quantum Circuits

+Diffusion models







Variational autoencoders

Kingma, Welling, 1312.6114

Close connection to the variational calculus we have learned

$$p(\boldsymbol{x}) = \frac{e^{-\beta E(\boldsymbol{x})}}{Z}$$

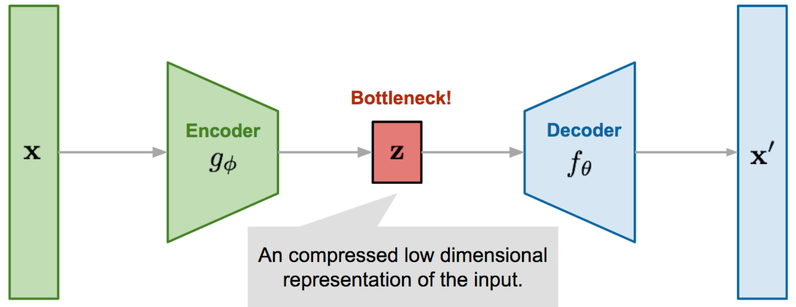
$$p(\boldsymbol{z}\,|\,\boldsymbol{x}) = \frac{p(\boldsymbol{x},\boldsymbol{z})}{p(\boldsymbol{x})}$$

Variational free energy

$$\int d\boldsymbol{x}\, q(\boldsymbol{x})\left[\ln q(\boldsymbol{x}) + \beta E(\boldsymbol{x})\right] \geq -\ln Z$$

Variational Bayes/Variational inference

$$\int d\boldsymbol{z}\, q(\boldsymbol{z}\,|\,\boldsymbol{x})\left[\ln q(\boldsymbol{z}\,|\,\boldsymbol{x}) - \ln p(\boldsymbol{x},\boldsymbol{z})\right] \geq -\ln p(\boldsymbol{x})$$



For each data we introduce

$$\mathcal{L}(\boldsymbol{x}) = \langle -\ln p(\boldsymbol{x}, \boldsymbol{z}) + \ln q(\boldsymbol{z}|\boldsymbol{x}) \rangle_{z\sim q(z|x)}, \tag{53}$$

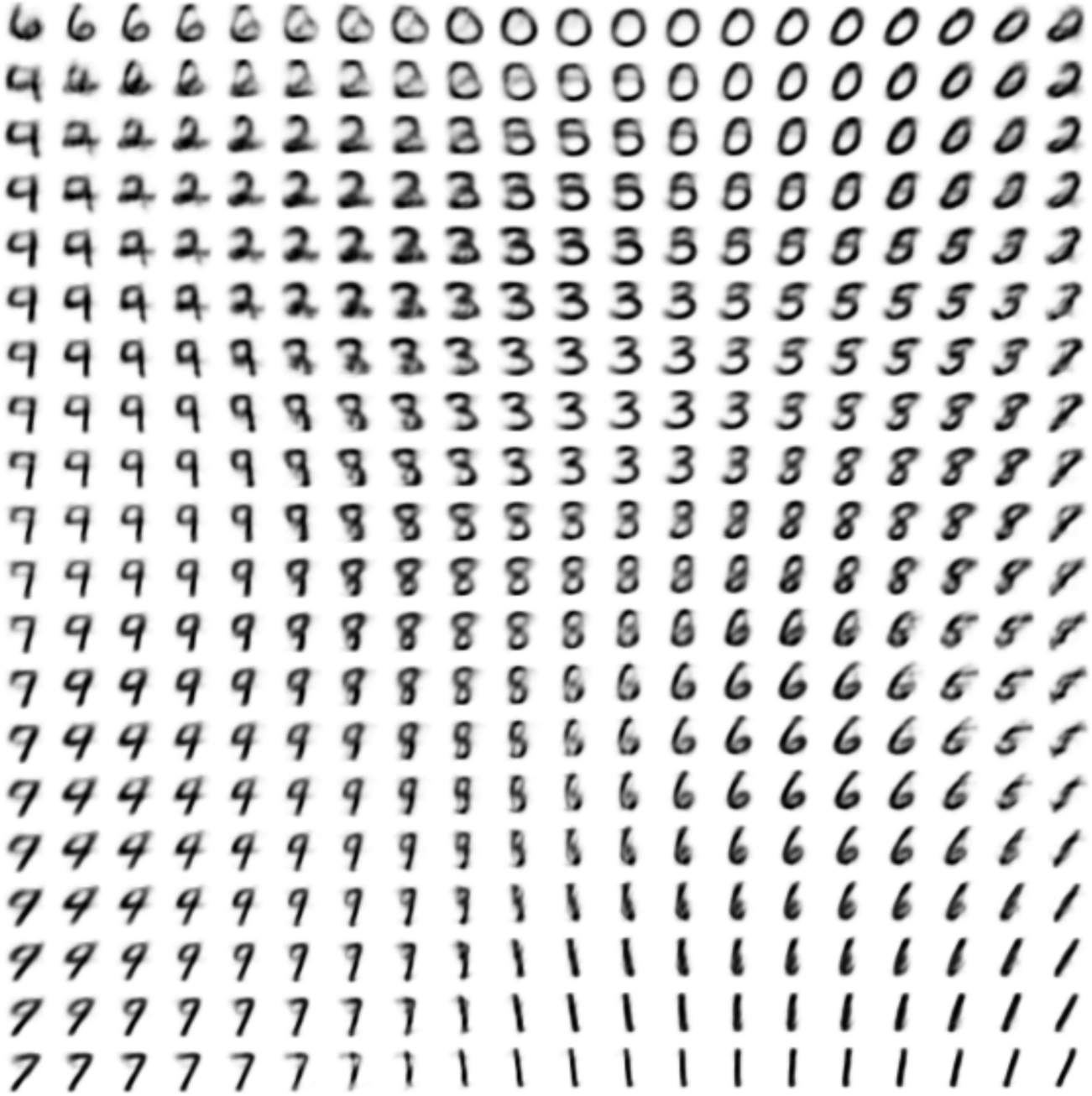
which is a variational upper bound of $-\ln p(\boldsymbol{x})$ since $\mathcal{L}(\boldsymbol{x}) + \ln p(\boldsymbol{x}) = \mathbb{KL}(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})) \geq 0$. We see that $q(\boldsymbol{z}|\boldsymbol{x})$ provides a variational approximation of the posterior $p(\boldsymbol{z}|\boldsymbol{x})$. By minimizing $\mathcal{L}$ one effectively pushes the two distributions together. And the variational free energy becomes exact only when $q(\boldsymbol{z}|\boldsymbol{x})$ matches to $p(\boldsymbol{z}|\boldsymbol{x})$. In fact, $-\mathcal{L}$ is called evidence lower bound (ELBO) in variational inference.

We can obtain an alternative form of the variational free energy

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) = -\langle \ln p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) \rangle_{z\sim q_{\boldsymbol{\phi}}(z|x)} + \mathbb{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})). \tag{54}$$

The first term of Eq. (54) is the reconstruction negative log-likelihood, while the second term is the KL divergence between the approximate posterior distribution and the latent prior. We also be explicit about the network parameters $\boldsymbol{\theta}, \boldsymbol{\phi}$ of the encoder and decoder.

http://wangleiphy.github.io/lectures/PILtutorial.pdf

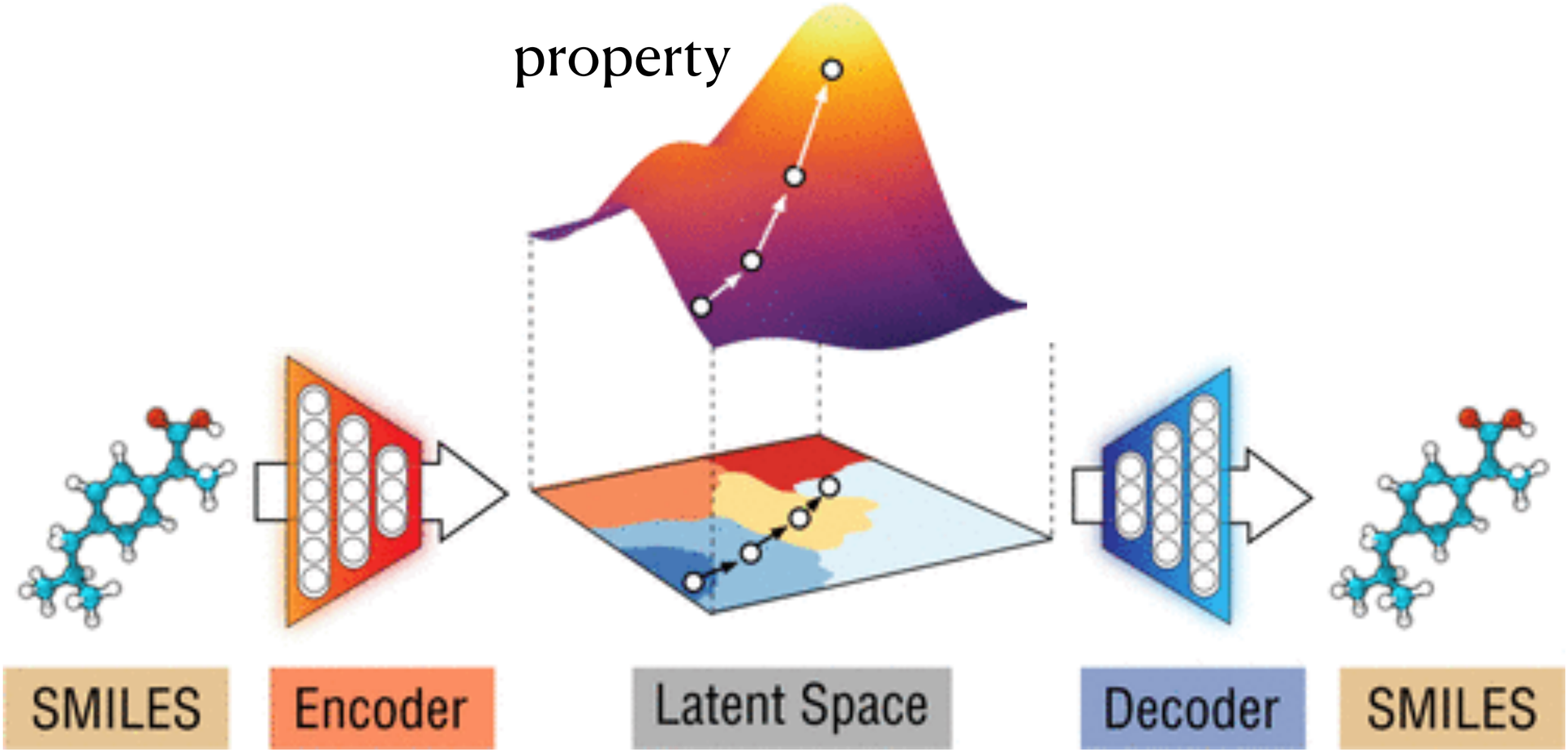


Learned MNIST latent space

Kingma, Welling, 1312.6114

Chemical design using continuous latent variables

Gomez-Bombarelli et al,1610.02415



Normalizing flows

Parallel WaveNet 1711.10433

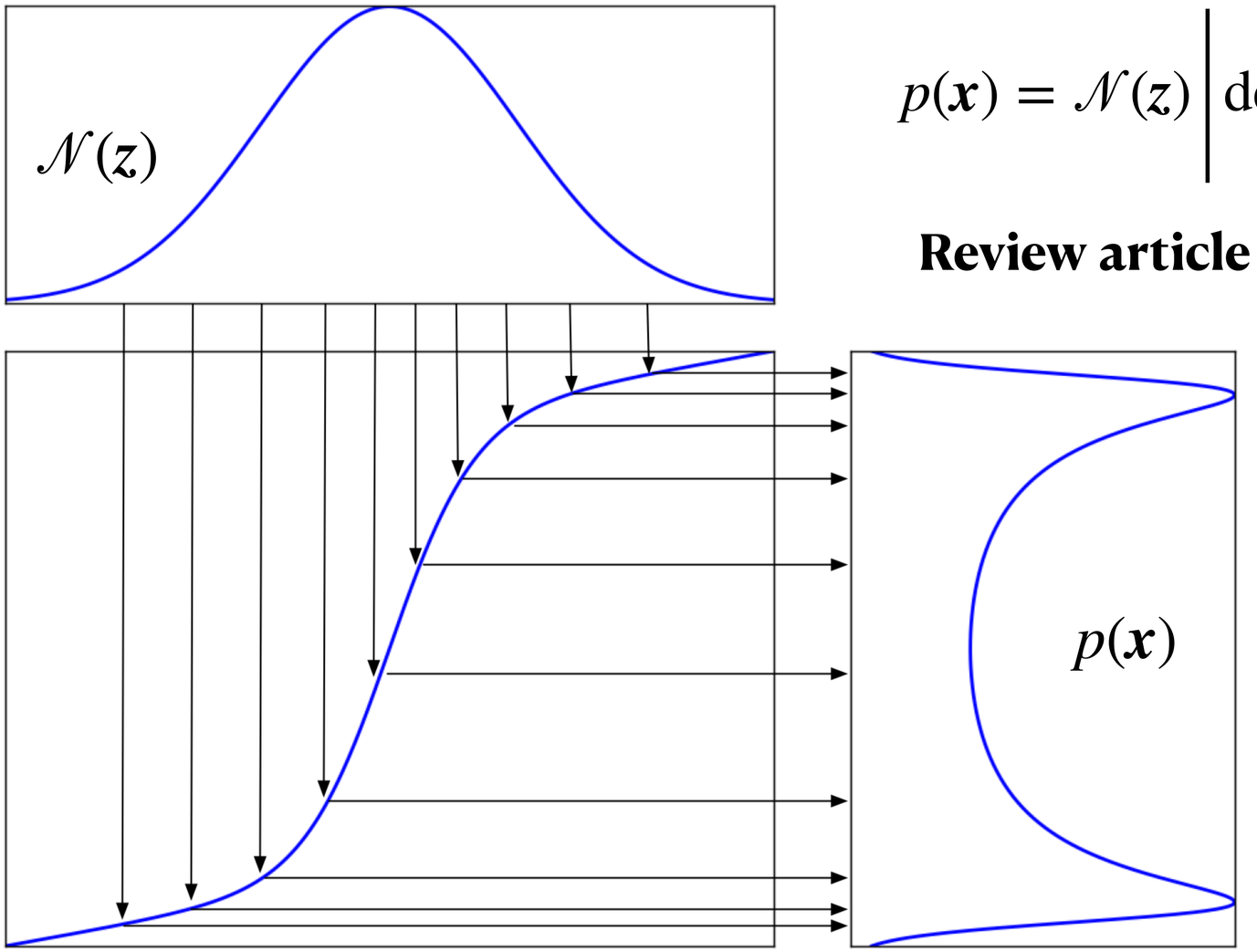
https://deepmind.com/blog/high-fidelity-speech-synthesis-wavenet/

Glow 1807.03039

https://blog.openai.com/glow/

Normalizing flow in a nutshell

Base density



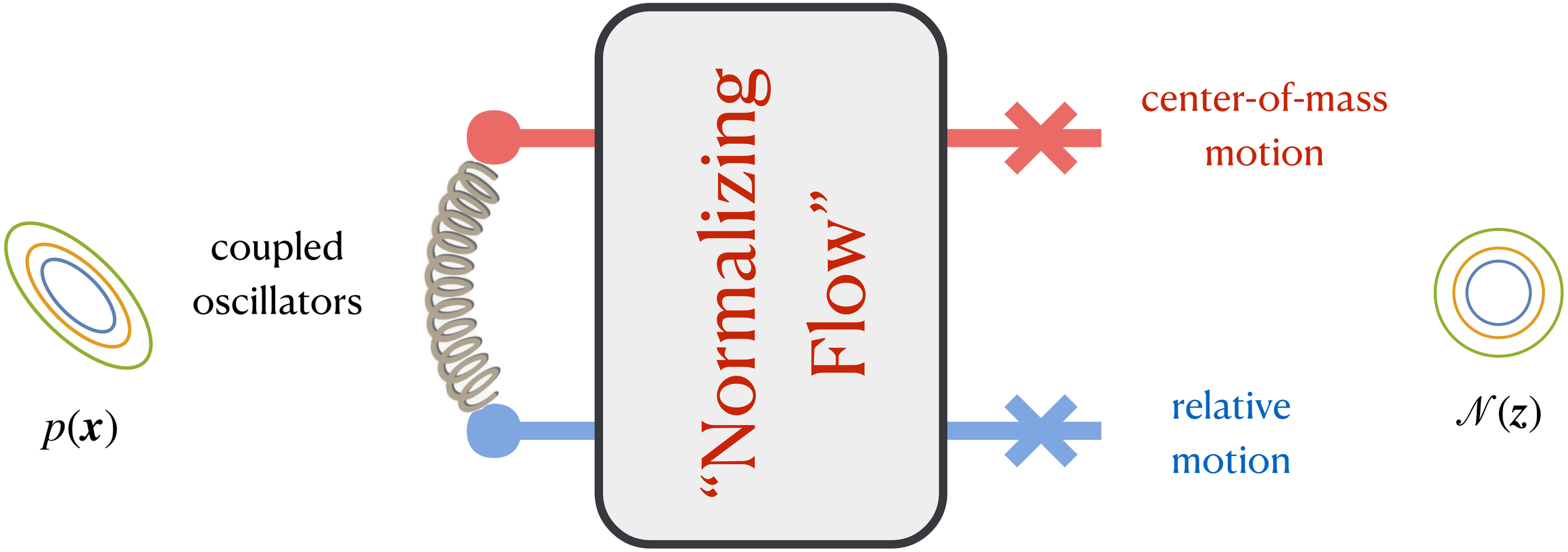
$$p(\boldsymbol{x}) = \mathcal{N}(z)\left|\det\left(\frac{\partial z}{\partial \boldsymbol{x}}\right)\right|$$

Review article 1912.02762

"neural net" with 1 neuron

Target density

Normalizing flow for physics: an intuition



High-dimensional, composable, learnable, nonlinear transformations

Example of a building block

Forward

arbitrary neural nets

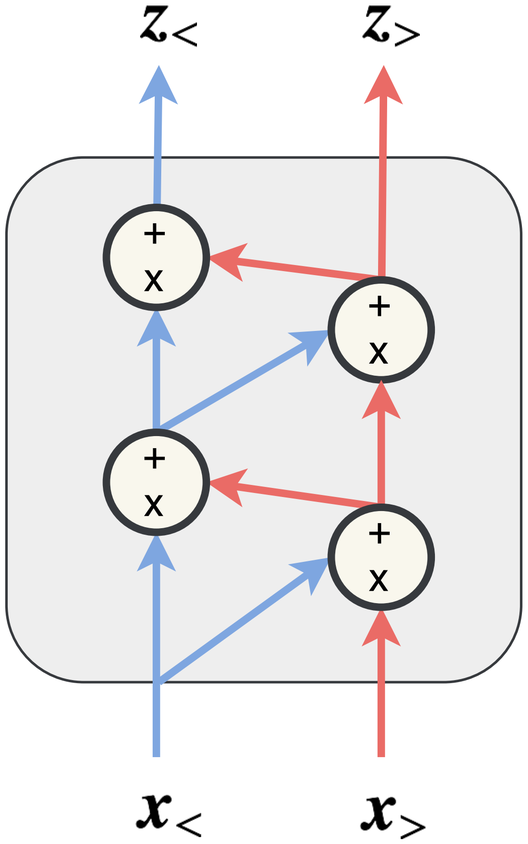
$$\begin{cases} \boldsymbol{x}_< = \boldsymbol{z}_< \\ \boldsymbol{x}_> = \boldsymbol{z}_> \odot e^{s(\boldsymbol{z}_<)} + t(\boldsymbol{z}_<) \end{cases}$$

Inverse

$$\begin{cases} \boldsymbol{z}_< = \boldsymbol{x}_< \\ \boldsymbol{z}_> = (\boldsymbol{x}_> - t(\boldsymbol{x}_<)) \odot e^{-s(\boldsymbol{x}_<)} \end{cases}$$

Log-Abs-Jacobian-Det

$$\ln \left| \det \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{z}} \right) \right| = \sum_i [s(\boldsymbol{z}_<)]_i$$

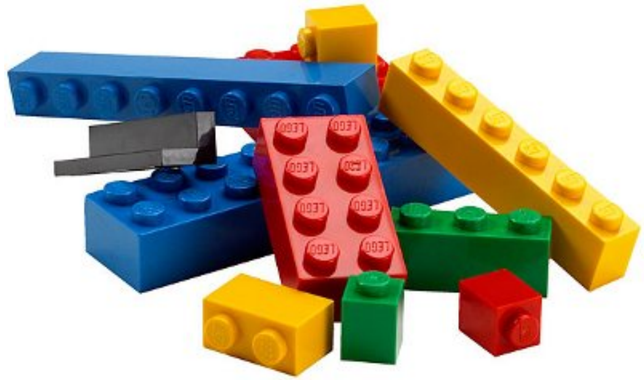


Real NVP, Dinh et al,1605.08803

Turns out to have surprising connection Störmer–Verlet integration (later)

Flow architecture design

Composability

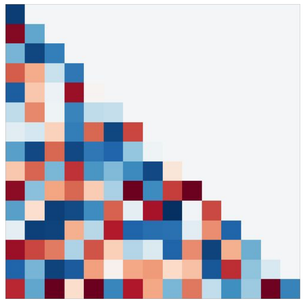


$$z = \mathcal{T}(\boldsymbol{x})$$

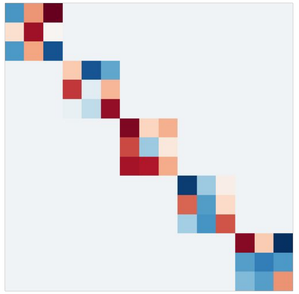
$$\mathcal{T} = \mathcal{T}_1 \circ \mathcal{T}_2 \circ \mathcal{T}_3 \circ \cdots$$

Balanced efficiency & inductive bias

$$\left|\det\left(\frac{\partial z}{\partial x}\right)\right|$$



Autoregressive



Neural RG

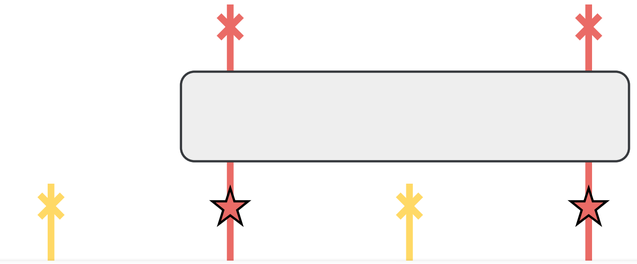
$$\frac{\partial p(\boldsymbol{x}, t)}{\partial t} + \nabla \cdot \left[p(\boldsymbol{x}, t)\boldsymbol{v}\right] = 0$$

Continuous flow

Neural network renormalization group

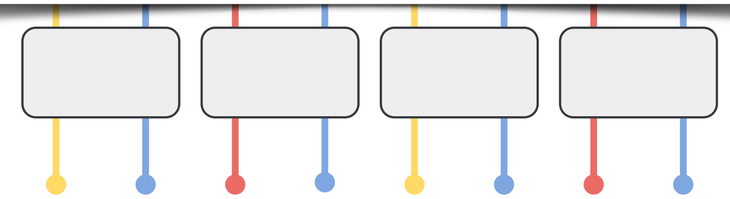
Li, LW, PRL '18 [li012589/NeuralRG]

Collective variables

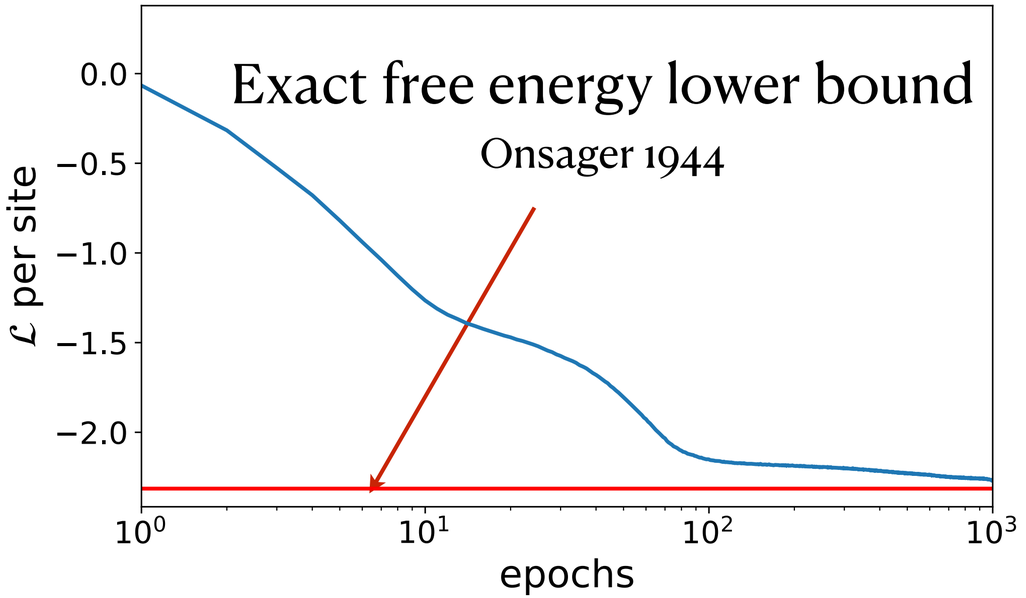


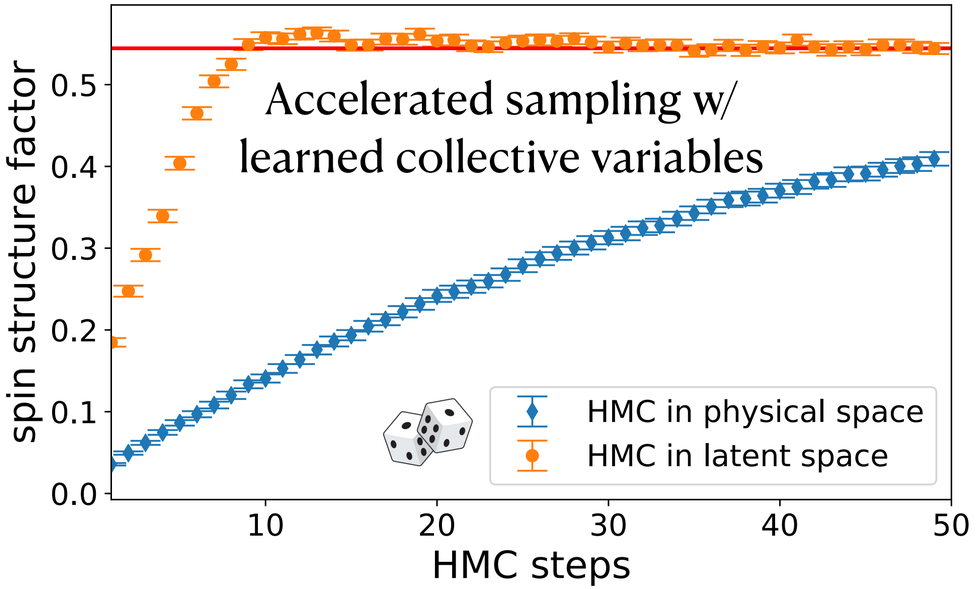
Probability Transformation

$$\ln p(\boldsymbol{x}) = \ln \mathcal{N}(\boldsymbol{z}) - \ln \left| \det \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{z}} \right) \right|$$

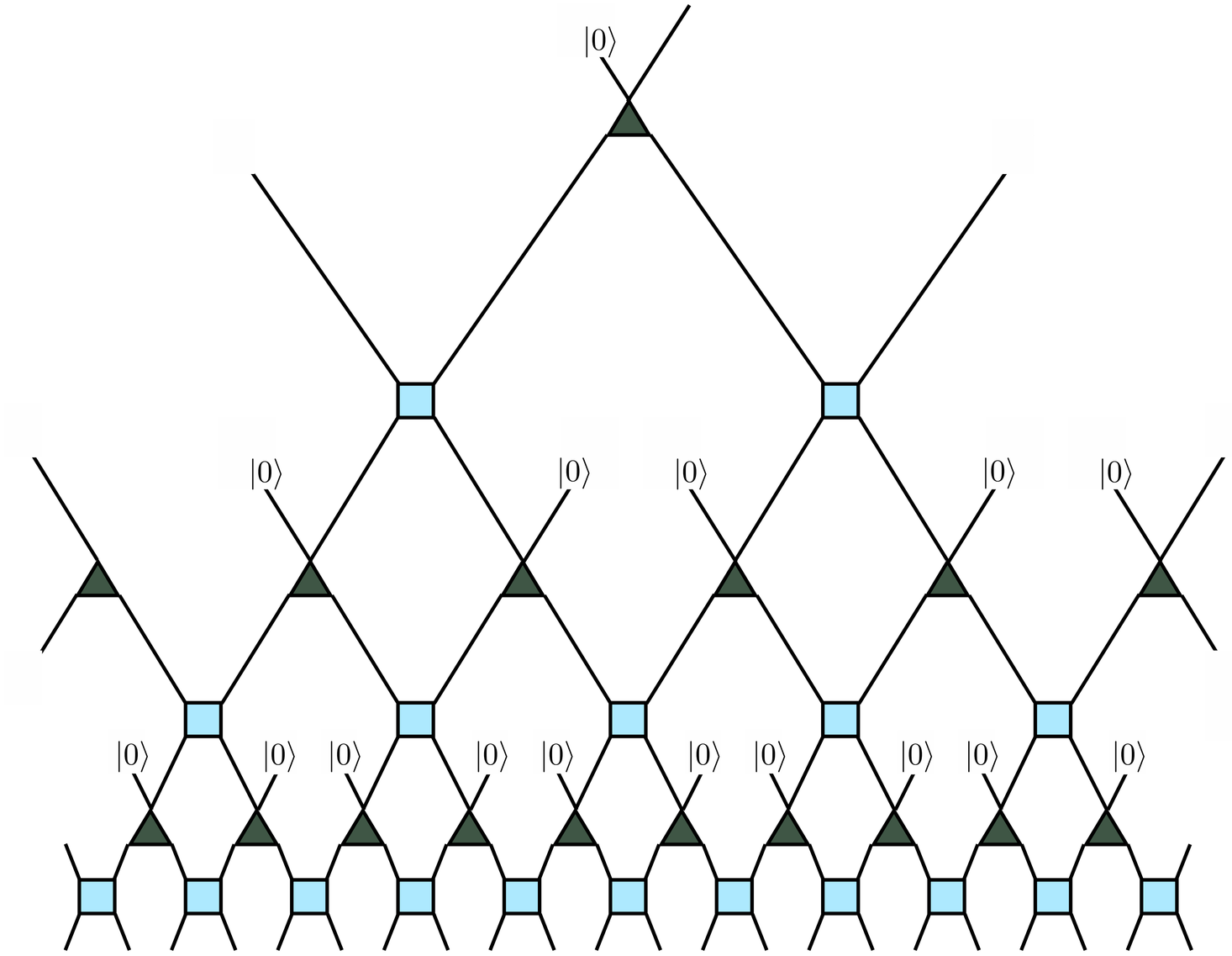


Physical variables

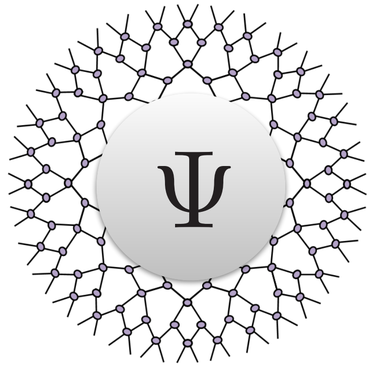




Quantum version of the architecture

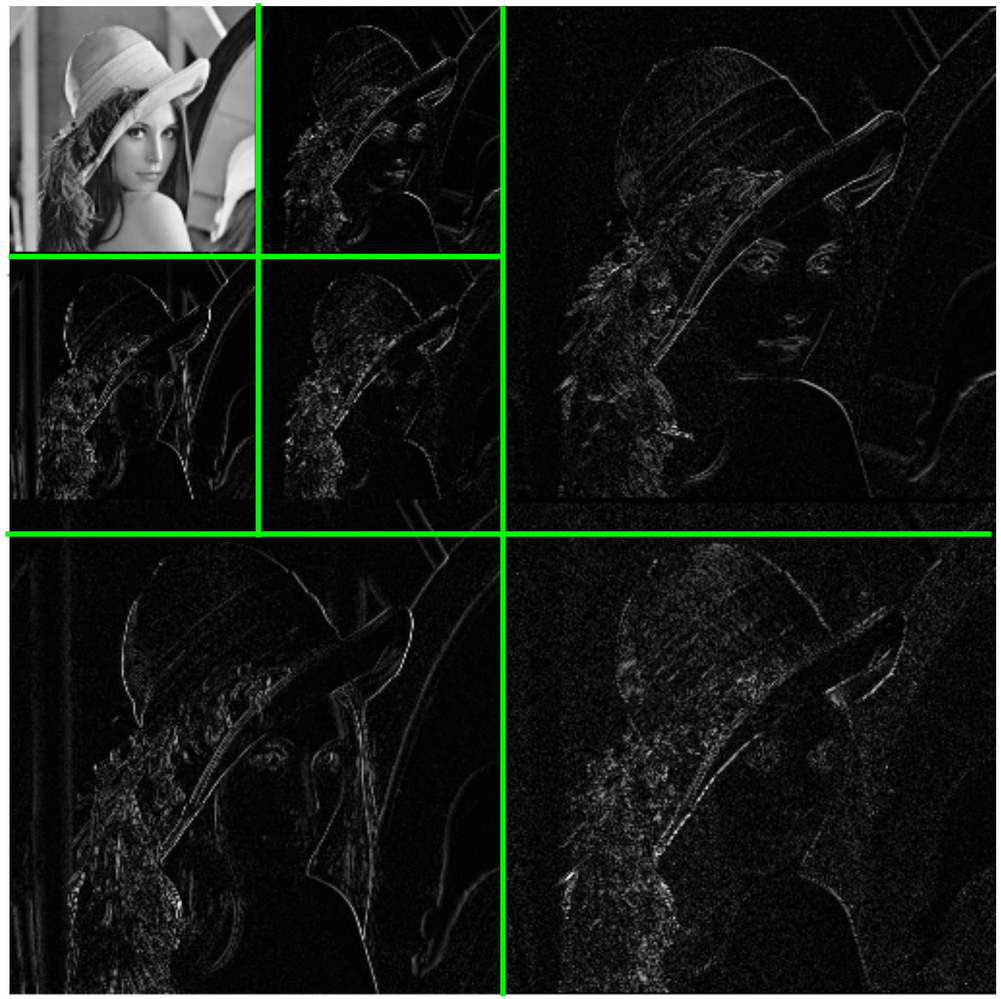


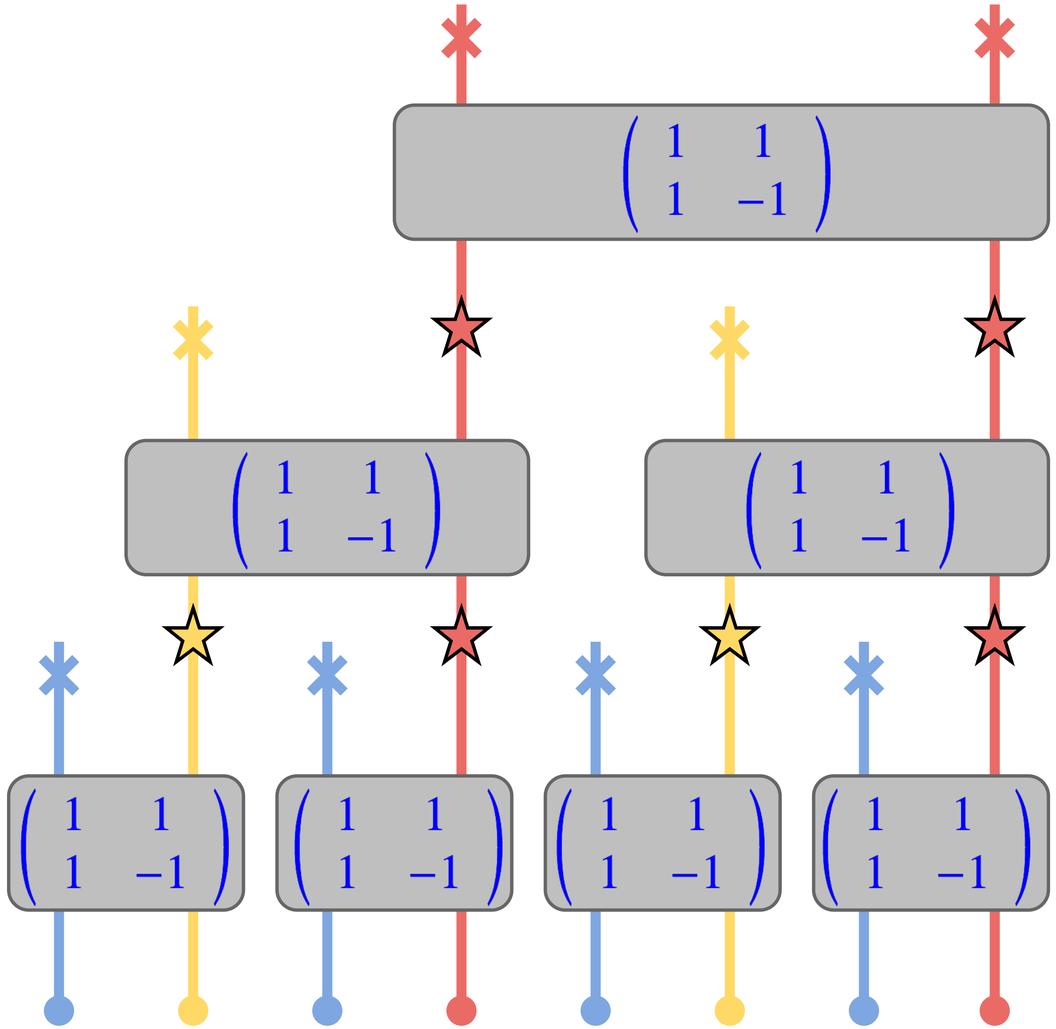
Entangled qubits



Multi-Scale
Entanglement
Renormalization
Ansatz

Connection to wavelets





Nonlinear & adaptive generalizations of wavelets

Guy, Wavelets & RG1999+ White, Evenbly, Qi, Wavelets, MERA, and holographic mapping 2013+

Continuous normalizing flows

$$\ln p(\boldsymbol{x}) = \ln \mathcal{N}(\boldsymbol{z}) - \ln \left| \det \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{z}} \right) \right|$$

Consider infinitesimal change-of-variables Chen et al 1806.07366

$$\boldsymbol{x} = \boldsymbol{z} + \varepsilon \boldsymbol{v}$$

$$\ln p(\boldsymbol{x}) - \ln \mathcal{N}(\boldsymbol{z}) = - \ln \left| \det \left(1 + \varepsilon \frac{\partial \boldsymbol{v}}{\partial \boldsymbol{z}} \right) \right|$$

$\varepsilon \to 0$

$t = 1$ $t = 0$

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}$$

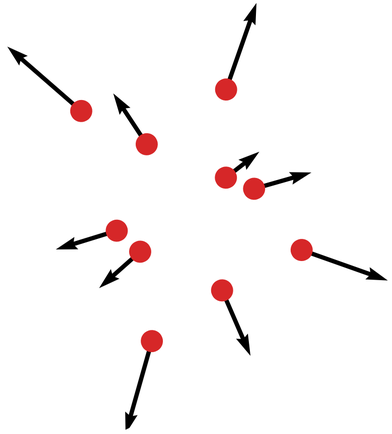
$$\frac{d \ln p(\boldsymbol{x}, t)}{dt} = - \nabla \cdot \boldsymbol{v}$$

Fluid physics behind flows

Zhang, E, LW 1809.10188
wangleiphy/MongeAmpereFlow

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}$$

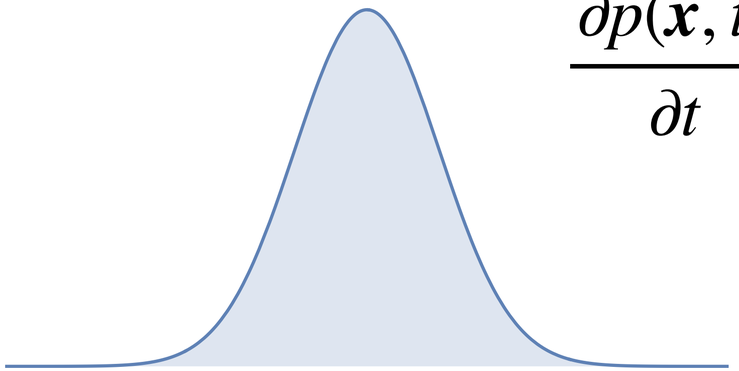
$$\frac{d \ln p(\boldsymbol{x}, t)}{dt} = -\nabla \cdot \boldsymbol{v}$$



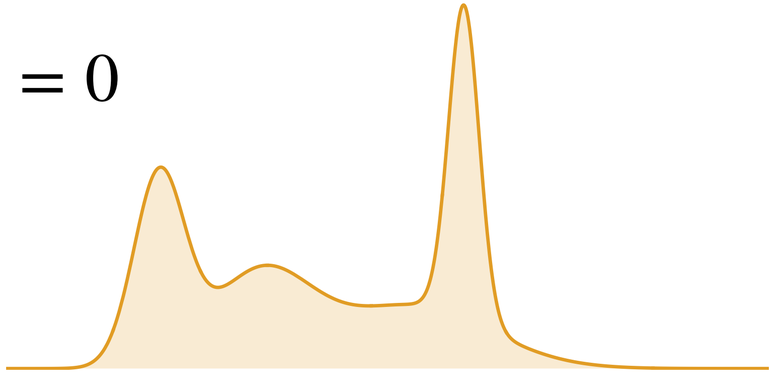
$$\frac{d}{dt} = \frac{\partial}{\partial t} + \boldsymbol{v} \cdot \nabla$$

"material derivative"

$$\frac{\partial p(\boldsymbol{x}, t)}{\partial t} + \nabla \cdot \left[p(\boldsymbol{x}, t) \boldsymbol{v} \right] = 0$$



Simple density



Complex density

Infinitesimal Flows Another way to reduce the computational overhead of normalizing flows is to use an ordinary differential equation to generate f [2]. In this case, the probability distribution changes over a finite time from $p(\mathbf{x}; 0)$ to $p(\mathbf{z}; T)$, where $\mathbf{z}$ is the end point of a curve defined by the ODE $\dot{\mathbf{x}}(t) = \mathbf{v}(\mathbf{x}(t))$, $\mathbf{x}(0) = \mathbf{x}$. For a small time step dt, we can approximate $\mathbf{x}(t + dt)$ to first order as $\mathbf{x}(t + dt) = \mathbf{x}(t) + dt\mathbf{v}(\mathbf{x}(t)) + \mathcal{O}(dt^2)$. Plugging this into Eq. 2 yields:

$$\begin{aligned} \log p(\mathbf{x} + dt\mathbf{v}(\mathbf{x}) + \mathcal{O}(dt^2); t + dt) &= \log p(\mathbf{x}; t) - \log |\mathbf{J}_f(\mathbf{x})| && (3) \\ &= \log p(\mathbf{x}; t) - \log |\mathbf{I} + dt\mathbf{J}_{\mathbf{v}}(\mathbf{x}) + \mathcal{O}(dt^2)| && (4) \end{aligned}$$

Taking a Taylor series gives:

$$\log p(\mathbf{x}; t + dt) + dt\mathbf{v}(\mathbf{x})^T \nabla \log p(\mathbf{x}; t + dt) = \log p(x; t) - dt\mathrm{Tr}(\mathbf{J}_v(\mathbf{x})) + \mathcal{O}(dt^2) \quad (5)$$

which, in the limit as $dt \to 0$, becomes:

$$\frac{\partial \log p(\mathbf{x}; t)}{\partial t} = -\mathbf{v}(\mathbf{x})^T \nabla \log p(\mathbf{x}; t) - \mathrm{Tr}(\mathbf{J}_{\mathbf{v}}(\mathbf{x})) = -\mathbf{v}^T \nabla \log p(\mathbf{x}; t) - \nabla \cdot \mathbf{v} \quad (6)$$

after some rearranging of terms. Here $\nabla\cdot$ is the divergence of a vector field, which is just another way of writing the trace of the Jacobian. The right-hand side of this equation is also the trace of the *Stein operator* of the distribution $p(\mathbf{x})$ applied to the function $\mathbf{v}(\mathbf{x})$, and plays a critical role in Stein variational gradient descent (SVGD) [13]. Switching from the log density to the density (and dropping the t for clarity), we find this expression can be simplified considerably:

$$\begin{aligned} \frac{1}{p(\mathbf{x})} \frac{\partial p(\mathbf{x})}{\partial t} &= -\mathbf{v}^T \frac{\nabla p(\mathbf{x})}{p(\mathbf{x})} - \nabla \cdot \mathbf{v} \\ \frac{\partial p(\mathbf{x})}{\partial t} &= -\mathbf{v}^T \nabla p(\mathbf{x}) - p(\mathbf{x}) \nabla \cdot \mathbf{v} \\ &= -\nabla \cdot (\mathbf{v}(\mathbf{x}) p(\mathbf{x})) \quad (7) \end{aligned}$$

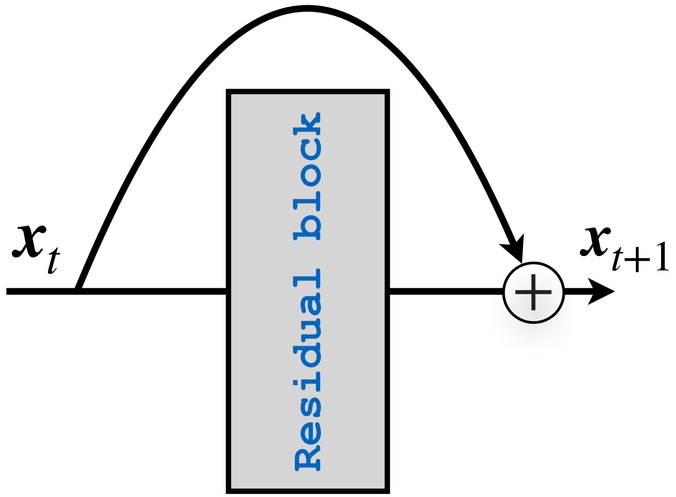
This may also be familiar as the drift term of the Fokker-Planck equation [11, Eq. 6.48] or the continuity equation for conservation of mass in fluid mechanics. We will denote the change to a

Pfau & Rezende

2012.02035

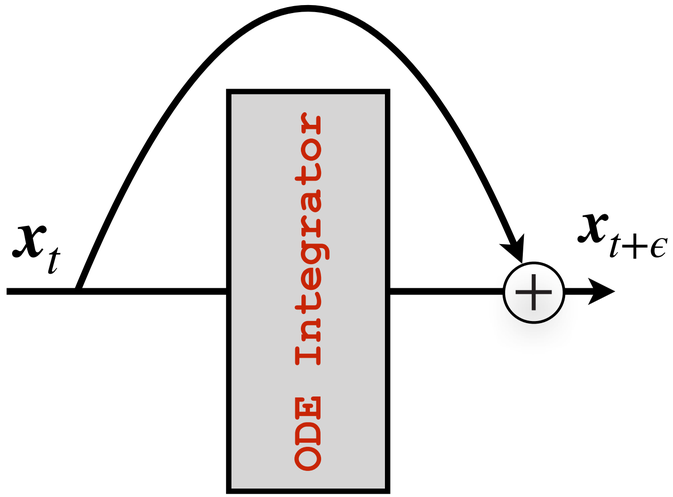
Neural Ordinary Differential Equations

Residual network



$$x_{t+1} = x_t + v(x_t)$$

ODE integration



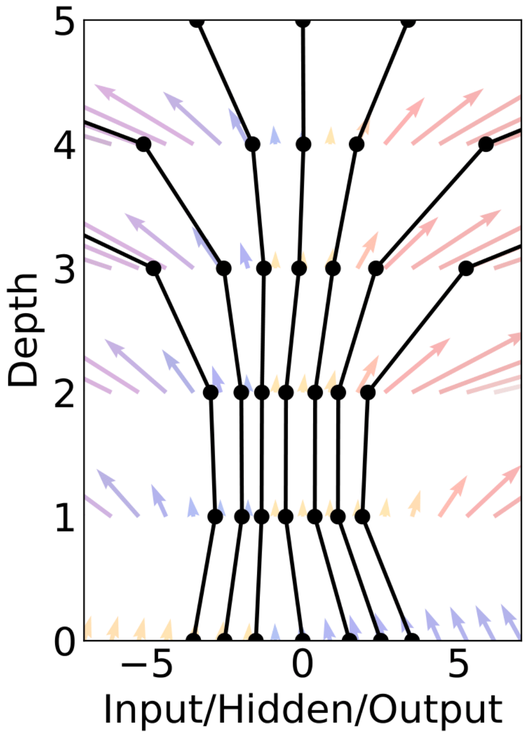
$$dx/dt = v(x)$$

Chen et al, 1806.07366

Harbor el al 1705.03341
Lu et al 1710.10121,
E Commun. Math. Stat 17'...

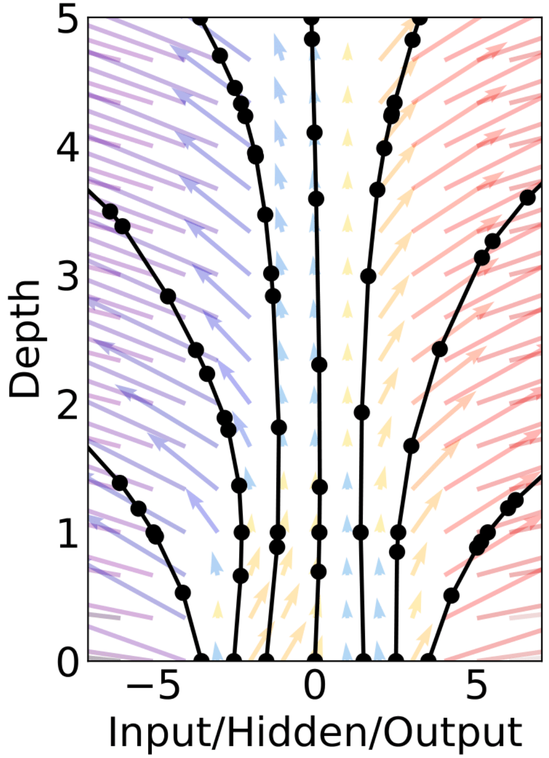
Neural Ordinary Differential Equations

Residual network



$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{v}(\boldsymbol{x}_t)$$

ODE integration



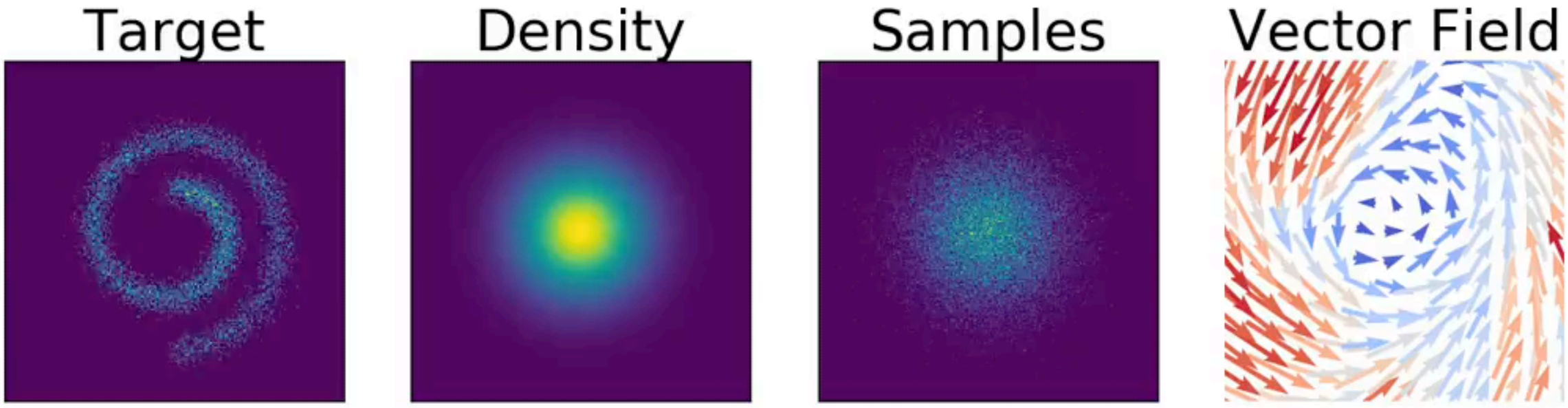
$$d\boldsymbol{x}/dt = \boldsymbol{v}(\boldsymbol{x})$$

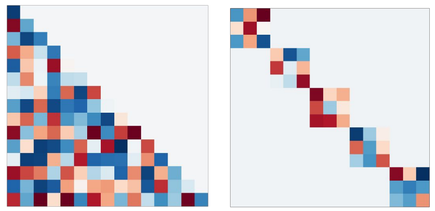
Chen et al, 1806.07366

Harbor el al 1705.03341
Lu et al 1710.10121,
E Commun. Math. Stat 17'...

Continuous normalizing flows implemented with NeuralODE

Chen et al, 1806.07366, Grathwohl et al 1810.01367





Continuous normalizing flow have no structural constraints on the transformation Jacobian

The two use cases

Zhang, E, LW, 1809.10188

Maximum likelihood estimation

$$\mathcal{N}(\boldsymbol{x}) \xleftarrow{\int_T^0 \dot{\boldsymbol{x}}\, dt}$$

"learn from data"

$$\mathscr{L} = -\,\mathbb{E}_{\boldsymbol{x}\sim\text{data}}\left[\ln p(\boldsymbol{x})\right]$$

Variational free energy

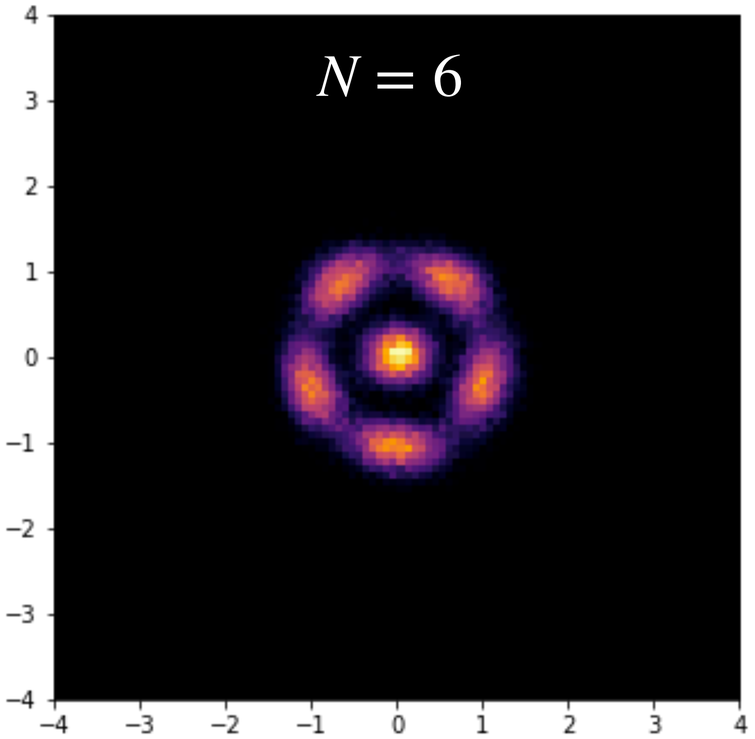
$$\mathcal{N}(\boldsymbol{x}) \xrightarrow{\int_0^T \dot{\boldsymbol{x}}\, dt} \frac{e^{-E(\boldsymbol{x})}}{Z}$$

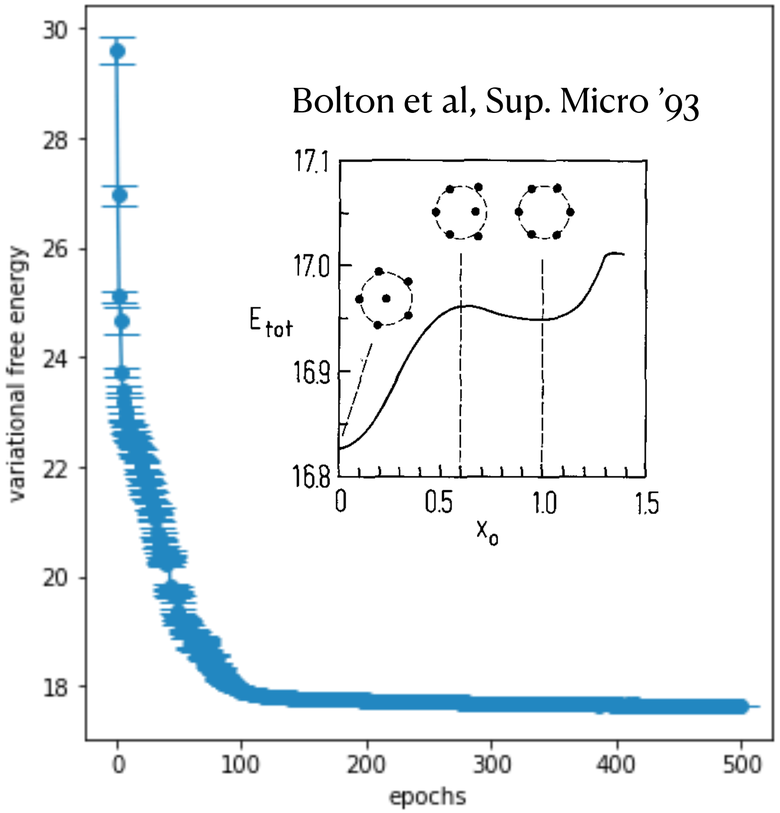
"learn from Hamiltonian"

$$F = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[k_B T \ln p(\boldsymbol{x}) + H(\boldsymbol{x})\right]$$

Demo: Classical Coulomb gas in a harmonic trap

$$H = \sum_{i<j} \frac{1}{|\boldsymbol{x}_i - \boldsymbol{x}_j|} + \sum_i^N \boldsymbol{x}_i^2$$



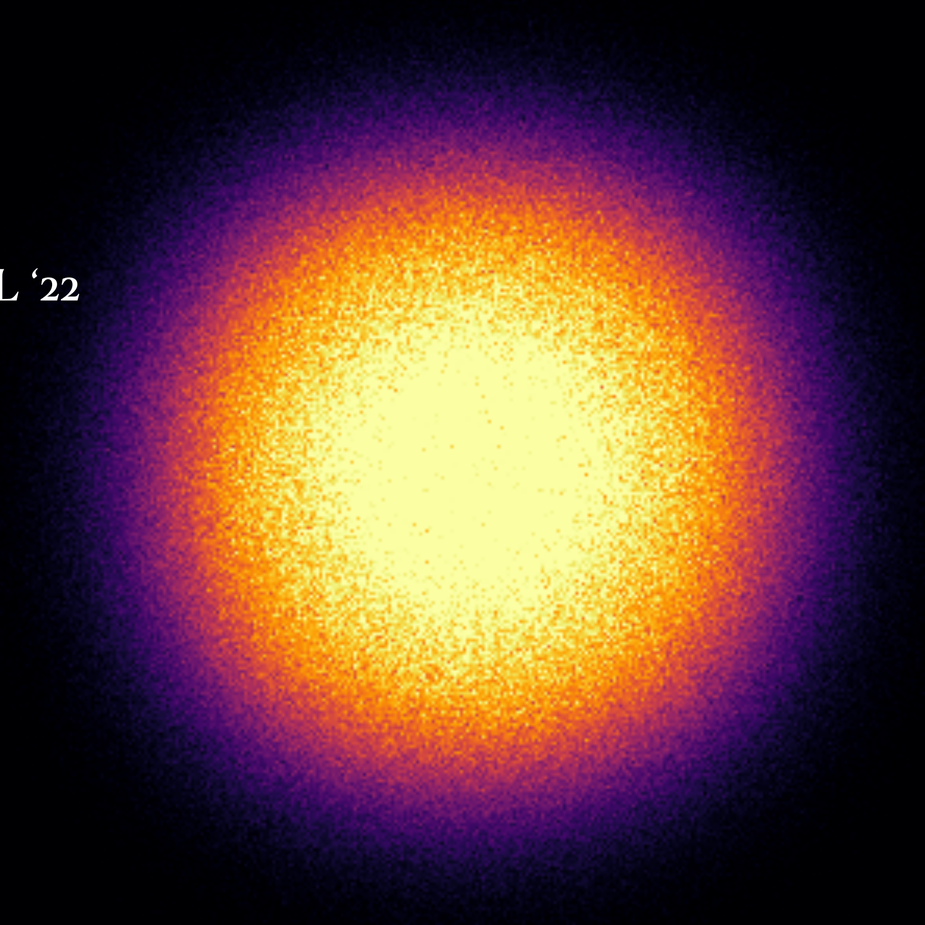


https://github.com/wangleiphy/ml4p/blob/main/projects/quantum_dot.ipynb

Fermi Flow

Xie, Zhang, LW, 2105.08644, JML ‘22

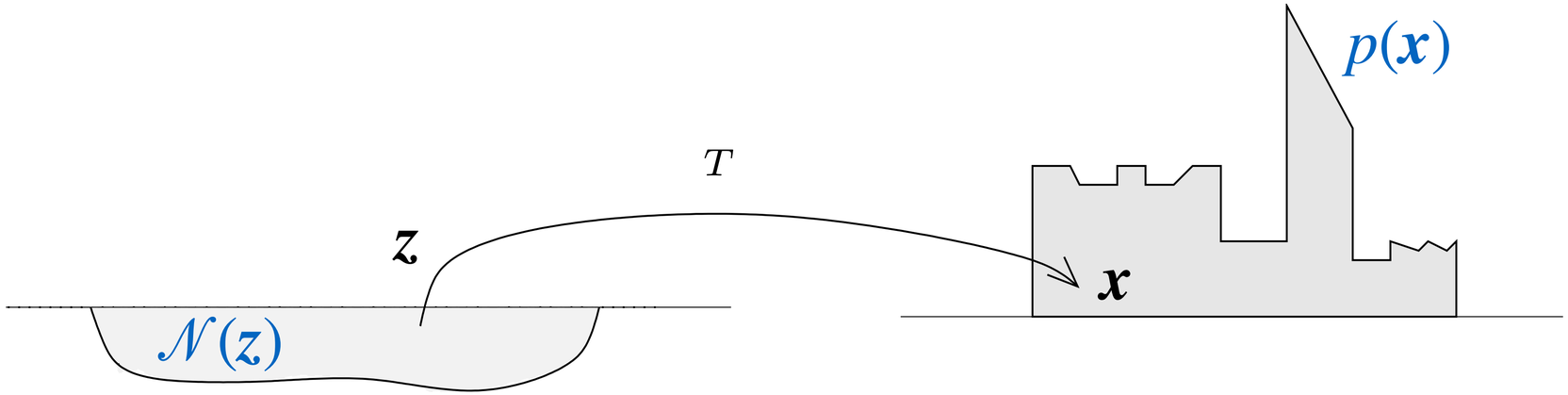
github.com/fermiflow



Continuous flow of electron density in a quantum dot

Optimal Transport Theory

Monge problem (1781): How to transport earth with optimal cost ?



Monge | Kantorovich | Koopmans | Dantzig | Brenier | Otto | McCann | Villani | Figalli

Nobel Prize in Economics ’75

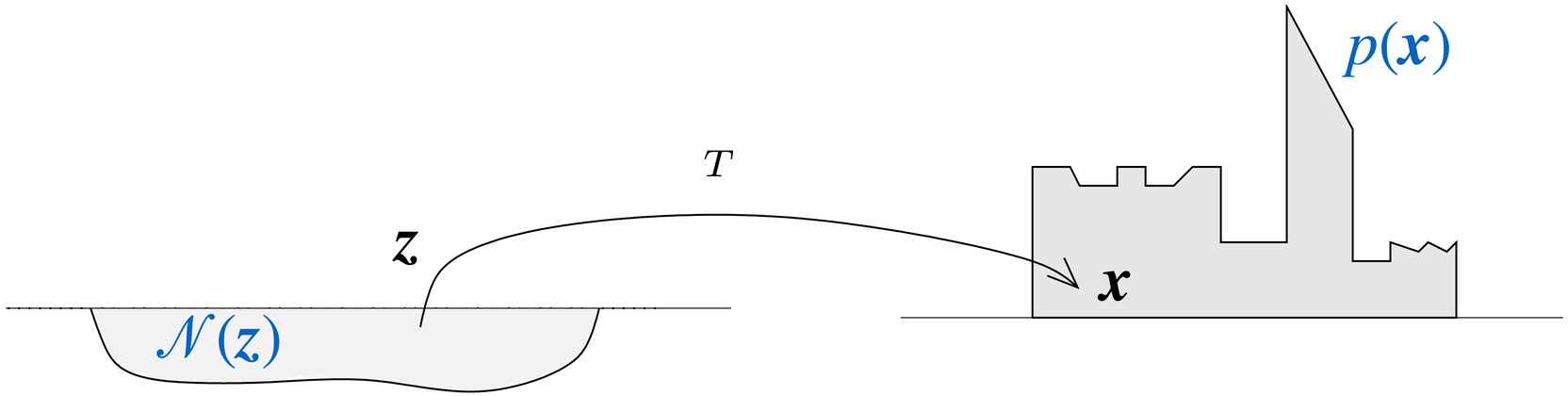
Fields Metal ’10

Fields Metal ’18

from Cuturi, Solomon NISP 2017 tutorial

Optimal Transport Theory

Monge problem (1781): How to transport earth with optimal cost ?



★ Brenier theorem (1991)

Under certain conditions
the optimal map is

$$z \mapsto x = \nabla u(z)$$

Monge-Ampère Equation

$$\frac{\mathcal{N}(z)}{p(\nabla u(z))} = \det\left(\frac{\partial^2 u}{\partial z_i \partial z_j}\right)$$

Monge-Ampère Flow

Zhang, E, LW 1809.10188

wangleiphy/MongeAmpereFlow

$$\frac{\partial \rho(\boldsymbol{x}, t)}{\partial t} + \nabla \cdot \left[\rho(\boldsymbol{x}, t)\, \nabla \varphi\right] = 0$$

1. Drive the flow with an "irrotational" velocity field

2. Impose symmetry to the scalar valued potential for symmetric generative model

$$\varphi(g\,\boldsymbol{x}) = \varphi(\boldsymbol{x}) \quad \Longrightarrow \quad \rho(g\,\boldsymbol{x}) = \rho(\boldsymbol{x})$$

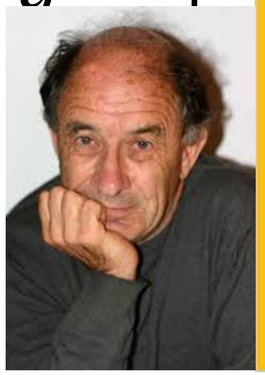
Hamiltonian dynamics: phase space flow

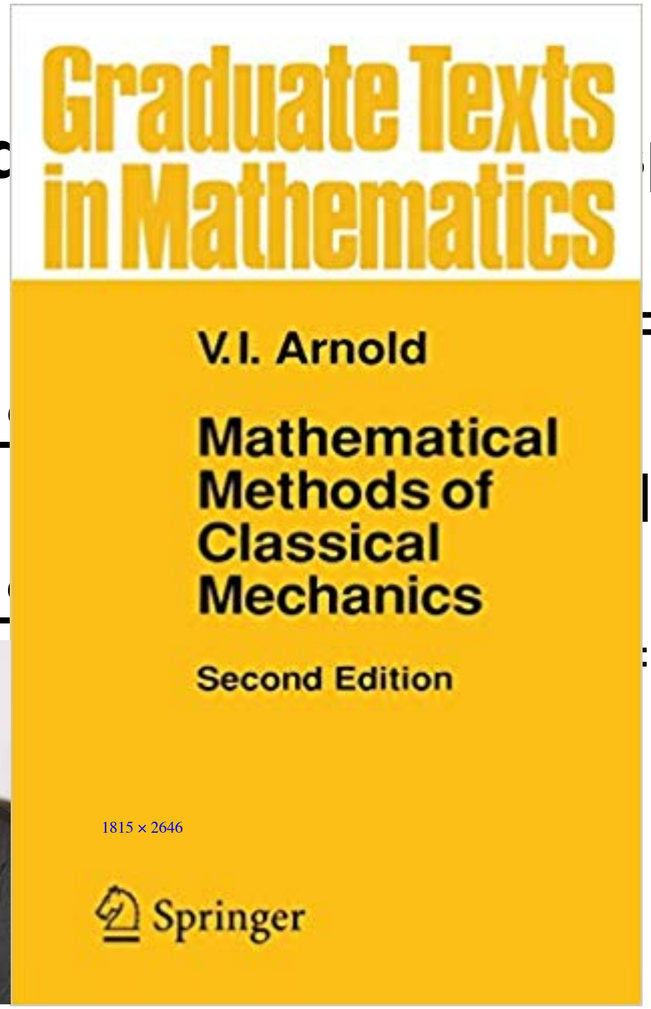
Hamiltonian ec **pace va** **ic gradient flow**

$= (p, q$ $\nabla_x H(x) J$

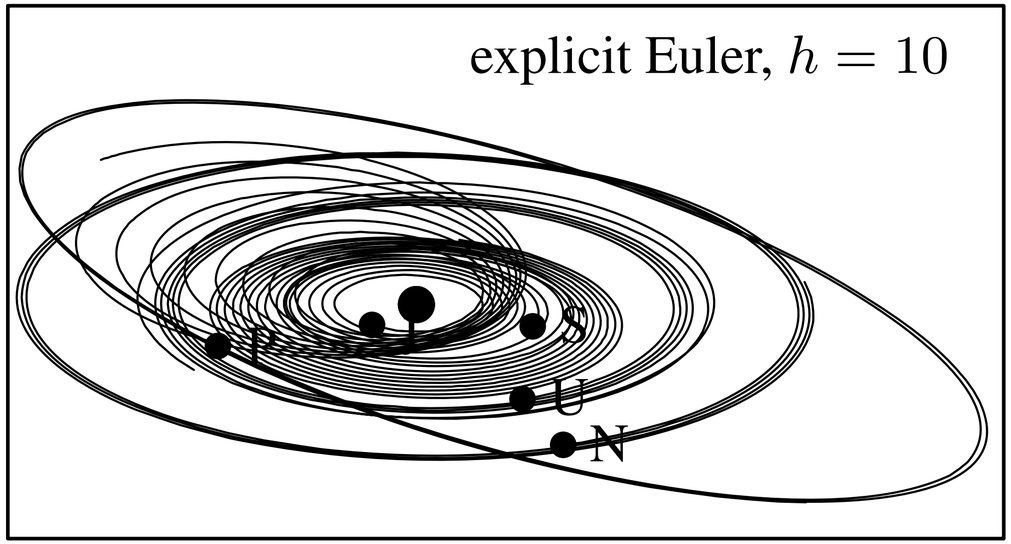
$$\begin{cases} \dot{p} = - \\ \dot{q} = + \end{cases}$$

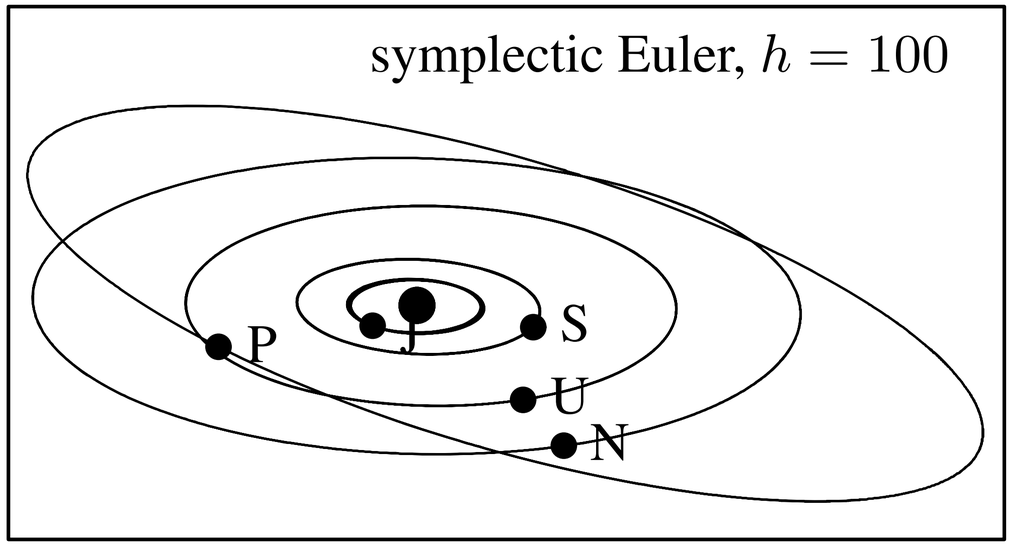
$\left(\begin{matrix} I \\ -I \end{matrix} \right.$





Symplectic Integrators





from Hairer et al, Geometric Numerical Integration

Canonical Transformations

$$\boldsymbol{x} = (p, q) \quad \xleftrightarrow{\text{Change of variables}} \quad z = (P, Q)$$

which satisfies $\left(\nabla_{\boldsymbol{x}} z\right) J \left(\nabla_{\boldsymbol{x}} z\right)^T = J$ symplectic condition

one has $\dot{z} = \nabla_z K(z) J$ where $K(z) = H \circ \boldsymbol{x}(z)$

Preserves Hamiltonian dynamics in the “latent phase space”

A probabilisitic perspective

Canonical transformation deforms phase space density $p(\boldsymbol{x}) = e^{-\beta H(\boldsymbol{x})}$

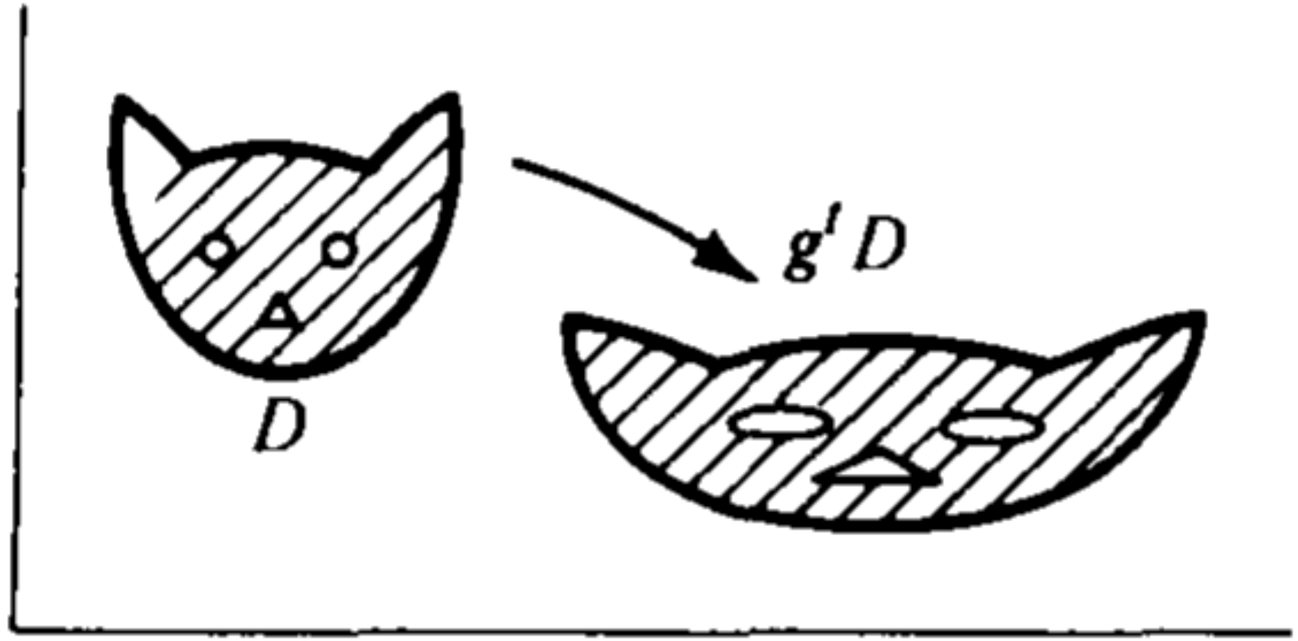


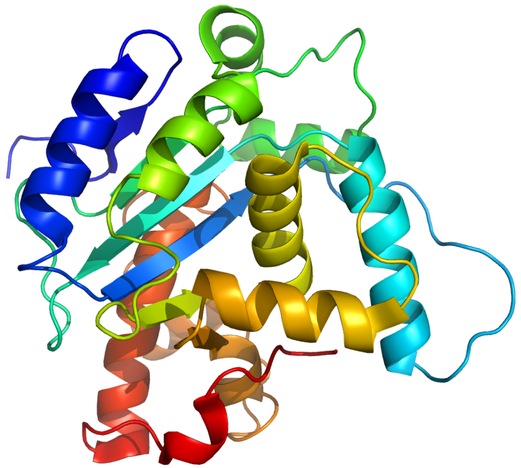
Image from Arnold '78

Use a symplectic flow to learn canonical transformations!

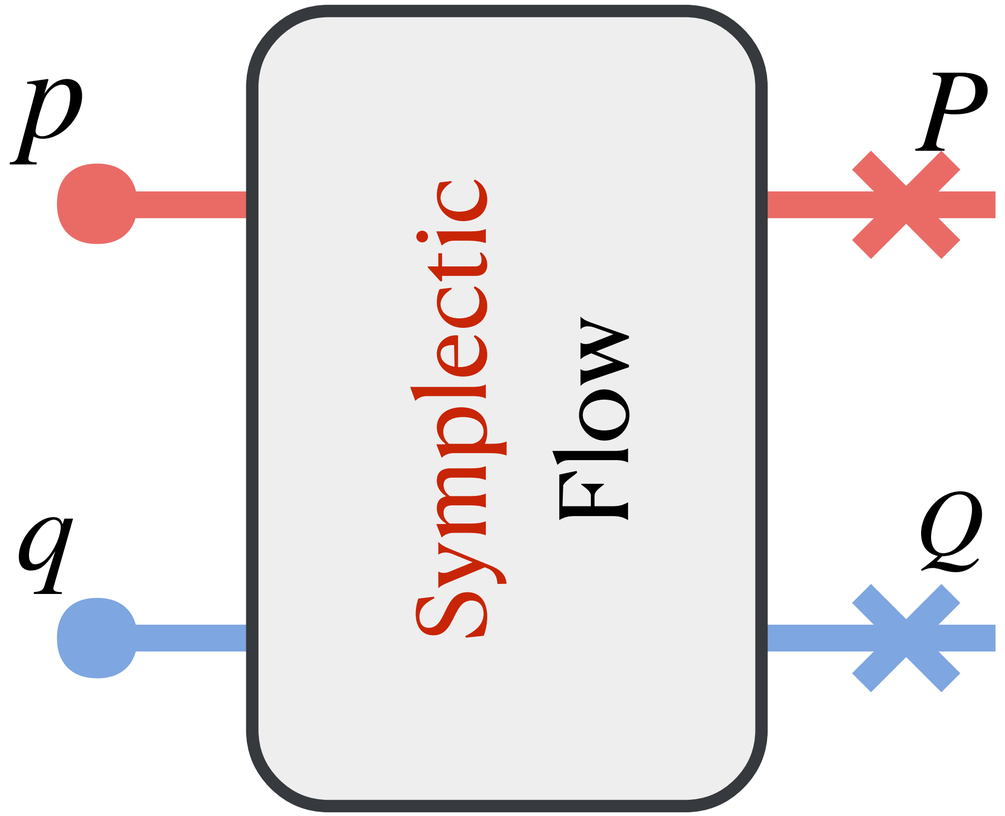
Neural Canonical Transformations

Li et al, PRX '20

$H(p, q)$



physical space



$$K(P, Q) = \sum_k \frac{P_k^2 + \omega_k^2 Q_k^2}{2}$$

latent space

Learn the network parameter and the latent harmonic frequency

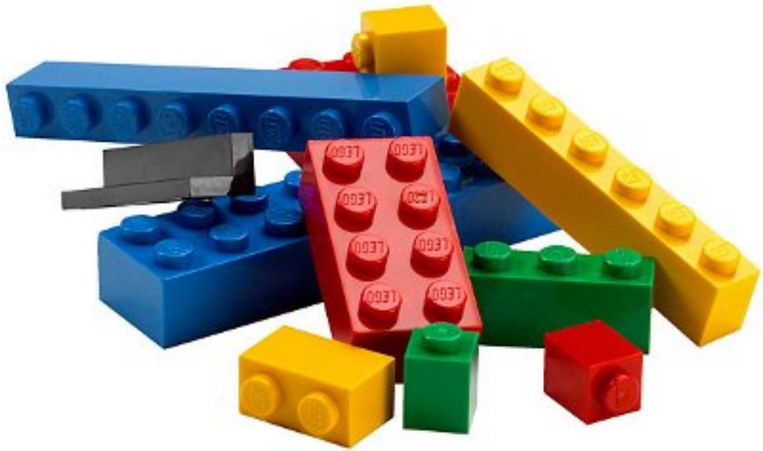
Modular design of the symplectic network

$$z = \mathcal{T}(x)$$

$$\mathcal{T} = \mathcal{T}_1 \circ \mathcal{T}_2 \circ \mathcal{T}_3 \circ \cdots$$

$$\left(\nabla_x z\right) J \left(\nabla_x z\right)^T = J$$

symplectic group



Compose symplectic primitives to form a deep neural network

Symplectic primitives

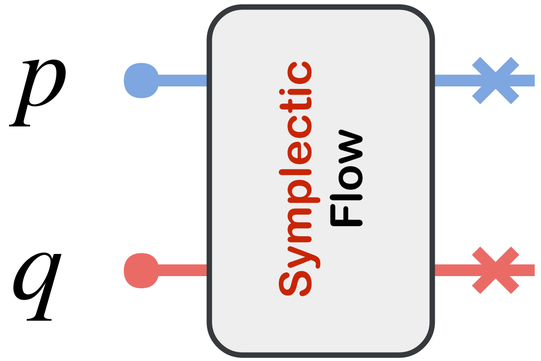
- Linear transformation: Symplectic Lie algebra

- Continuous-time flow: Symplectic generating functions

  Symplectic integrator as a “layer” in deep net

  Harbor el al 1705.03341
  Lu et al 1710.10121
  E, Commun. Math. Stat 17
  Chen et al 1806.07366
  Zhu et al, 2004.13830
  ...

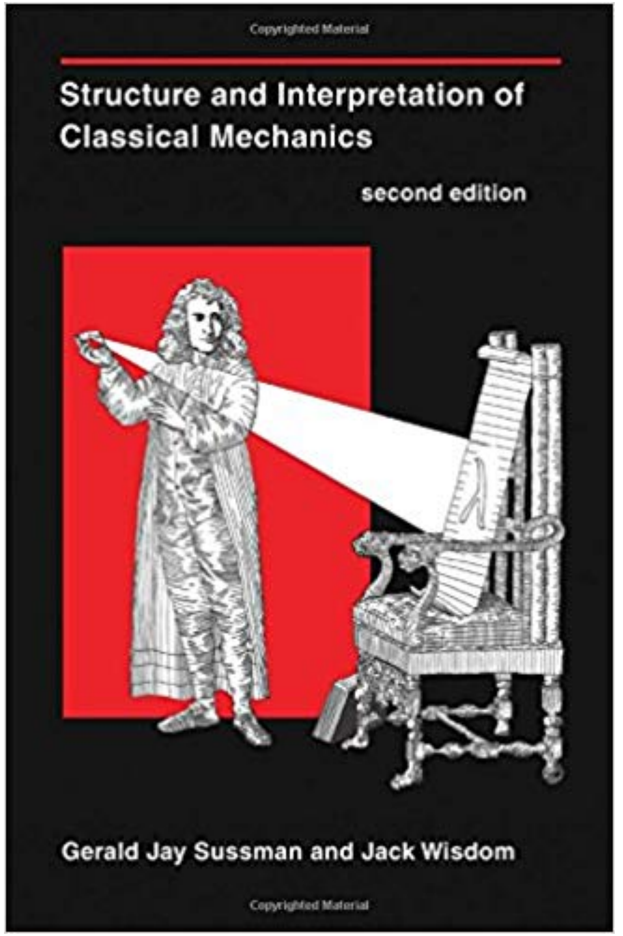
- **Neural point transformation**



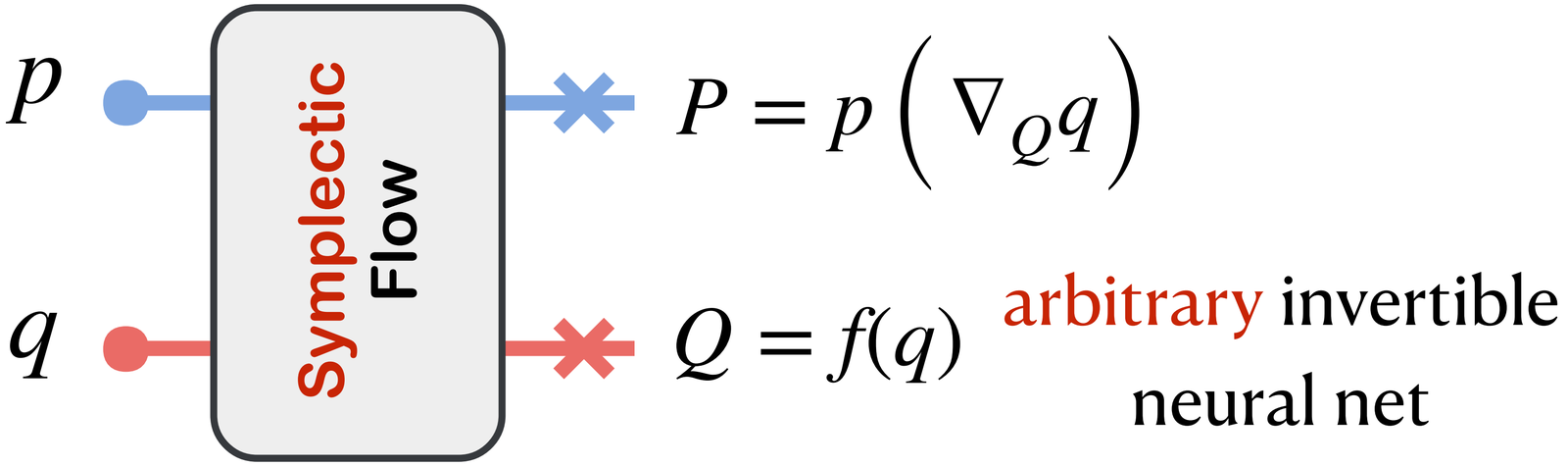
$$P = p\left(\nabla_Q q\right)$$

$$Q = f(q)$$

arbitrary invertible neural net



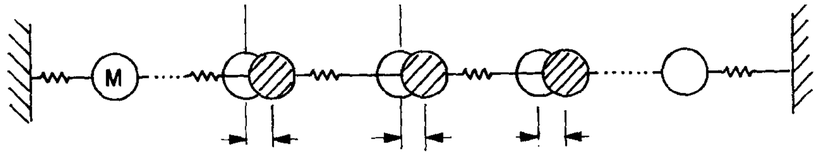
Neural point transformation

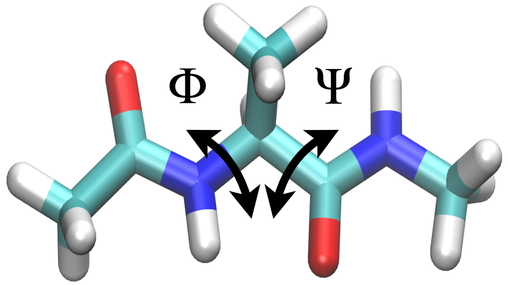



```
Q = forward(params, q)
_, vjp = jax.vjp(lambda Q: inverse(params, Q), Q)
P = vjp(p)[0]
```

How is this going to be useful ?

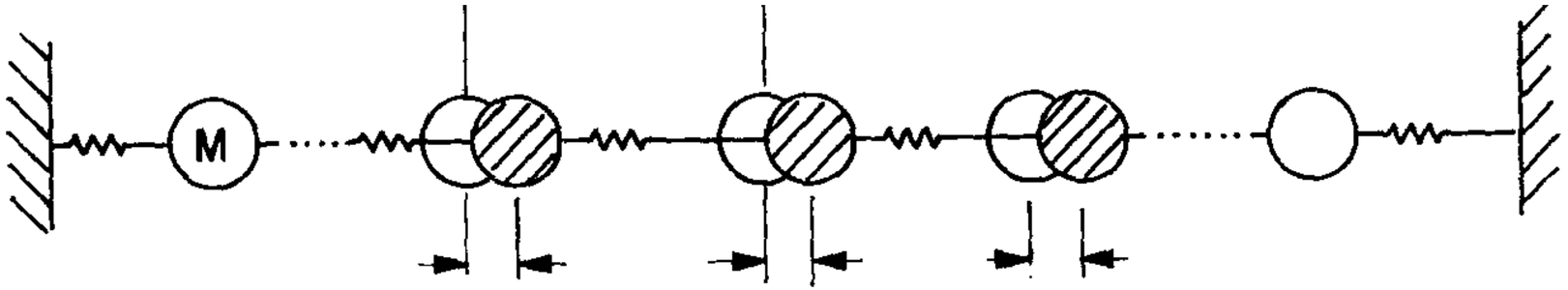
Let's play with examples!





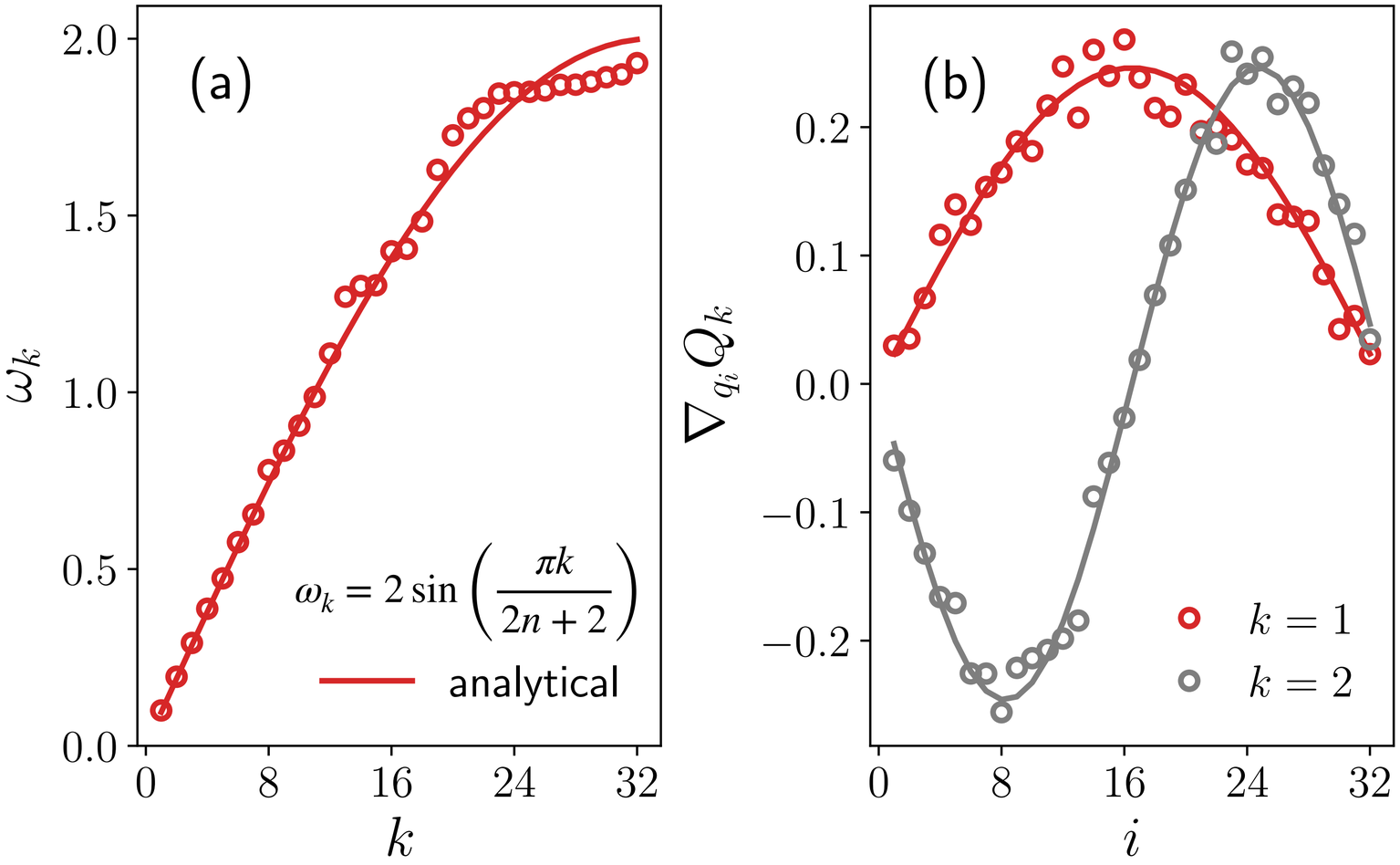
Warm up: Harmonic Chain

$$H = \frac{1}{2}\sum_{i=1}^{n}\left[p_i^2 + (q_i - q_{i-1})^2\right]$$



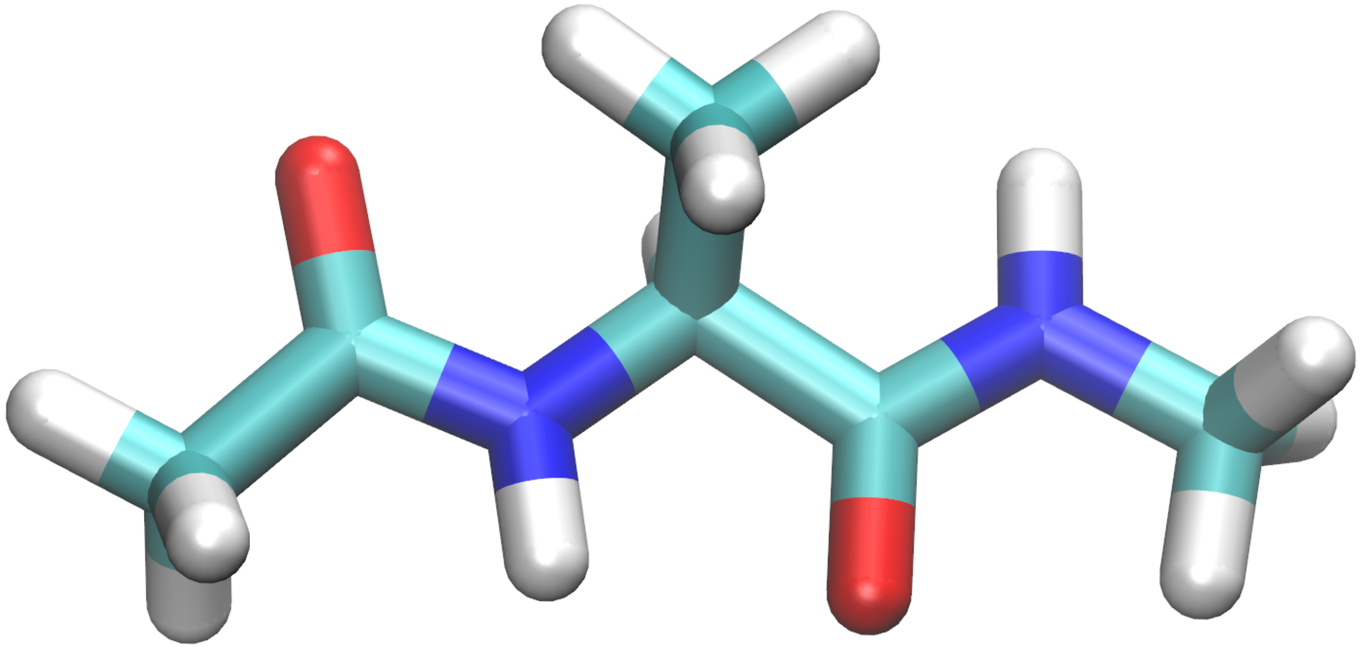
Fermi–Pasta–Ulam–Tsingou problem w/o nonlinearity

Learning the normal modes



Consistency check: neural nets can learn linear coordinate transformations

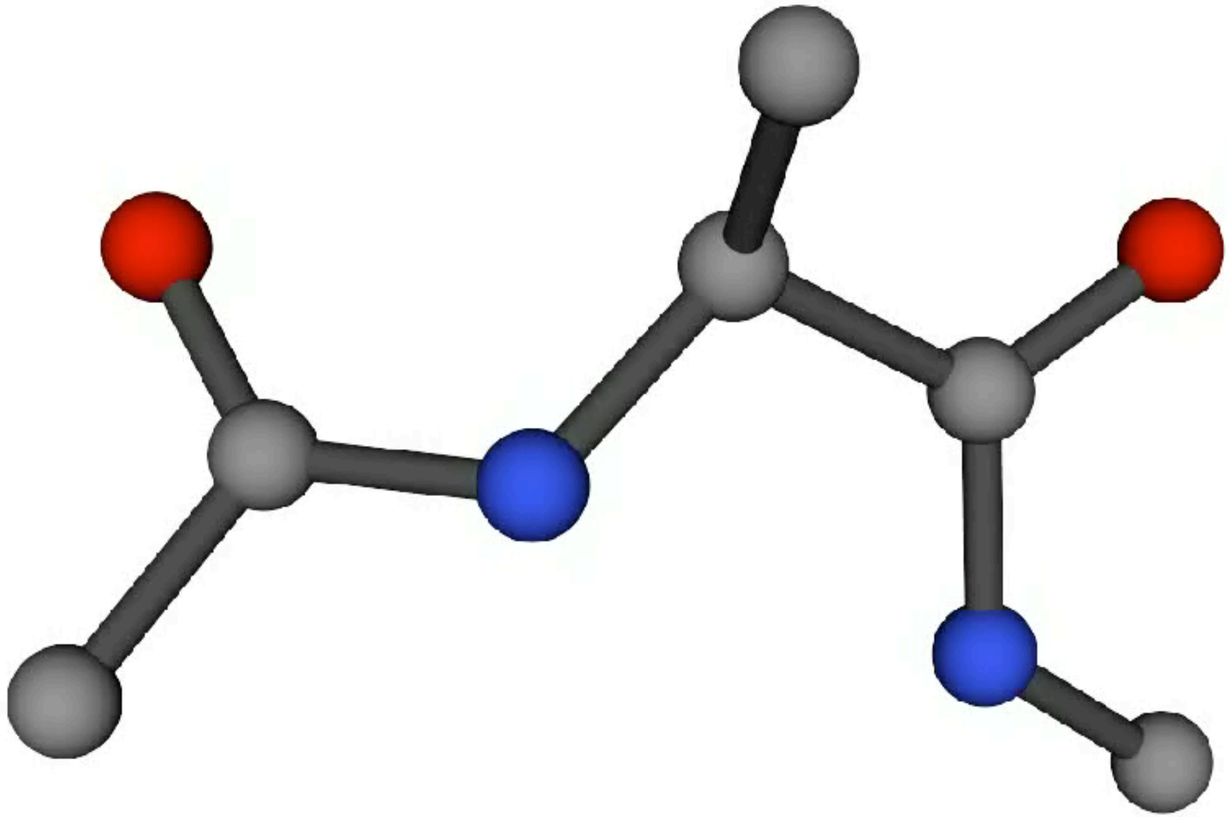
Alanine Dipeptide

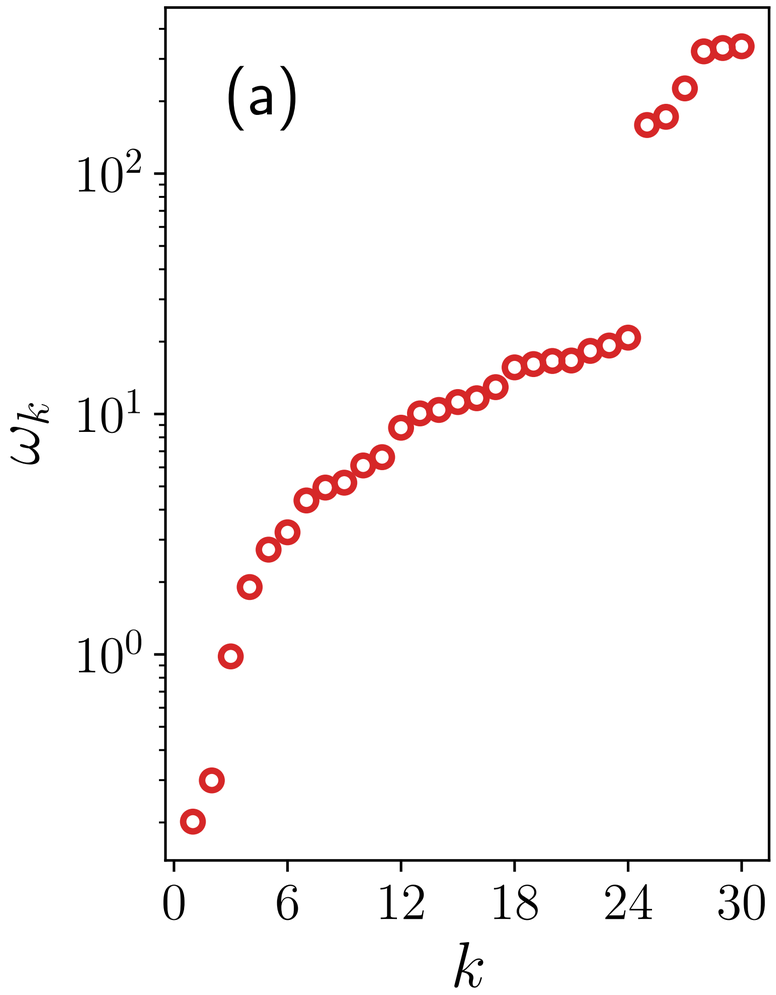


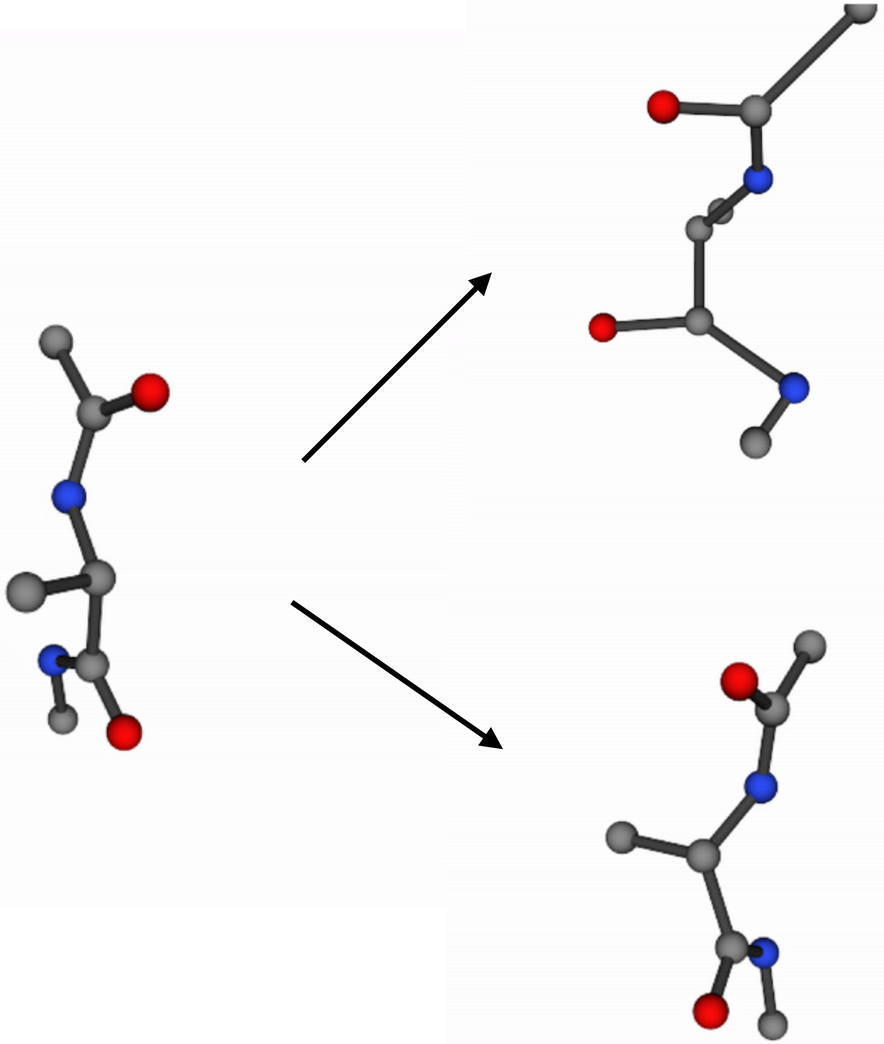
250 ns molecular dynamics simulation data at 300 K

https://markovmodel.github.io/mdshare/ALA2/#alanine-dipeptide

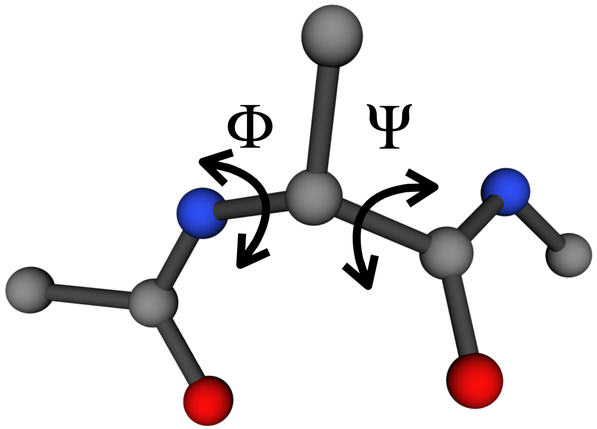
More than 3 hours of video ...



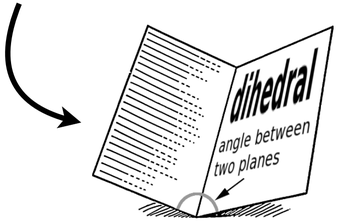


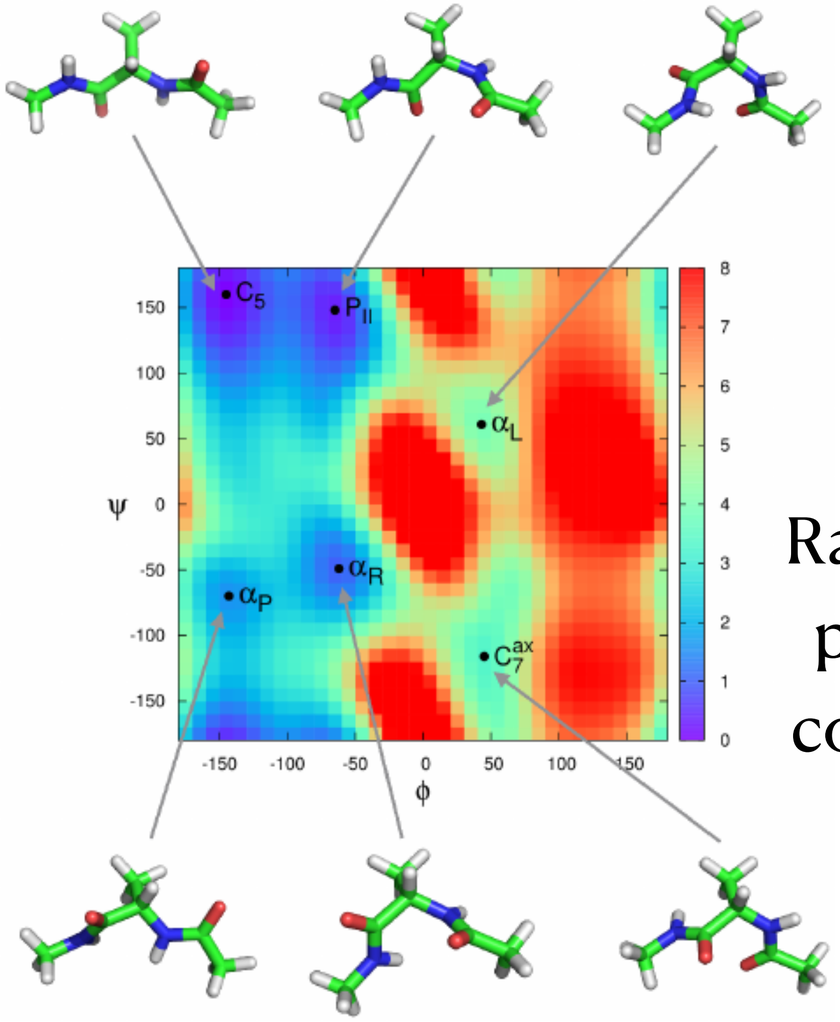


Neural canonical transformation identifies nonlinear slow modes!



slow motion of the two torsion angles





Ramachandran plot of stable conformations

Dimensional reduction to slow collective variables

useful for control, prediction, enhanced sampling...

check the paper 1910.00024, PRX '20 for more examples & applications

MNIST handwritten digits

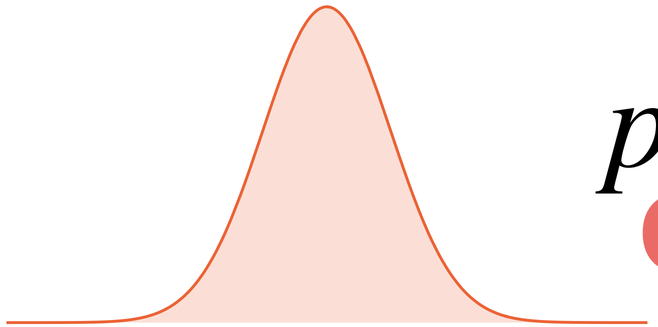
Data scientists:

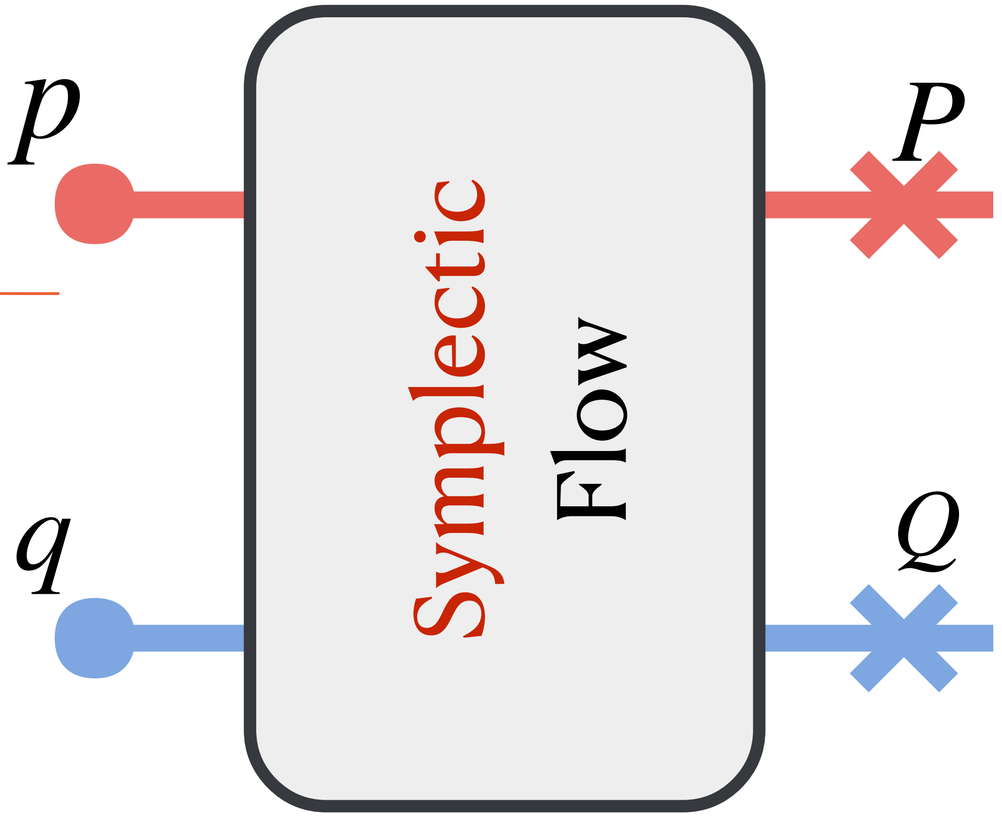
"50,000 grayscale images with 28x28 pixels"

Physical Chemists:

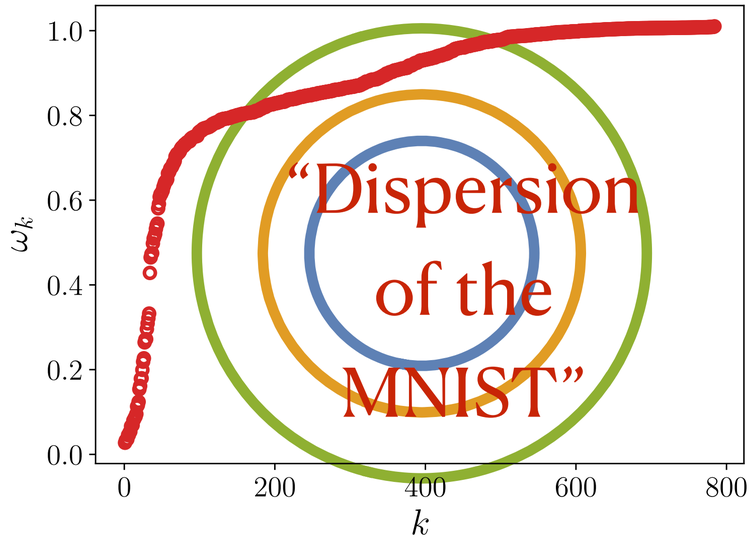
"Stable conformations of a molecule with 784 degrees of freedom"

Learning slow variables of MNIST

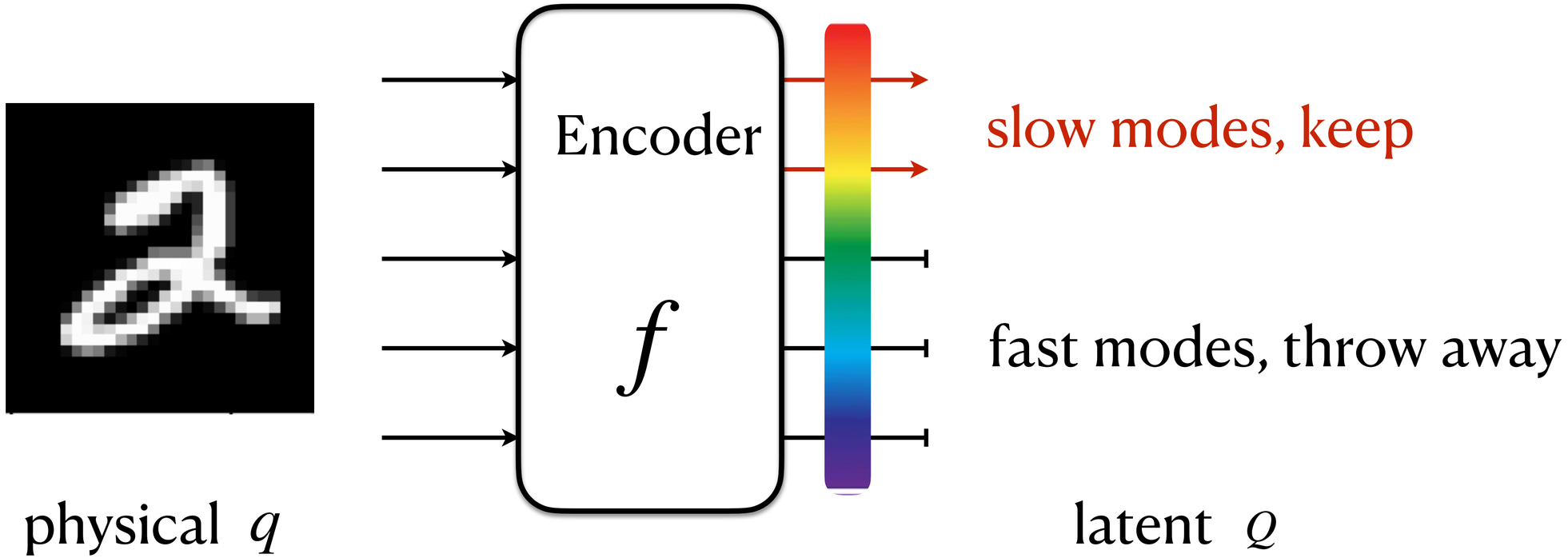




$$K(P,Q) = \sum_k \frac{P_k^2 + \omega_k^2 Q_k^2}{2}$$



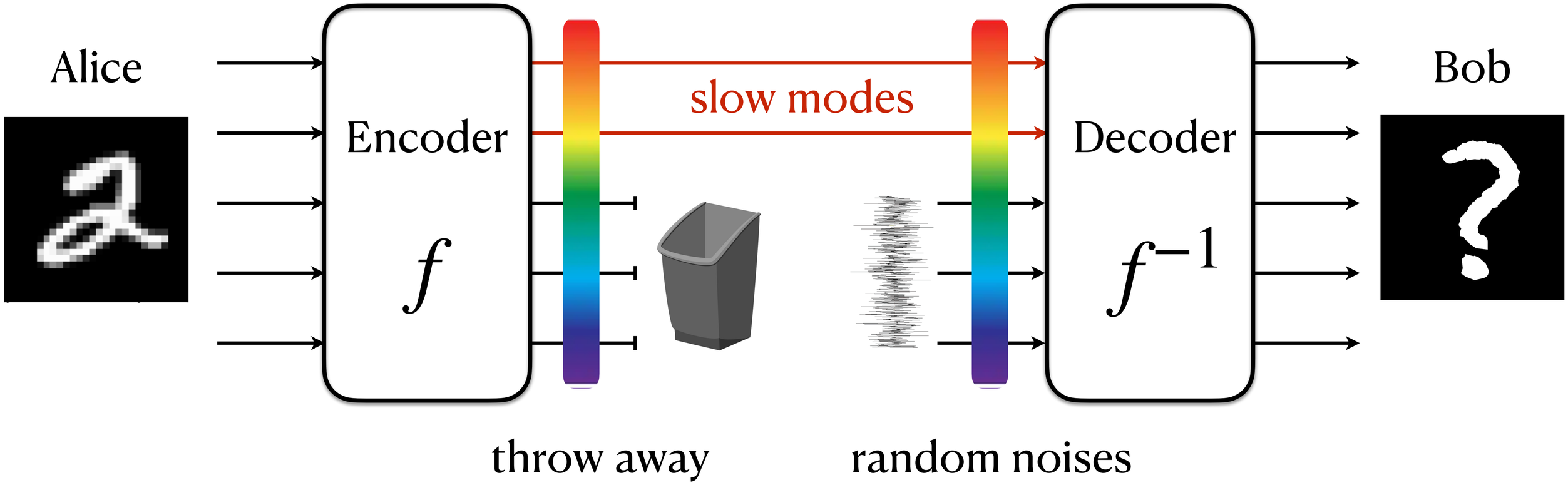
Conceptual compression



Compress by keeping slow collective variables

Kingma et al, 1312.6114 Gregor et al, 1604.08772 Dinh et al, 1605.08803
autoencoders/hierarchical network architecture/hyperbolic latent space...

Neural Alice-Bob game



Conceptual Compression of MNIST

Original

1/784 kept

Conceptual Compression of MNIST

Original

2/784 kept

Conceptual Compression of MNIST

Original

3/784 kept

Conceptual Compression of MNIST

Original

4/784 kept

Conceptual Compression of MNIST

Original

5/784 kept

Conceptual Compression of MNIST

Original

10/784 kept

Conceptual Compression of MNIST

Original

15/784 kept

Conceptual compression of MNIST

Original

20/784 kept

“A Hamiltonian Extravaganza”

—Danilo J. Rezende@DeepMind

Symplectic ODE-Net, 1909.12077

Hamiltonian Graph Networks with ODE Integrators, 1909.12790

Symplectic RNN, 1909.13334

Equivariant Hamiltonian Flows, 1909.13739

Hamiltonian Generative Network, 1909.13789 http://tiny.cc/hgn

Neural Canonical Transformation with Symplectic Flows, 1910.00024

See also Bondesan & Lamacraft, *Learning Symmetries of Classical Integrable Systems*,1906.04645

Training: Monte Carlo Gradient Estimators

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p_{\boldsymbol{\theta}}} \left[f(\boldsymbol{x}) \right]$$

Review: 1906.10652

Reinforcement learning
Variational inference
Variational Monte Carlo
Variational quantum algorithms
...

Score function estimator (REINFORCE)

$$\nabla_\theta \mathbb{E}_{\boldsymbol{x} \sim p_\theta} \left[f(\boldsymbol{x}) \right] = \mathbb{E}_{\boldsymbol{x} \sim p_\theta} \left[f(\boldsymbol{x}) \nabla_\theta \ln p_\theta(\boldsymbol{x}) \right]$$

Pathwise estimator (Reparametrization trick) $\boldsymbol{x} = g_\theta(\boldsymbol{z})$

$$\nabla_\theta \mathbb{E}_{\boldsymbol{x} \sim p_\theta} \left[f(\boldsymbol{x}) \right] = \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{z})} \left[\nabla_\theta f(g_\theta(\boldsymbol{z})) \right]$$

10.1 Guidance in Choosing Gradient Estimators

With so many competing approaches, we offer our rules of thumb in choosing an estimator, which follow the intuition we developed throughout the paper:

- If our estimation problem involves continuous functions and measures that are continuous in the domain, then using the pathwise estimator is a good default. It is relatively easy to implement and a default implementation, one without other variance reduction, will typically have variance that is low enough so as not to interfere with the optimisation.
- If the cost function is not differentiable or a black-box function then the score-function or the measure-valued gradients are available. If the number of parameters is low, then the measure-valued gradient will typically have lower variance and would be preferred. But if we have a high-dimensional parameter set, then the score function estimator should be used.
- If we have no control over the number of times we can evaluate a black-box cost function, effectively only allowing a single evaluation of it, then the score function is the only estimator of the three we reviewed that is applicable.
- The score function estimator should, by default, always be implemented with at least a basic variance reduction. The simplest option is to use a baseline control variate estimated with a running average of the cost value.
- When using the score-function estimator, some attention should be paid to the dynamic range of the cost function and its variance, and to find ways to keep its value bounded within a reasonable range, e.g., transforming the cost so that it is zero mean, or using a baseline.
- For all estimators, track the variance of the gradients if possible and address high variance by using a larger number of samples from the measure, decreasing the learning rate, or clipping the gradient values. It may also be useful to restrict the range of some parameters to avoid extreme values, e.g., by clipping them to a desired interval.
- The measure-valued gradient should be used with some coupling method for variance reduction. Coupling strategies that exploit relationships between the positive and negative components of the density decomposition, and which have shared sampling paths, are known for the commonly-used distributions.
- If we have several unbiased gradient estimators, a convex combination of them might have lower variance than any of the individual estimators.
- If the measure is discrete on its domain then the score-function or measure-valued gradient are available. The choice will again depend on the dimensionality of the parameter space.
- In all cases, we strongly recommend having a broad set of tests to verify the unbiasedness of the gradient estimator when implemented.

Mohamed et al, 1906.10652

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x} \sim p_{\boldsymbol{\theta}}} \left[f(\boldsymbol{x}) \right]$$

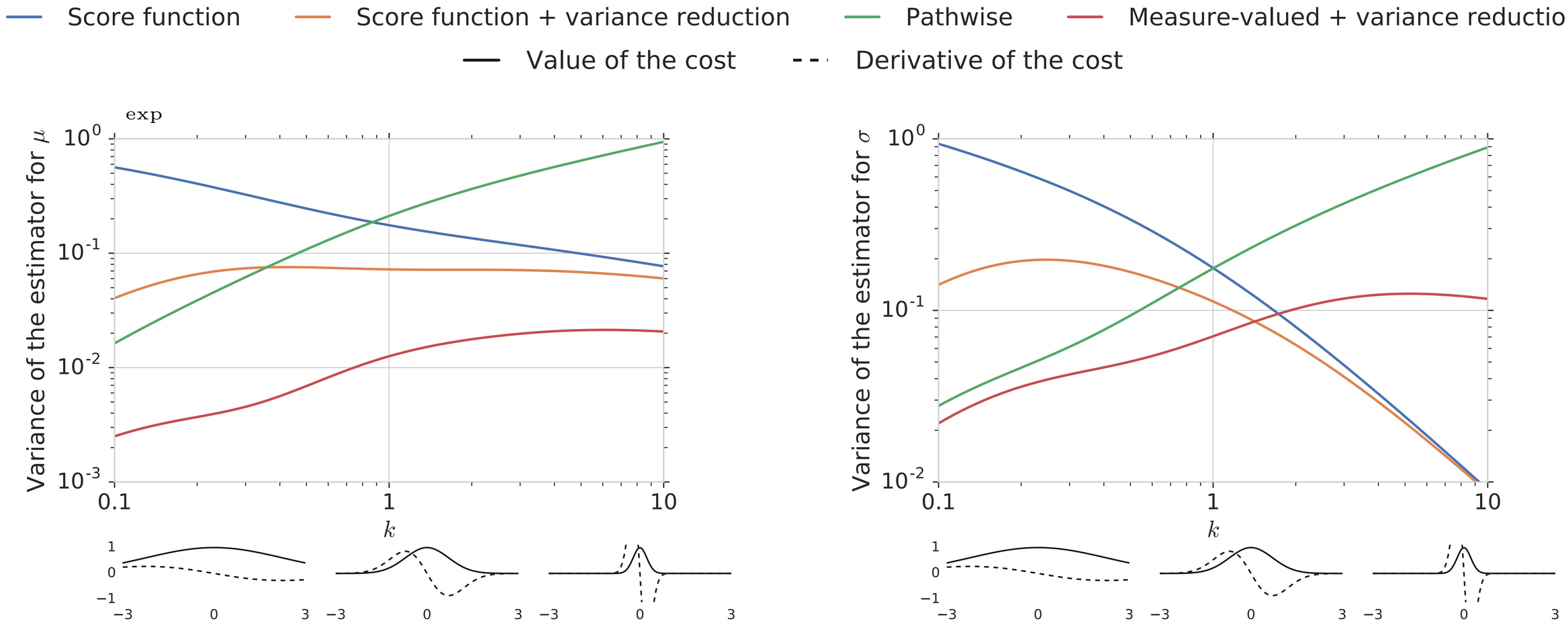
When to use which ?

More discussions

Roeder et al, 1703.09194

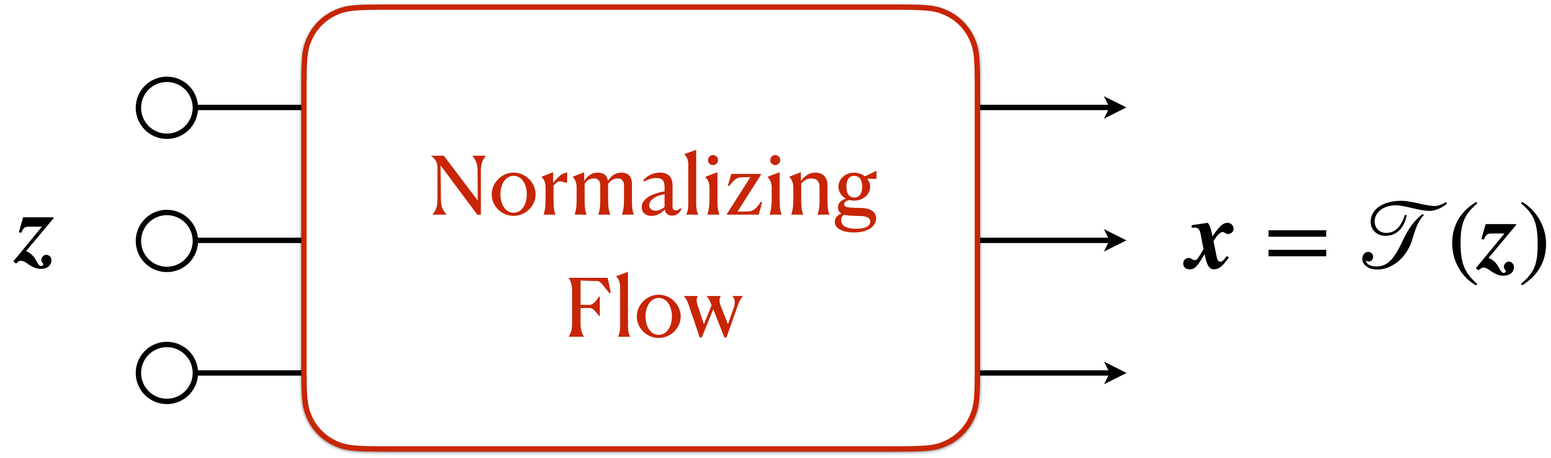
Vaitl et al 2206.09016, 2207.08219

$$\eta = \nabla_\theta \int \mathcal{N}(x|\mu, \sigma^2) f(x; k) dx; \quad \theta \in \{\mu, \sigma\}$$



https://github.com/deepmind/mc_gradients Mohamed et al, 1906.10652

Symmetries



Invariance

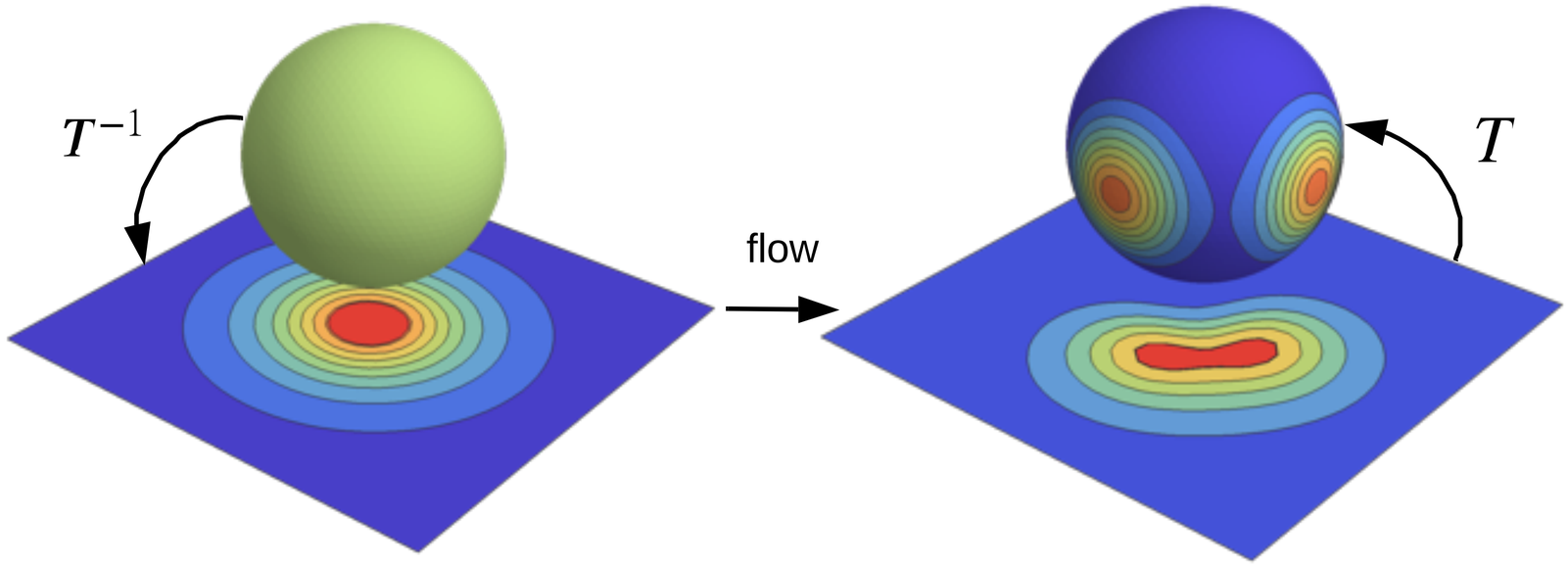
$$\rho(g\,\boldsymbol{x}) = \rho(\boldsymbol{x})$$

Equivariance

$$\mathcal{T}(g\,\boldsymbol{z}) = g\mathcal{T}(\boldsymbol{z})$$

Spatial symmetries, permutation symmetries, gauge symmetries...

Flow on manifolds

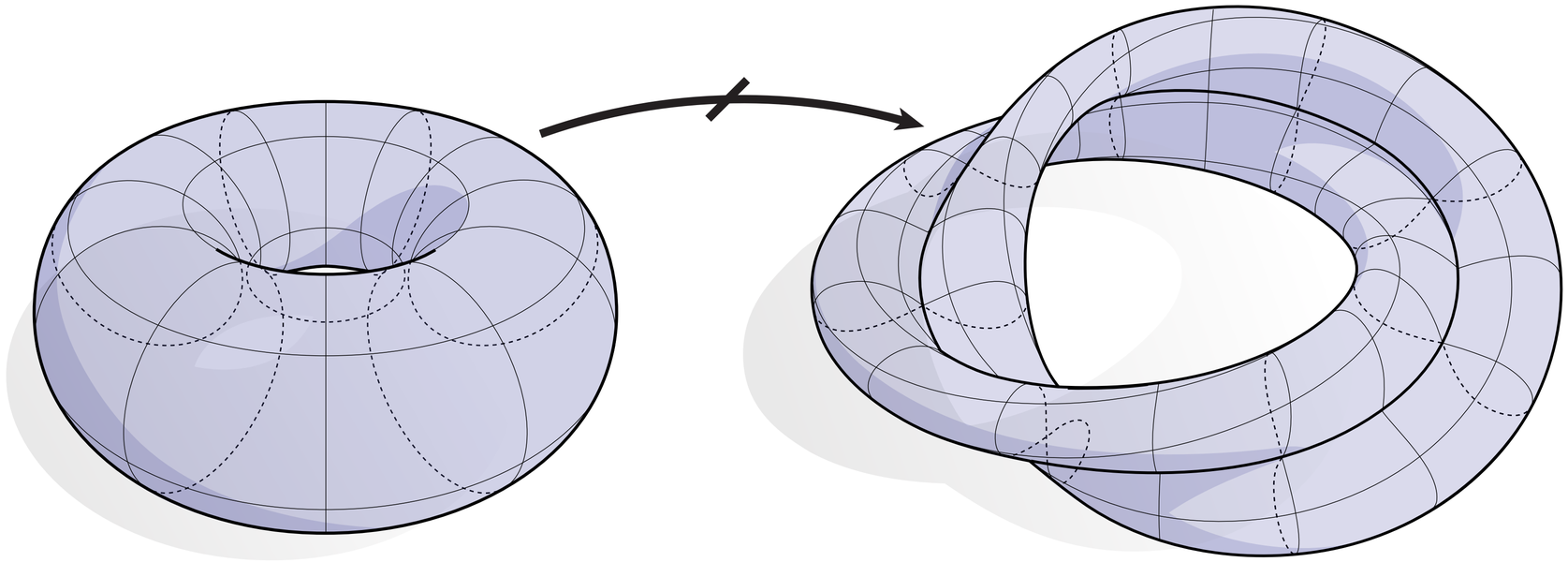


Periodic variables, gauge fields, ...

Gemici et al 1611.02304, Rezende et al, 2002.02428, Boyda et al, 2008.05456

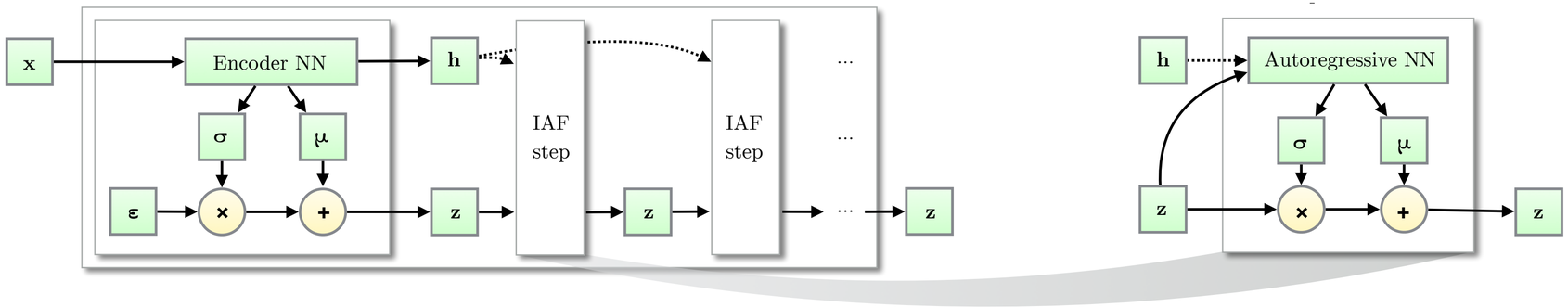
Neural ODE on manifolds, Falorsi et al, 2006.06663, Lou et al, 2006.10254, Mathieu et al, 2006.10605

Obstructions

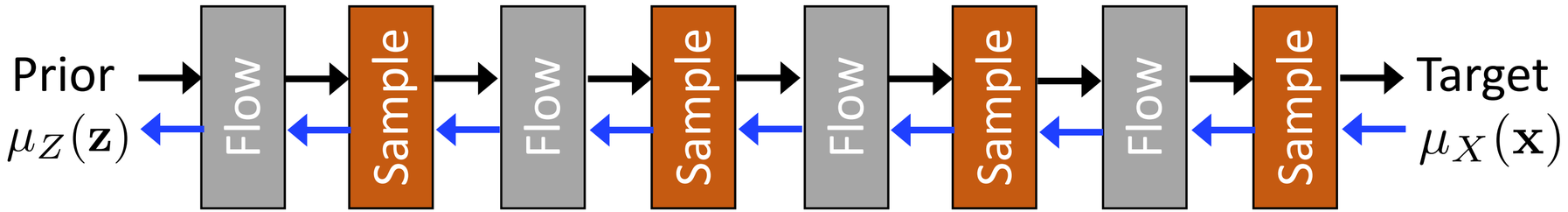


Dupont et al 1904.01681, Cornish et al, 1909.13833, Zhang et al, 1907.12998, Zhong et al, 2006.00392...

Mix with other approaches



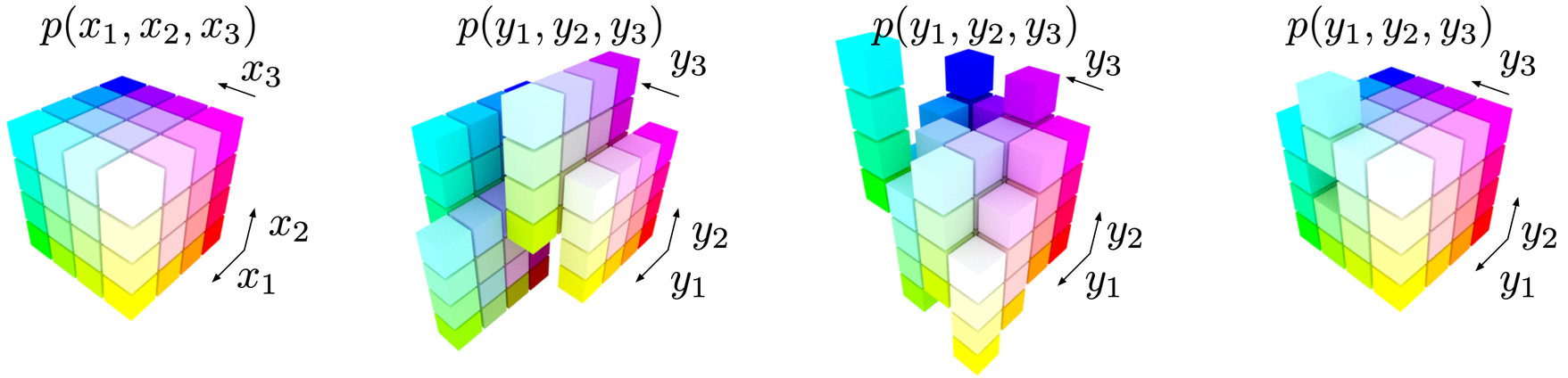
Kingma et al, 1606.04934,...



Levy et al, 1711.09268, Wu et al 2002.06707, ...

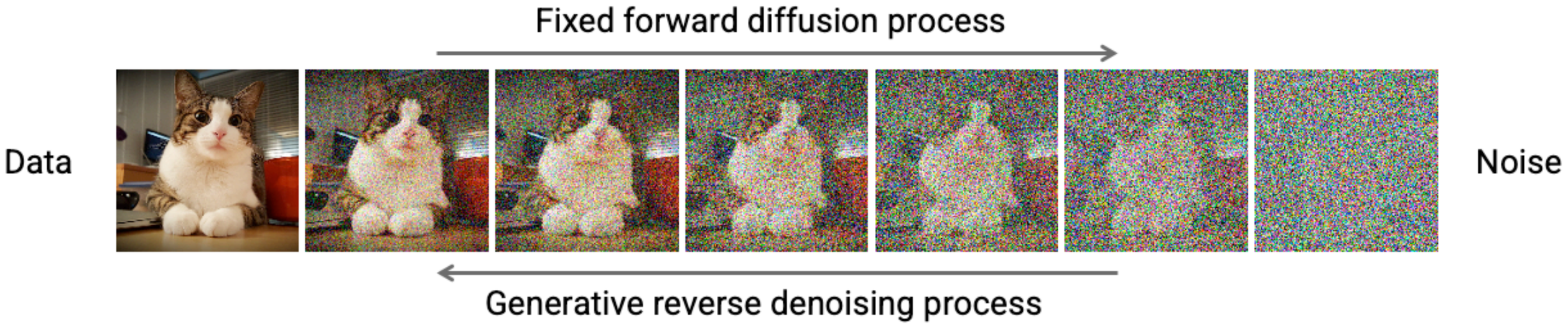
Discrete flows

$$p(\boldsymbol{x}) = p(\boldsymbol{y} = \mathcal{T}(\boldsymbol{x}))$$



Tran et al, 1905.10347, Hoogeboom et al, 1905.07376, van den Berg 2006.12459

Diffusion models



https://cvpr2022-tutorial-diffusion-models.github.io

Score matching

Minimizing Fisher divergence by-passes intractable partition function

$$\mathbb{F}(\pi \parallel p) = \int d\boldsymbol{x}\, \pi(\boldsymbol{x}) \left| \nabla \ln \pi(\boldsymbol{x}) - \nabla \ln p(\boldsymbol{x}) \right|^2$$

Draw samples with Langevin dynamics

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \frac{\epsilon}{2} \nabla \ln p(\boldsymbol{x}_{t-1}) + \sqrt{\epsilon}\, \mathcal{N}(0, I)$$

$$\boldsymbol{x}_T \sim p(\boldsymbol{x}_T) \quad \text{for} \quad \epsilon \to 0, \quad T \to \infty$$

From score matching to diffusion model

Denoising score matching Vincent 2011

$$\frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})p_{\text{data}}(\mathbf{x})}[\|\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}}\log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2].$$

Diffusion generative model

Sohl-Dickstein et al, 1503.03585

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right]$$

Denoising diffusion probabilistic model

Song et al, 1907.05600, Ho et al, 2006.11239

$$\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{t\sim\mathcal{U}(0,T)}}_{\text{diffusion time } t}\underbrace{\mathbb{E}_{\mathbf{x}_0\sim q_0(\mathbf{x}_0)}}_{\text{data sample } \mathbf{x}_0}\underbrace{\mathbb{E}_{\mathbf{x}_t\sim q_t(\mathbf{x}_t|\mathbf{x}_0)}}_{\text{diffused data sample } \mathbf{x}_t}||\underbrace{\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}_{\text{neural network}} - \underbrace{\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t|\mathbf{x}_0)}_{\text{score of diffused data sample}}||_2^2$$

A tale of three equations

Langevin equation (SDE)

$$q(\boldsymbol{x}_{t+dt}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_t + \boldsymbol{f}dt, 2Tdt\boldsymbol{I}) \qquad \text{or} \quad \boldsymbol{x}_{t+dt} - \boldsymbol{x}_t = \boldsymbol{f}dt + \sqrt{2Tdt}\,\mathcal{N}(0,1)$$

Fokker-Planck equation (PDE)

$$\frac{\partial p(\boldsymbol{x},t)}{\partial t} + \nabla \cdot \left[p(\boldsymbol{x},t)\boldsymbol{f}\right] - T\nabla^2 p(\boldsymbol{x},t) = 0$$

"Particle method" (ODE)

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f} - T\nabla \ln p(\boldsymbol{x},t)$$

(Another way to reverse the diffusion is via the reverse-time SDE Anderson 1982)

Maoutsa et al, 2006.00702
Song et al, 2011.13456

$$\mathcal{P}(\vec{x},t)=\int \mathrm{d}^3\vec{x}\,'\left(\frac{1}{4\pi D\epsilon}\right)^{3/2}\exp\left[-\frac{\left(\vec{x}-\vec{x}\,'-\epsilon\vec{v}(\vec{x}\,')\right)^2}{4D\epsilon}\right]\mathcal{P}(\vec{x}\,',t-\epsilon), \tag{9.18}$$

and simplified by the change of variables,

$$\begin{aligned}\vec{y}\ &=\vec{x}\,'+\epsilon\vec{v}(\vec{x}\,')-\vec{x}\quad\Longrightarrow\\ \mathrm{d}^3\vec{y}\ &=\mathrm{d}^3\vec{x}\,'\left(1+\epsilon\nabla\cdot\vec{v}(\vec{x}')\right)=\mathrm{d}^3\vec{x}\,'\left(1+\epsilon\nabla\cdot\vec{v}(\vec{x})+\mathcal{O}(\epsilon^2)\right).\end{aligned} \tag{9.19}$$

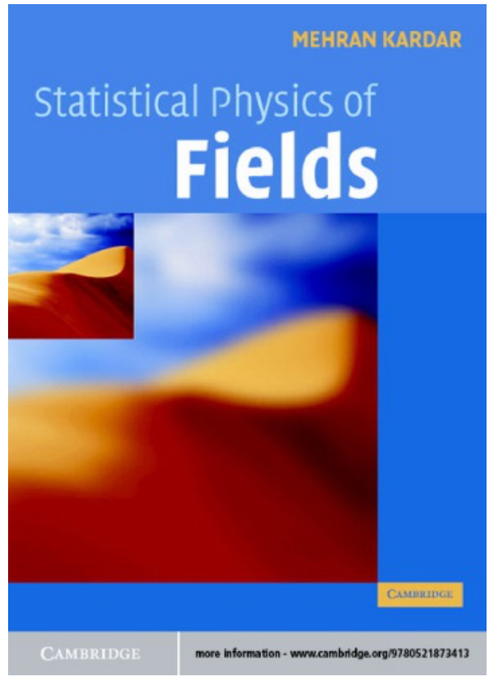
Keeping only terms at order of ϵ, we obtain

$$\begin{aligned}\mathcal{P}(\vec{x},t)&=\left[1-\epsilon\nabla\cdot\vec{v}(\vec{x})\right]\int \mathrm{d}^3\vec{y}\left(\frac{1}{4\pi D\epsilon}\right)^{3/2}\mathrm{e}^{-\frac{y^2}{4D\epsilon}}\ \mathcal{P}(\vec{x}+\vec{y}-\epsilon\vec{v}(\vec{x}),t-\epsilon)\\ &=\left[1-\epsilon\nabla\cdot\vec{v}(\vec{x})\right]\ \int \mathrm{d}^3\vec{y}\left(\frac{1}{4\pi D\epsilon}\right)^{3/2}\mathrm{e}^{-\frac{y^2}{4D\epsilon}}\\ &\times\left[\mathcal{P}(\vec{x},t)+(\vec{y}-\epsilon\vec{v}(\vec{x}))\cdot\nabla\mathcal{P}+\frac{y_iy_j-2\epsilon y_iv_j+\epsilon^2v_iv_j}{2}\nabla_i\nabla_j\mathcal{P}-\epsilon\frac{\partial\mathcal{P}}{\partial t}+\mathcal{O}(\epsilon^2)\right]\\ &=\left[1-\epsilon\nabla\cdot\vec{v}(\vec{x})\right]\left[\mathcal{P}-\epsilon\vec{v}\cdot\nabla+\epsilon D\nabla^2\mathcal{P}-\epsilon\frac{\partial\mathcal{P}}{\partial t}+\mathcal{O}(\epsilon^2)\right].\end{aligned} \tag{9.20}$$

Equating terms at order of ϵ leads to the *Fokker–Planck equation*,

$$\frac{\partial\mathcal{P}}{\partial t}+\nabla\cdot\vec{J}=0,\quad\text{with}\quad\vec{J}=\vec{v}\,\mathcal{P}-D\nabla\mathcal{P}. \tag{9.21}$$

from Langevin to Fokker-Planck



Lessons from diffusion models

Continuous normalizing flow has great potential: diffusion model is an “existence proof”

Going beyond maximum likelihood estimation training (even if we can)

Break the loss into small pieces, sample them (kind of layer-wise training)

https://blog.alexalemi.com/diffusion.html

The conditional trick (originated from denoising score matching Vincent 2011)

$$\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{t\sim\mathcal{U}(0,T)}}_{\text{diffusion time } t} \underbrace{\mathbb{E}_{\mathbf{x}_0\sim q_0(\mathbf{x}_0)}}_{\text{data sample } \mathbf{x}_0} \underbrace{\mathbb{E}_{\mathbf{x}_t\sim q_t(\mathbf{x}_t|\mathbf{x}_0)}}_{\text{diffused data sample } \mathbf{x}_t} || \underbrace{\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}_{\text{neural network}} - \underbrace{\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t|\mathbf{x}_0)}_{\text{score of diffused data sample}} ||_2^2$$

https://cvpr2022-tutorial-diffusion-models.github.io/

Claim:

$$\mathbb{E}_{x\sim q(x)}\,|\,s_\theta(x)-\nabla_x \ln q(x)\,|^2=\mathbb{E}_{x_0\sim q_0(x_0)}\mathbb{E}_{x\sim q(x|x_0)}\,|\,s_\theta(x)-\nabla_x \ln q(x\,|\,x_0)\,|^2+\text{const}.$$

↓

Independent of θ

$$q(x)=\int q(x\,|\,x_0)q_0(x_0)dx_0$$

Proof:

$$\mathbb{E}_{x_0\sim q_0(x_0)}\mathbb{E}_{x\sim q(x|x_0)}\,|\,s\,|^2=\int dx_0\int dx q_0(x_0)q(x\,|\,x_0)\,|\,s\,|^2=\int dx q(x)\,|\,s\,|^2=\mathbb{E}_{x\sim q(x)}\,|\,s^2\,|$$

$$\begin{aligned}\mathbb{E}_{x_0\sim q_0(x_0)}\mathbb{E}_{x\sim q(x|x_0)}[s\cdot\nabla \ln q(x\,|\,x_0)]&=\int dx_0\int dx q_0(x_0)q(x\,|\,x_0)\frac{s\cdot\nabla q(x\,|\,x_0)}{q(x\,|\,x_0)}\\&=\int dx_0\int dx q_0(x_0)s\cdot\nabla q(x\,|\,x_0)\\&=\int dx s\cdot\nabla q(x)=\mathbb{E}_{x\sim q(x)}[s\cdot\nabla \ln q(x)]\end{aligned}$$

要让$\boldsymbol{s_\theta(x_t,t)}$等于$\nabla_{\boldsymbol{x_t}}\log p(\boldsymbol{x_t}|\boldsymbol{x_0})$的加权平均【即$\nabla_{\boldsymbol{x_t}}\log p(\boldsymbol{x_t})$】只需要最小化$\|\boldsymbol{s_\theta(x_t,t)}-\nabla_{\boldsymbol{x_t}}\log p(\boldsymbol{x_t}|\boldsymbol{x_0})\|^2$的加权平均 https://spaces.ac.cn/archives/9209

Generative models and their physics genes

Goodfellow,
NIPS tutorial, 1701.00160

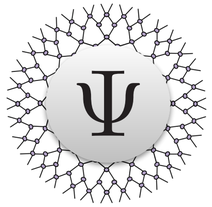
$p(\boldsymbol{x})$

Direct

GAN

Explicit density

Implicit density



Tensor Networks

Tractable density

-Fully visible belief nets

-NADE

-MADE

-PixelRNN

-Change of variables models (nonlinear ICA)

Approximate density

Variational

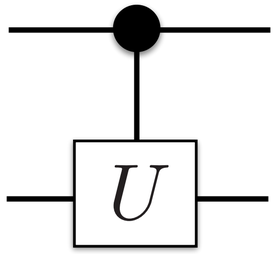
Variational autoencoder

Markov Chain

Boltzmann machine

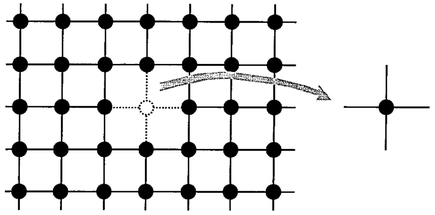
Markov Chain

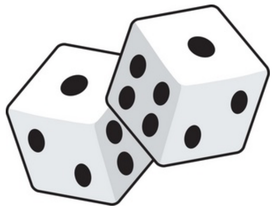
GSN

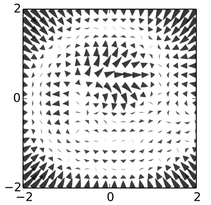


Quantum Circuits

+Diffusion models





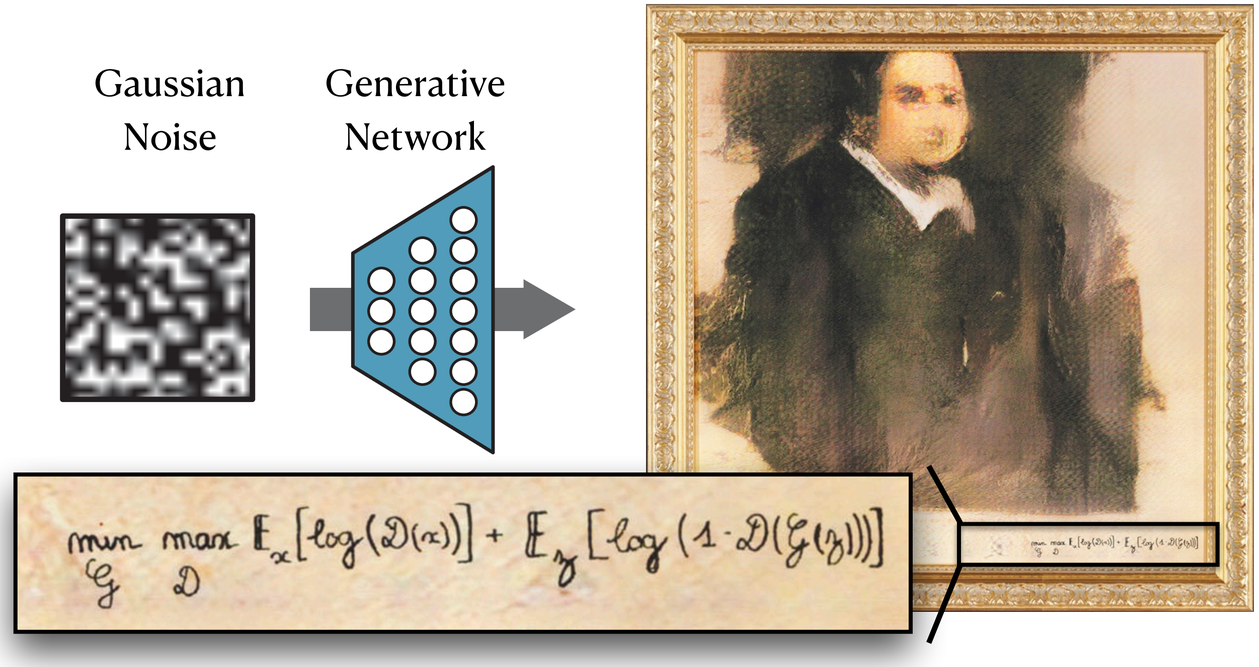


GAN

Likelihood free simulator

Prone to mode collapse

More tricky to train than others

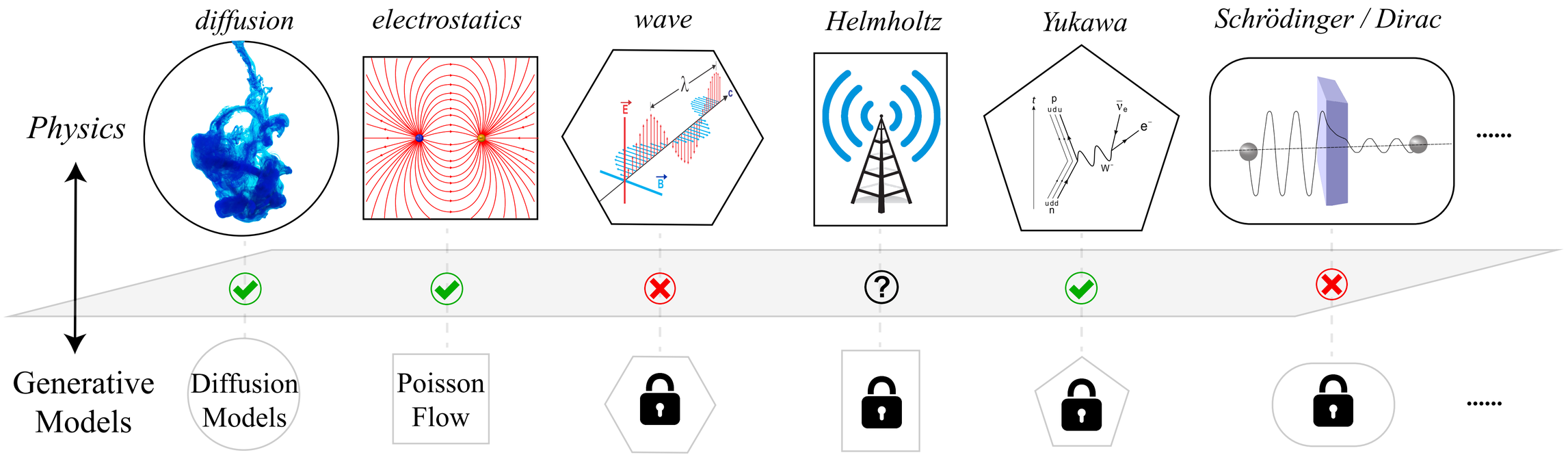


https://www.christies.com/Features/A-collaboration-between-two-artists-one-human-one-a-machine-9332-1.aspx

Performance have been surpassed by diffusion models

I found GAN to be less useful for serious scientific applications

Physical processes for generative modeling



Liu et al, 2304.02637