Neural Canonical Transformations

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

"A Hamiltonian Extravaganza"

—Danilo J. Rezende@DeepMind

Sep 26 *Symplectic ODE-Net*, 1909.12077

Sep 27 *Hamiltonian Graph Networks with ODE Integrators*, 1909.12790

Sep 29 *Symplectic RNN*, 1909.13334

Sep 30 *Equivariant Hamiltonian Flows*, 1909.13739

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

Neural Canonical Transformation with Symplectic Flows, 1910.00024

Hamiltonian dynamics

Hamiltonian equations

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

Hamiltonian dynamics

Hamiltonian equations

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

Phase space variables

$$\boldsymbol{x} = (p, q)$$

Symplectic metric

$$J = \begin{pmatrix} & I \\ -I & \end{pmatrix}$$

Hamiltonian dynamics

Hamiltonian equations

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

Phase space variables

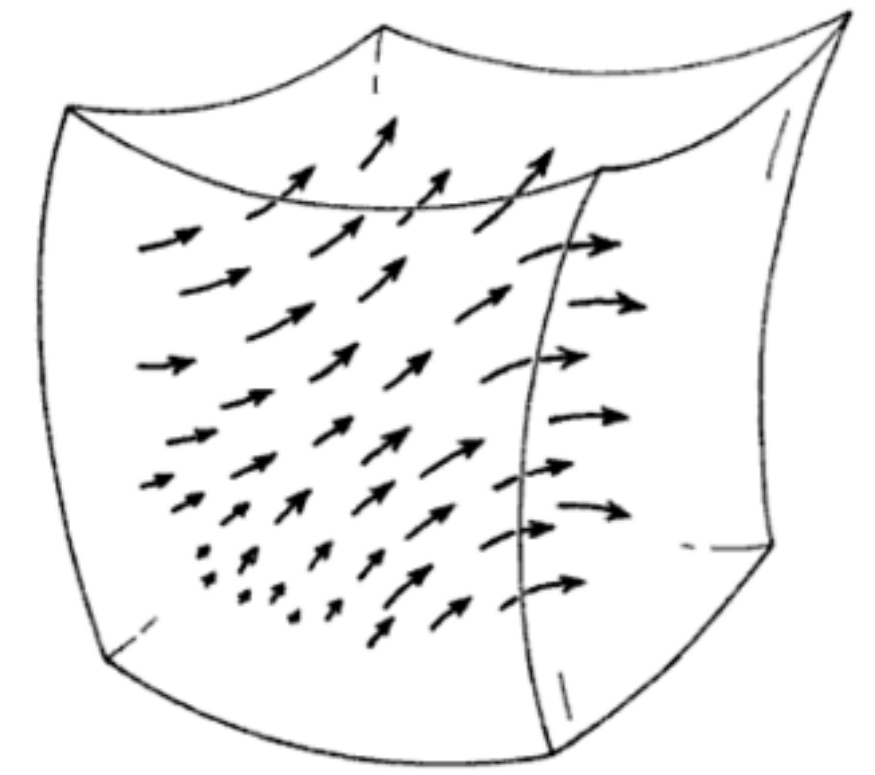
$$\boldsymbol{x} = (p, q)$$

Symplectic metric

$$J = \begin{pmatrix} & I \\ -I & \end{pmatrix}$$

Symplectic gradient flow

$$\dot{\boldsymbol{x}} = \nabla_{\boldsymbol{x}} H(\boldsymbol{x}) J$$



Hamiltonian dynamics

Hamiltonian equa **nce v** **tic gradient flow**

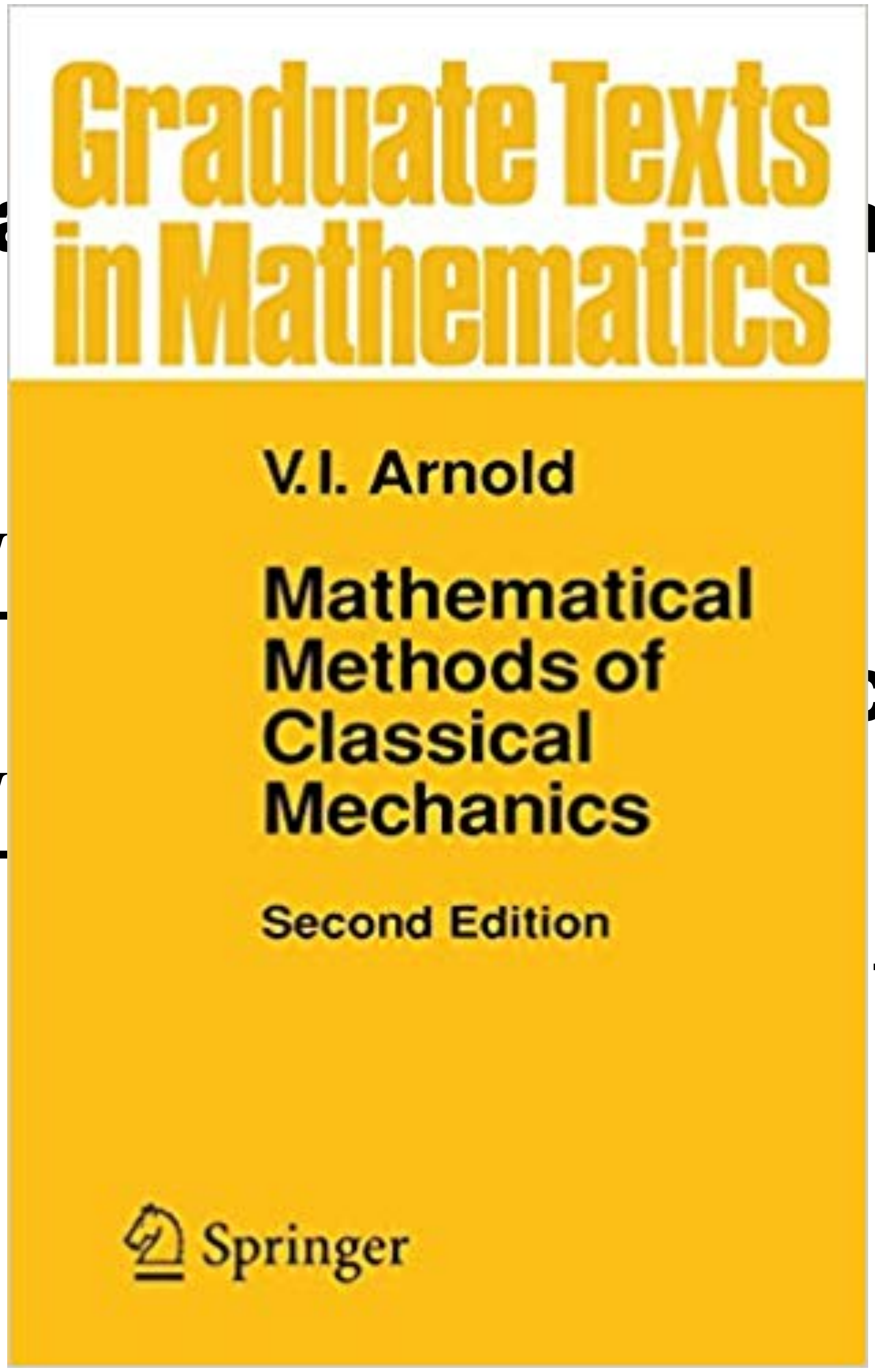
$$\begin{cases} \dot{p} = -\frac{\partial H}{\partial q} \\ \dot{q} = +\frac{\partial H}{\partial p} \end{cases}$$

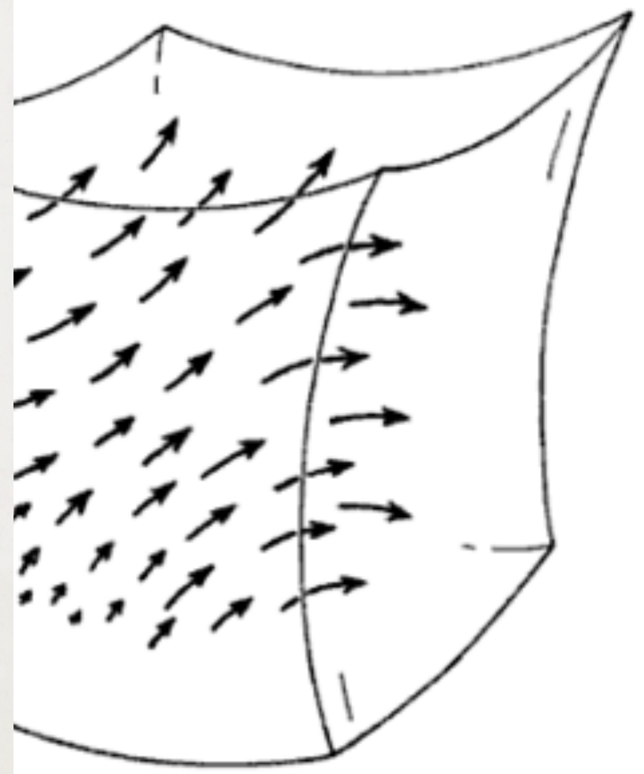
$(p, q$

$= \nabla_x H(\boldsymbol{x}) J$

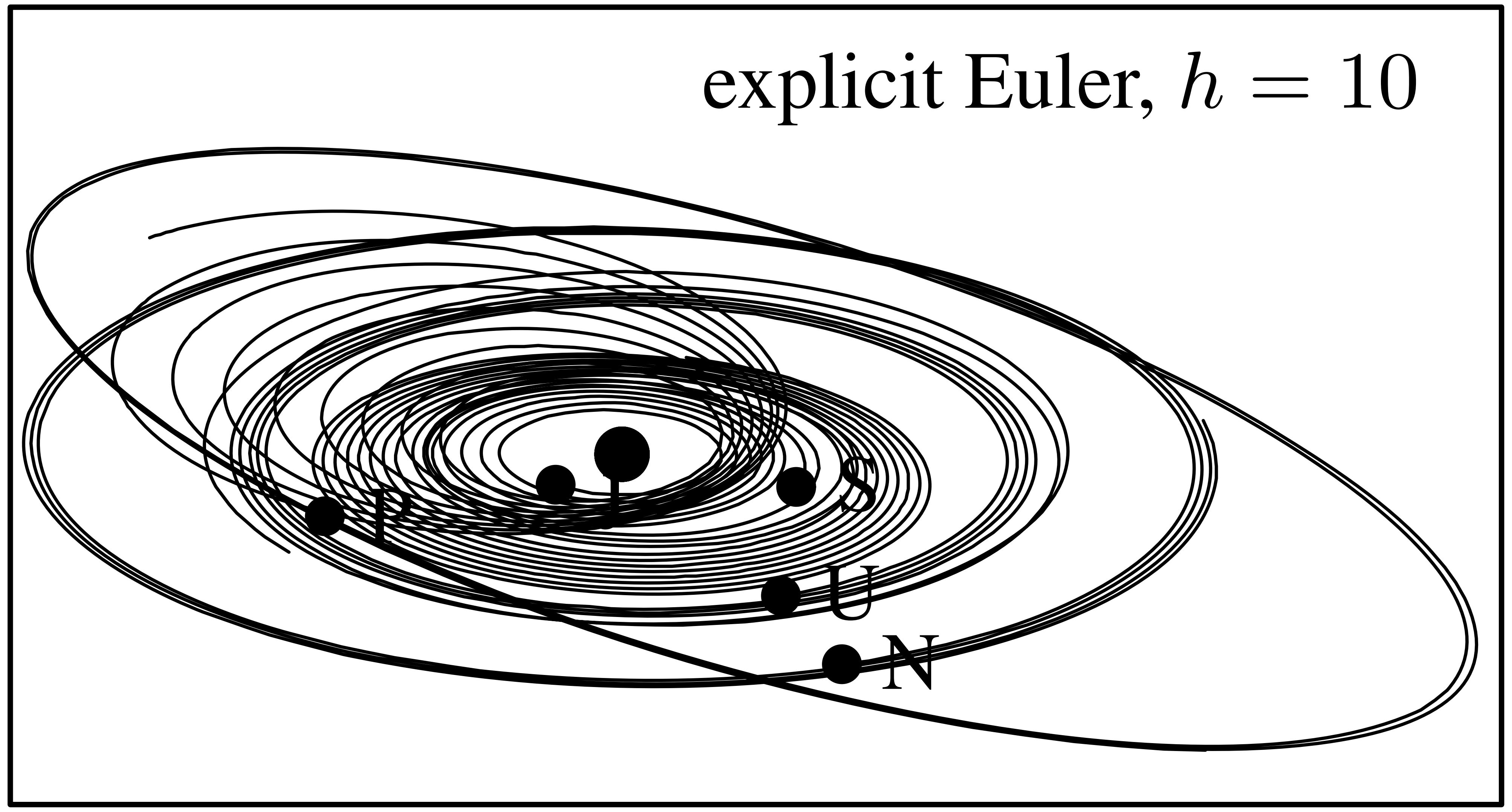
ctic r

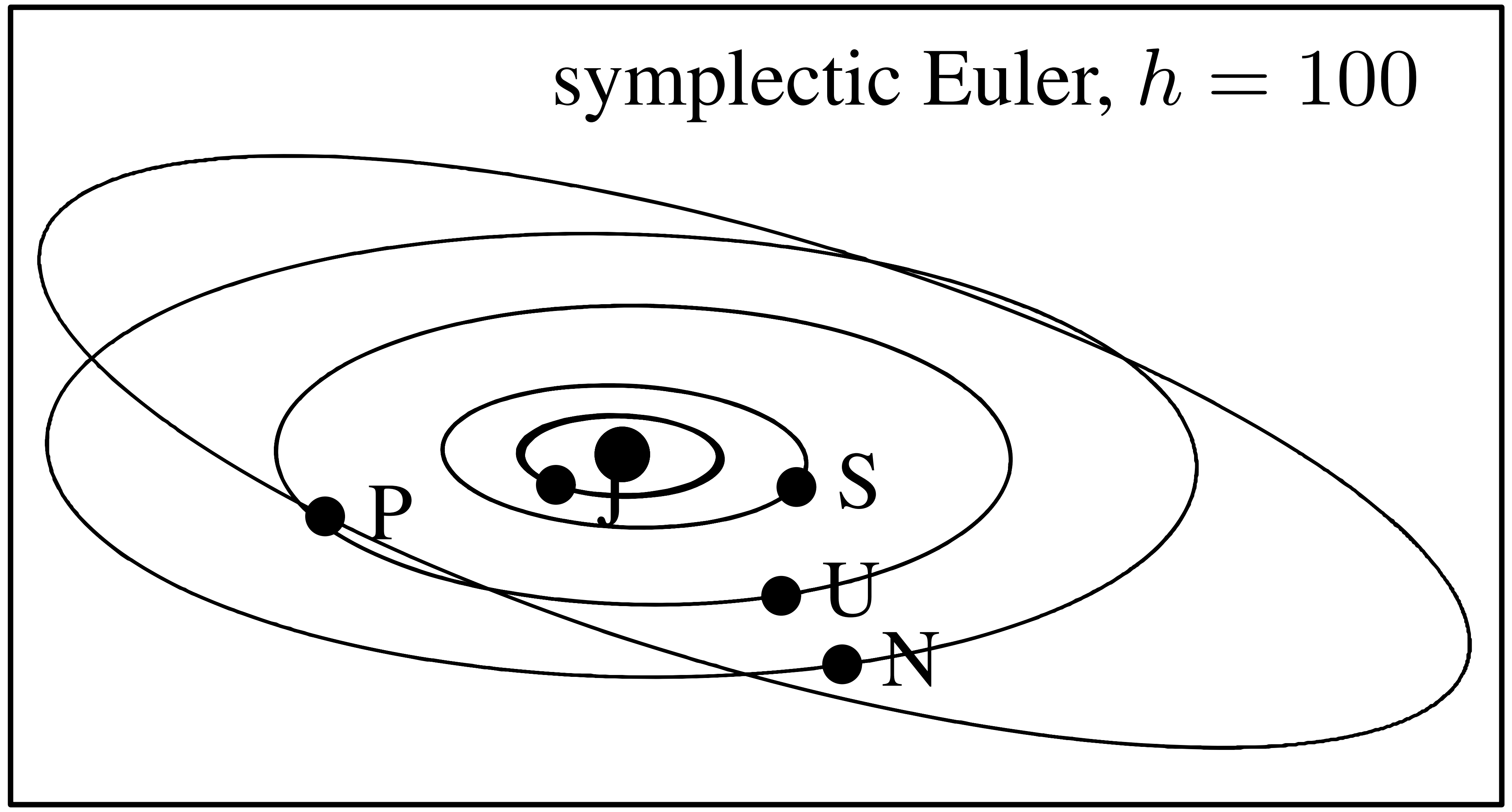
$-I$





Symplectic Integrators





from Hairer et al, Geometric Numerical Integration

Canonical Transformations

Change of variables

$$\boldsymbol{x} = (p, q) \longleftrightarrow \boldsymbol{z} = (P, Q)$$

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

Canonical Transformations

Change of variables

$$\boldsymbol{x} = (p, q) \longleftrightarrow \boldsymbol{z} = (P, Q)$$

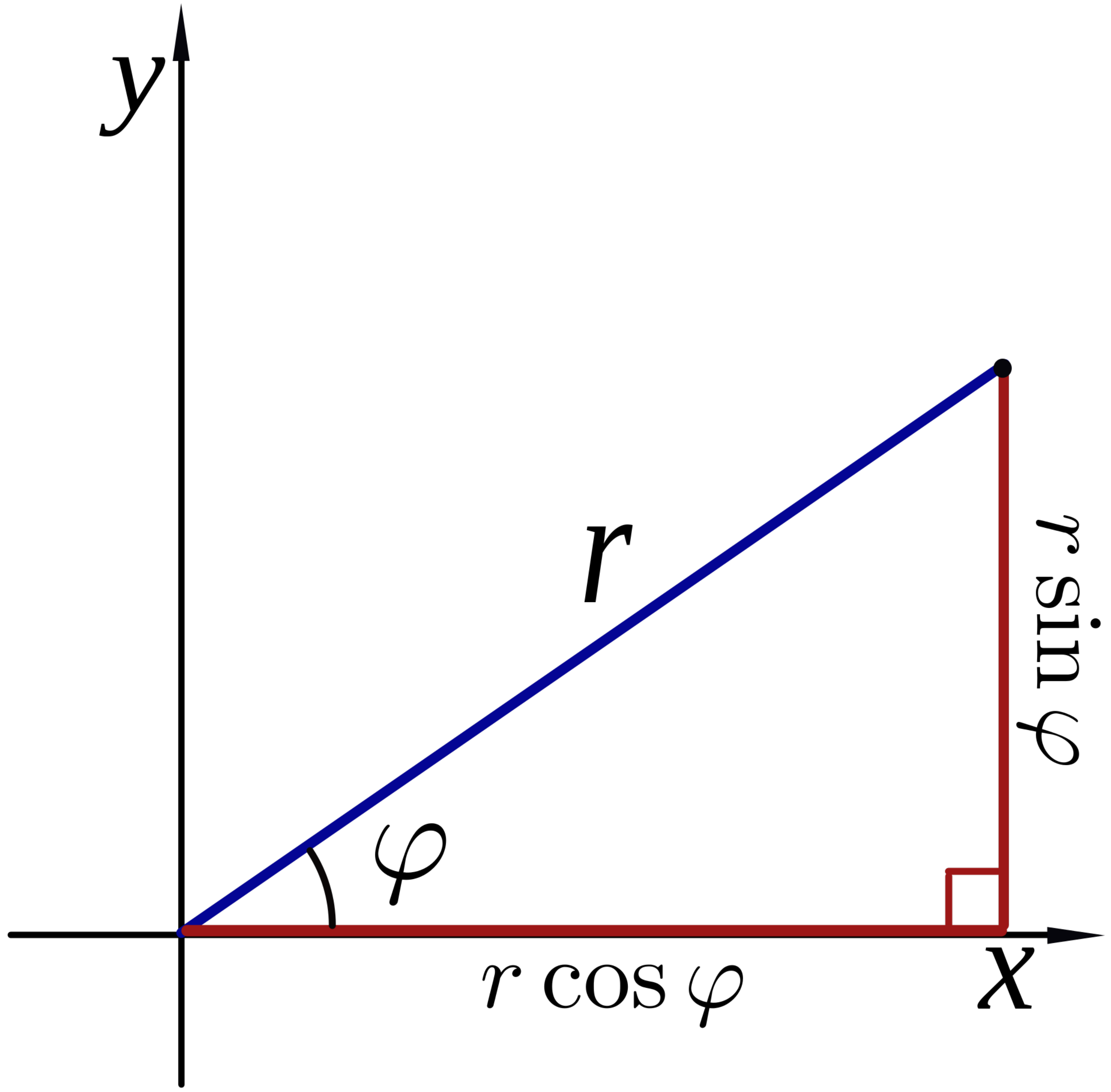
which satisfies $\left(\nabla_{\boldsymbol{x}} \boldsymbol{z}\right) J \left(\nabla_{\boldsymbol{x}} \boldsymbol{z}\right)^{T} = J$ symplectic condition

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

Preserves Hamiltonian dynamics in the "latent phase space"

Example: Cartesian <—> Polar

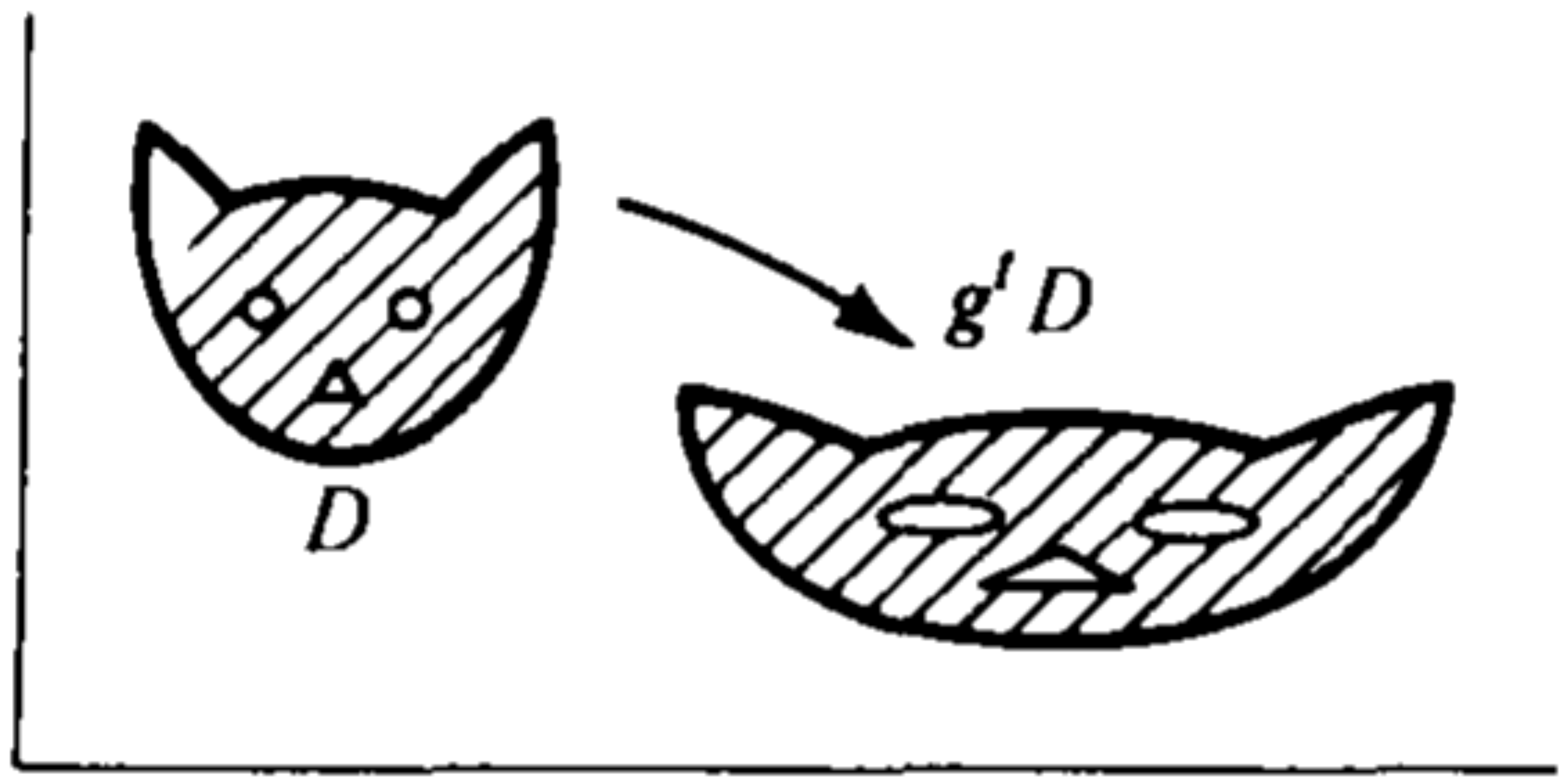
$$H = \frac{1}{2}\left(p_x^2 + p_y^2\right)$$



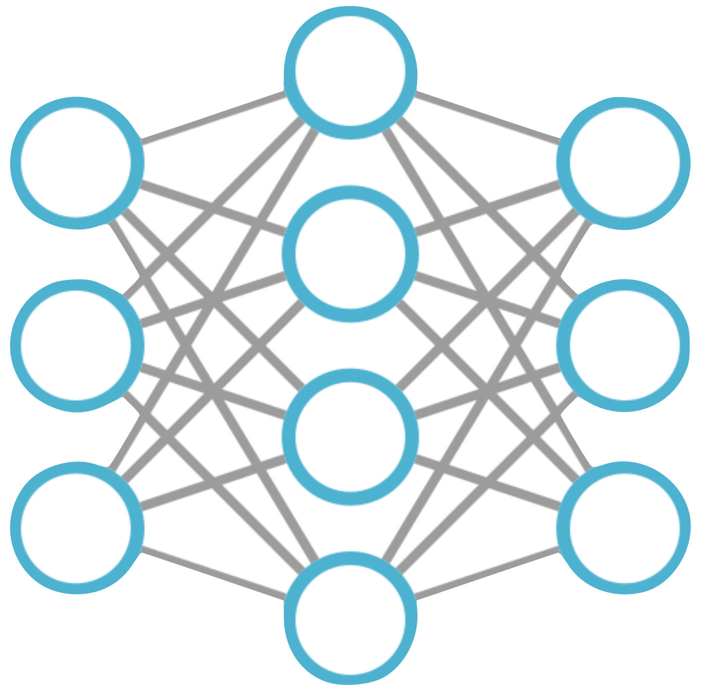
$$K = \frac{1}{2}\left(p_r^2 + \frac{1}{r^2}p_\varphi^2\right)$$

Phase space perspective

- Canonical transformation deforms phase space density $\rho(\boldsymbol{x}) = e^{-\beta H(\boldsymbol{x})}$
- Symplectic condition => Jacobian determinant = 1
- **Liouville theorem:** incompressible flow in phase space

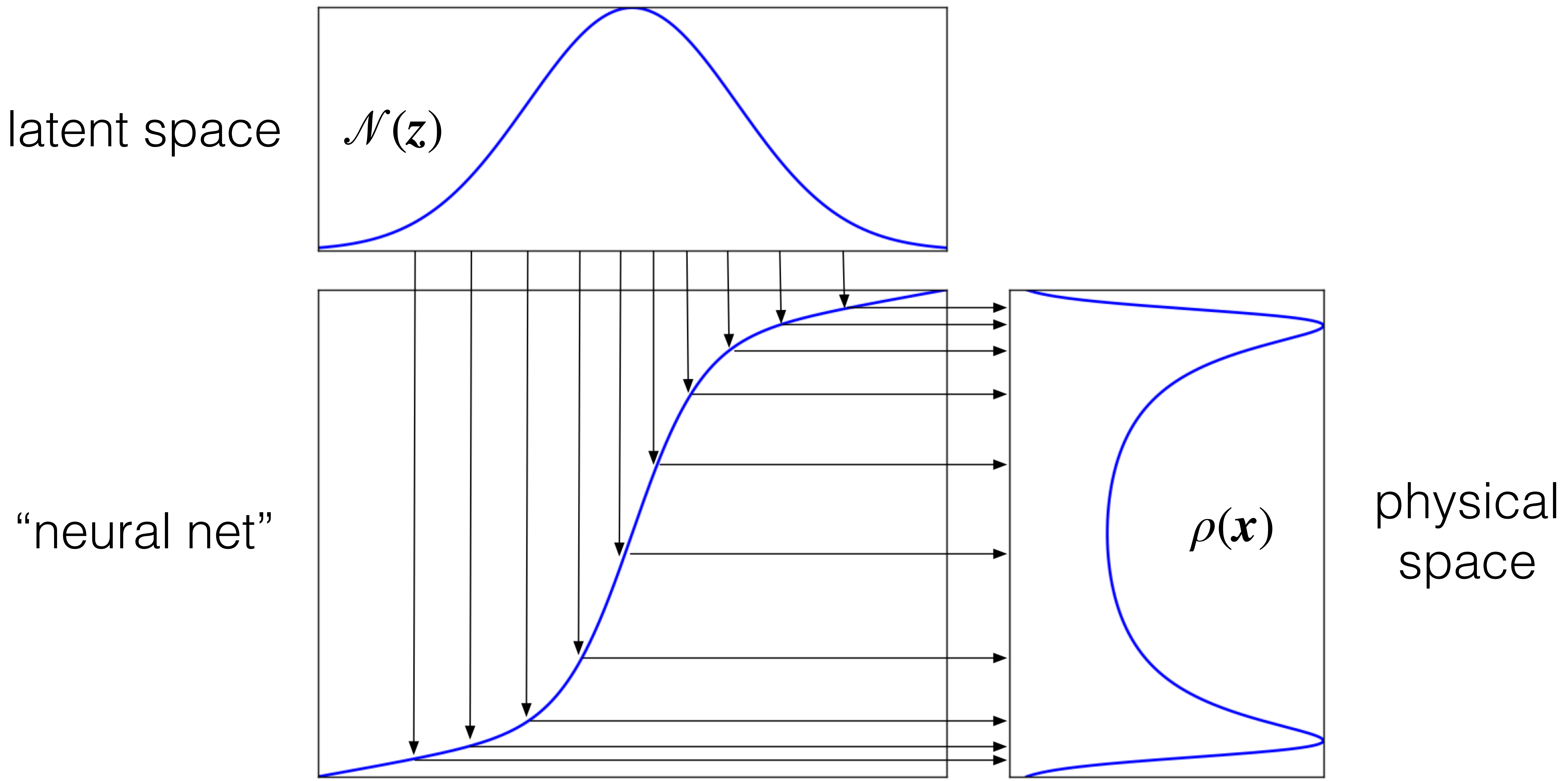


How to design useful canonical transformations ?

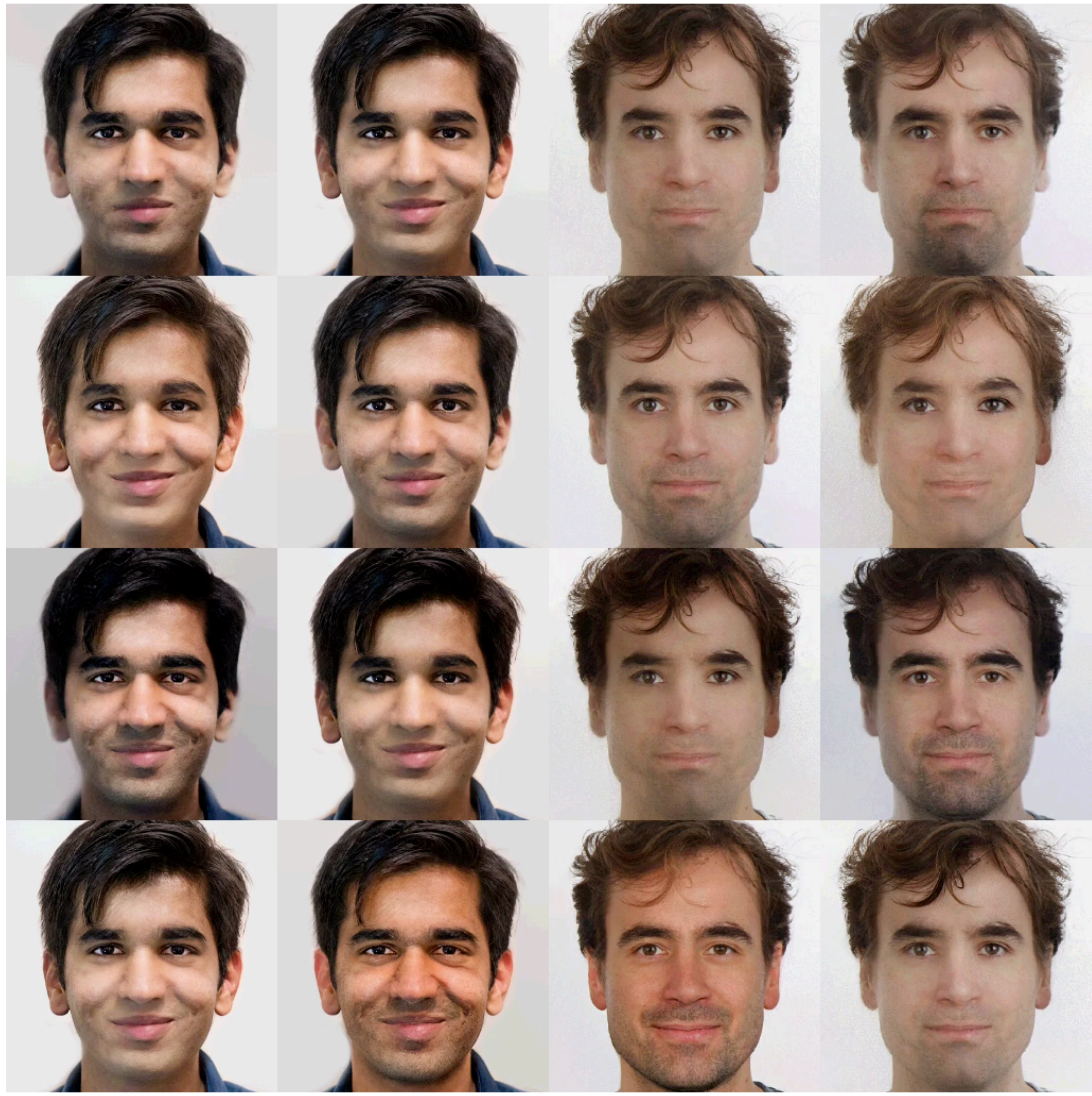


Neural Canonical Transformations

Neural transformation in 1d



Neural transformations in higher dims



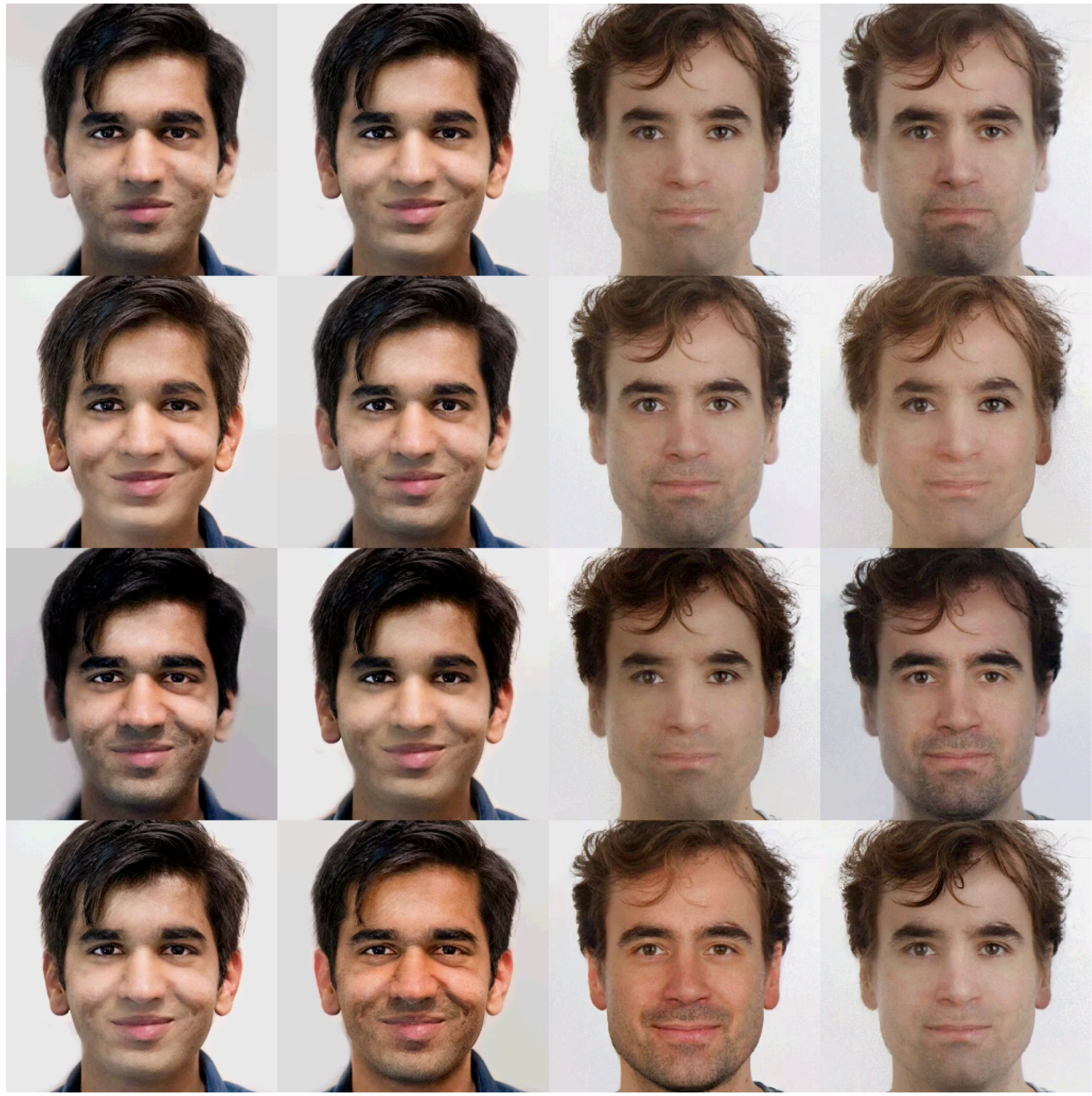
WaveNet 1609.03499 1711.10433

https://deepmind.com/blog/wavenet-generative-model-raw-audio/
https://deepmind.com/blog/high-fidelity-speech-synthesis-wavenet/

Glow 1807.03039

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

Neural transformations in higher dims



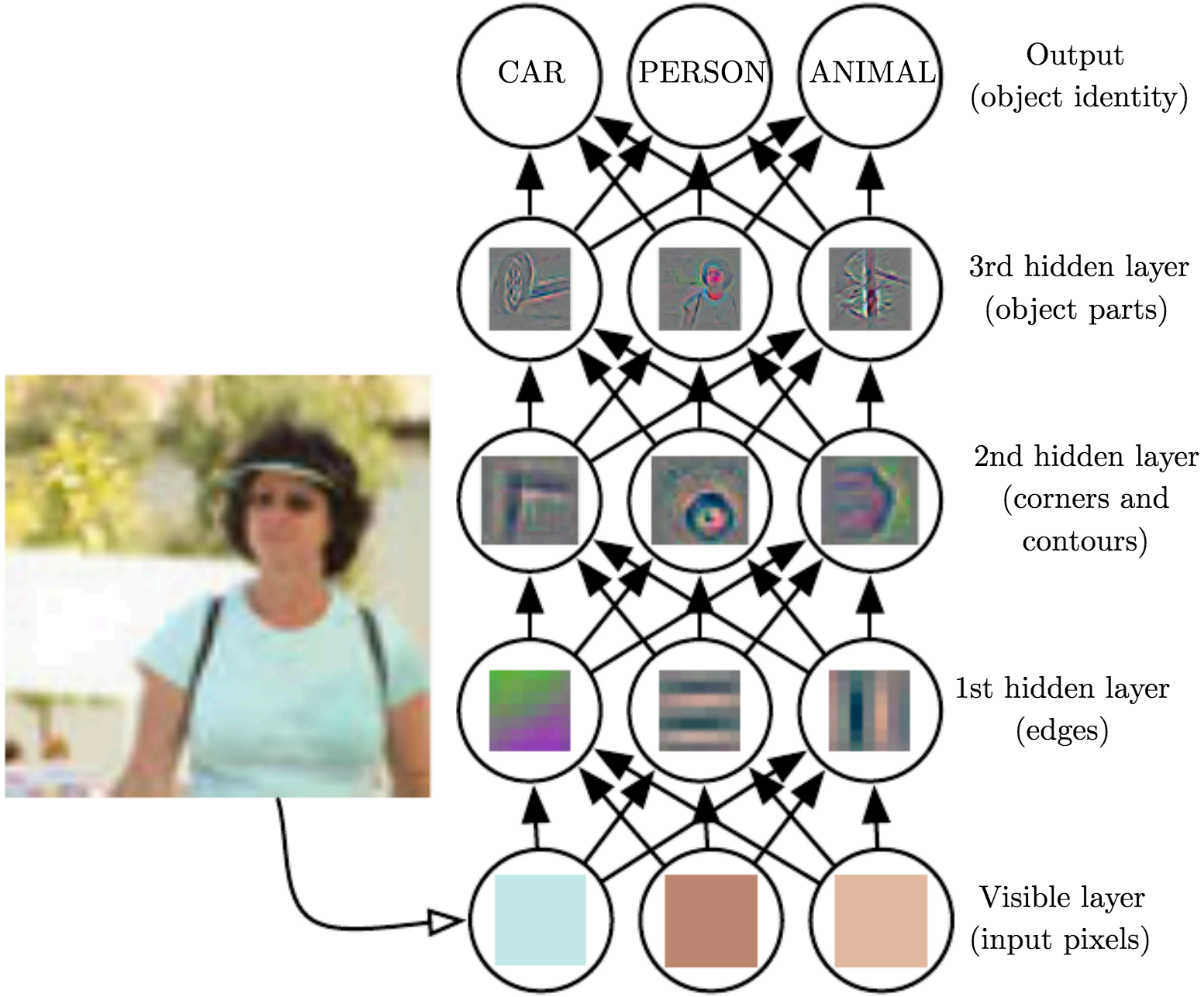
WaveNet 1609.03499 1711.10433

https://deepmind.com/blog/wavenet-generative-model-raw-audio/
https://deepmind.com/blog/high-fidelity-speech-synthesis-wavenet/

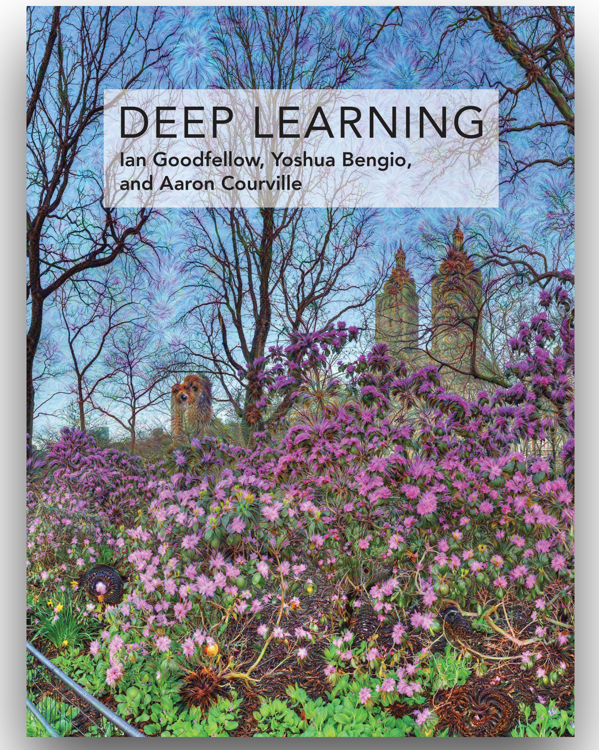
Glow 1807.03039

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

Representation Learning

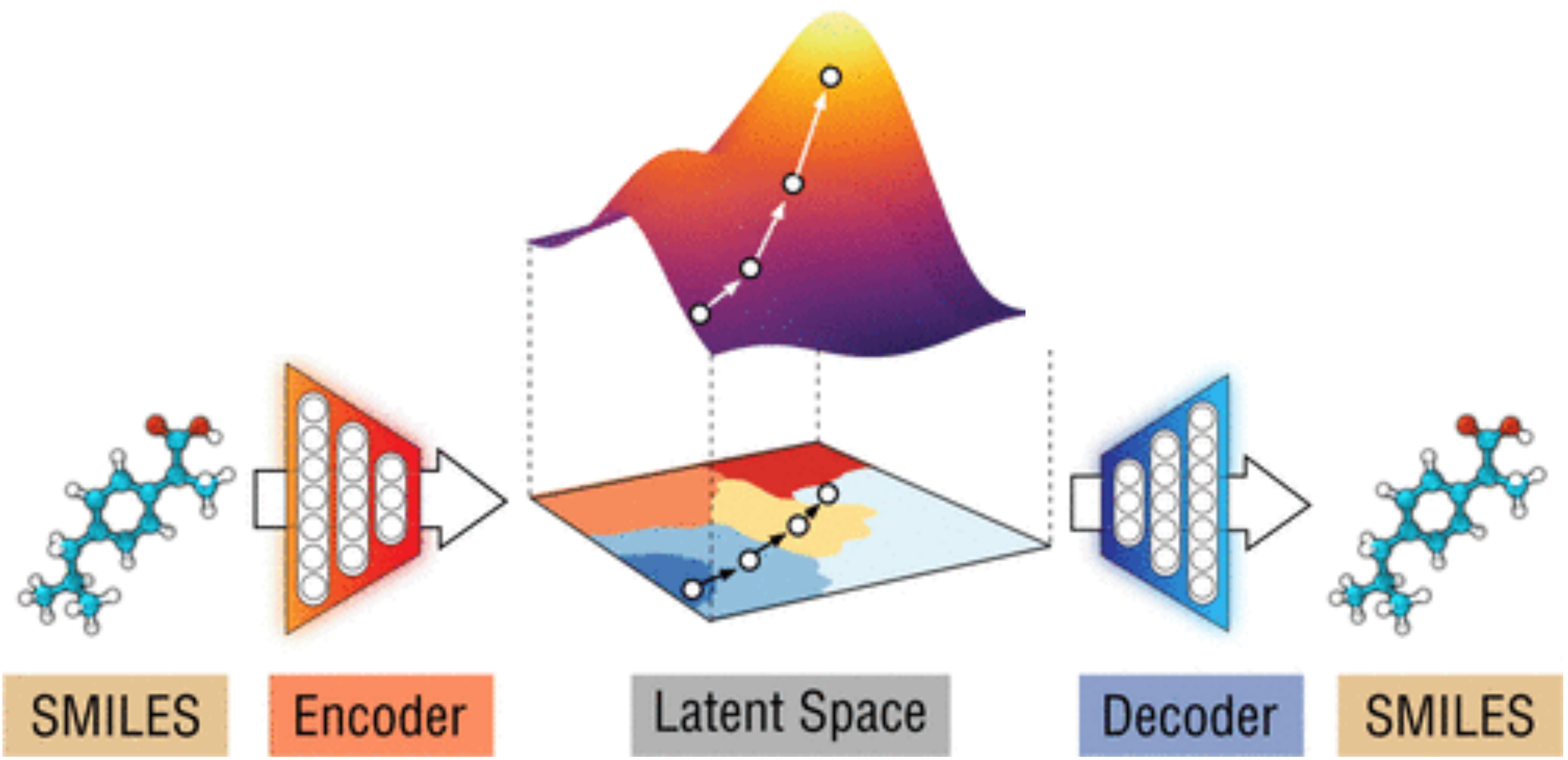


Page 6
Figure 1.2

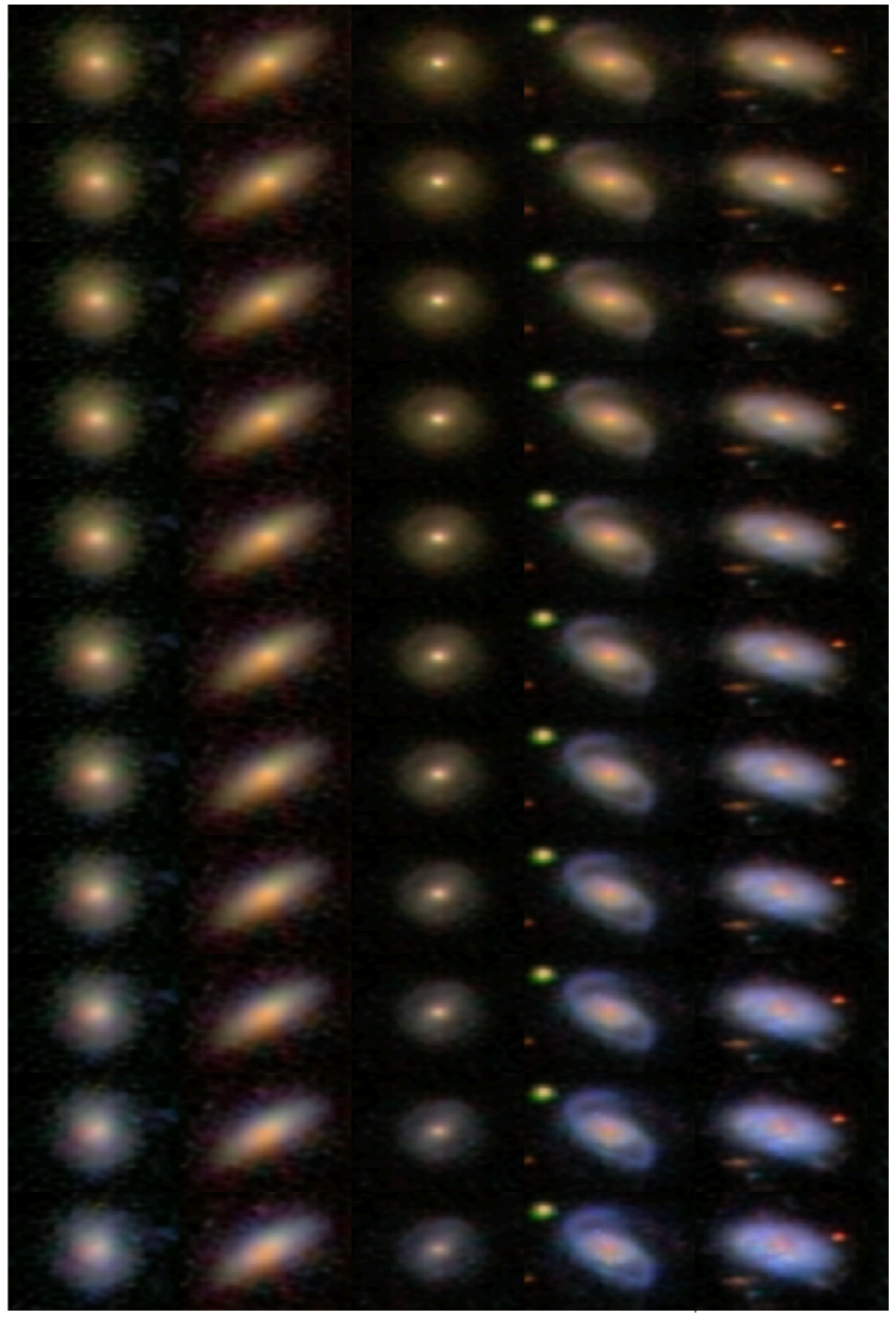


Goodfellow, Bengio, Courville, http://www.deeplearningbook.org/

Learning representation for science



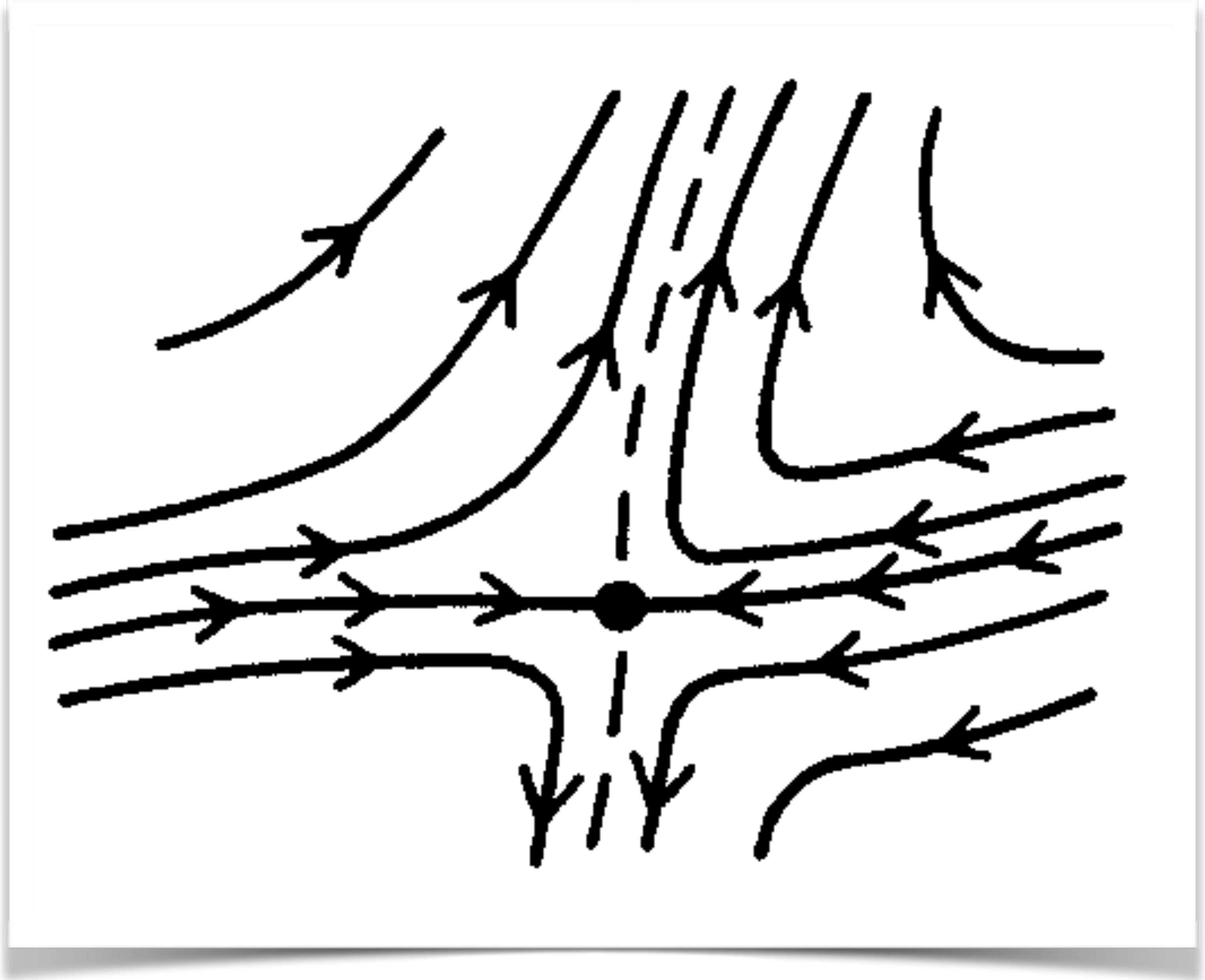
Automatic chemical design
Gomez-Bombarelli et al,1610.02415



Galaxy evolution
Schawinski et al, 1812.01114

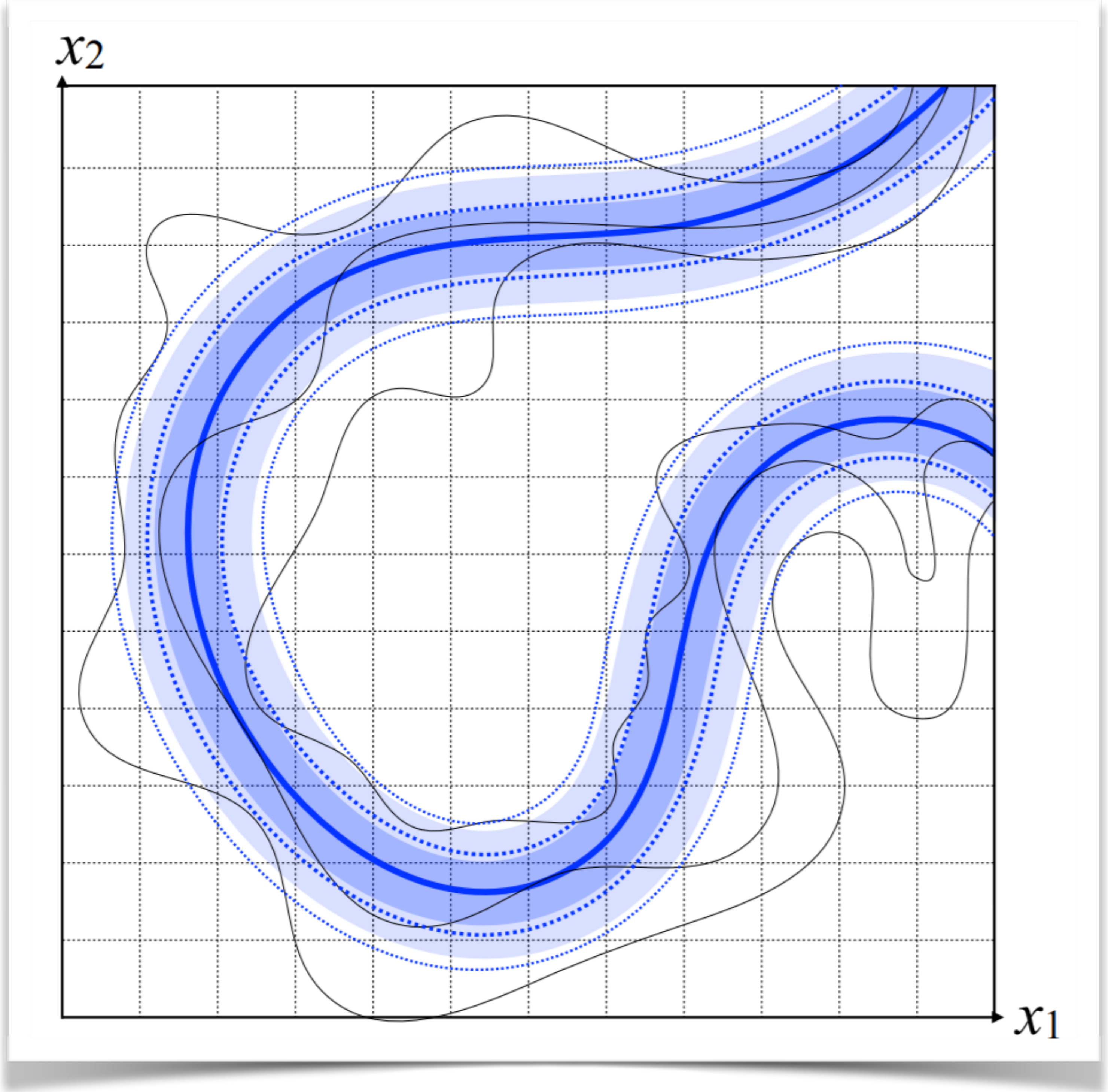
Representation learning in physics

Renormalization group



Effective theory emerges upon transformation to the variables

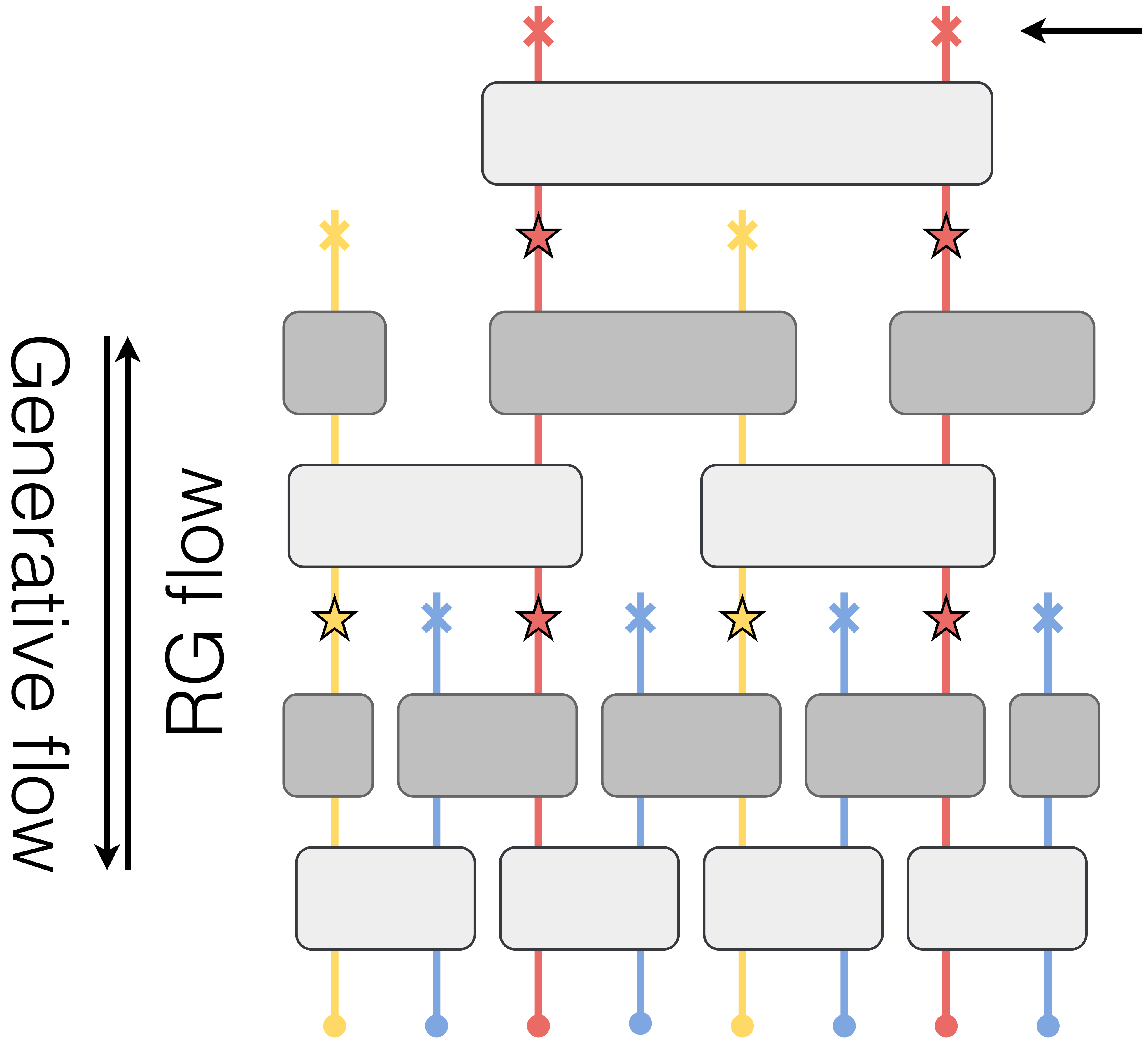
Monte Carlo update

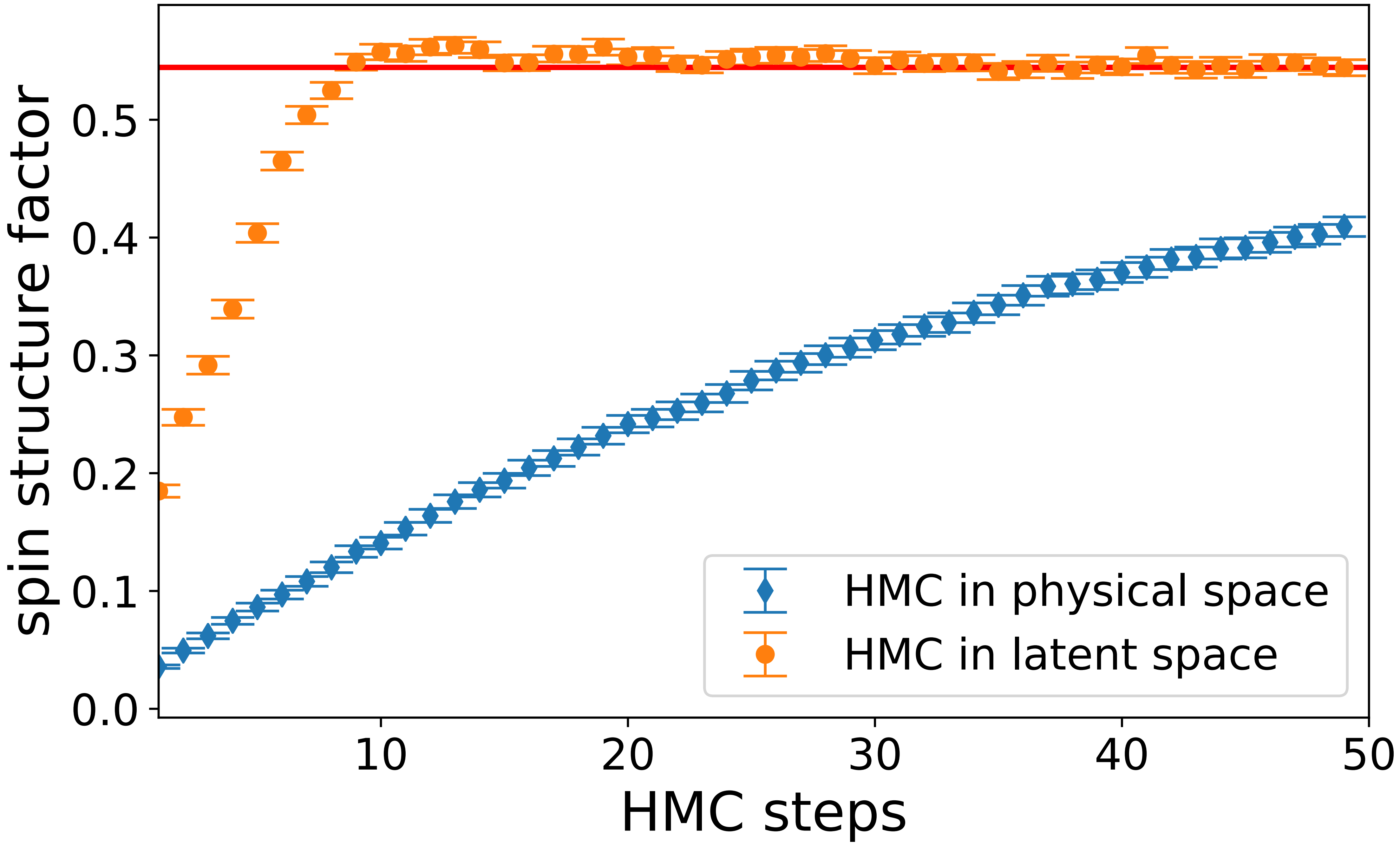


Physics happens on a manifold
Learning unfolds that manifold

Neural Renormalization Group

Multi-scale invertible neural network Li and LW, PRL '18

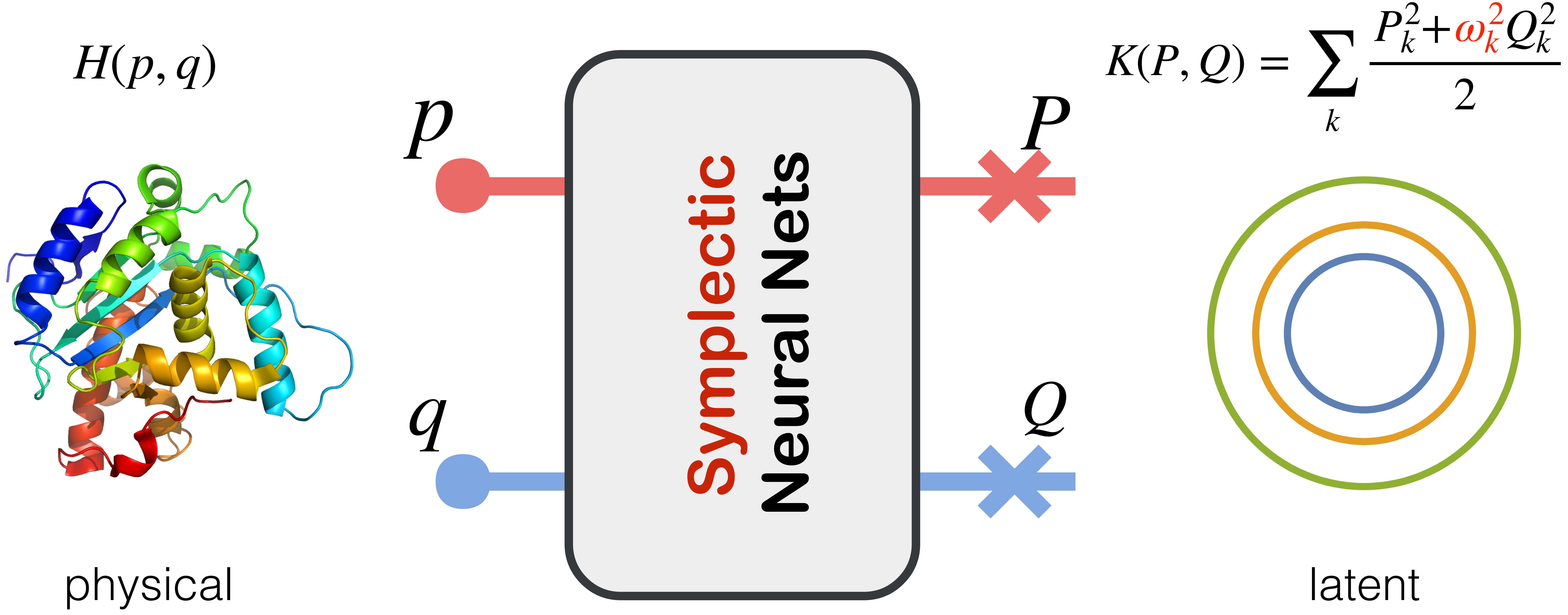




https://github.com/li012589/NeuralRG

Identify collective variables; and fast sampling w/ learned representation

Neural Canonical Transformations



Learn the network and the latent harmonic frequency together

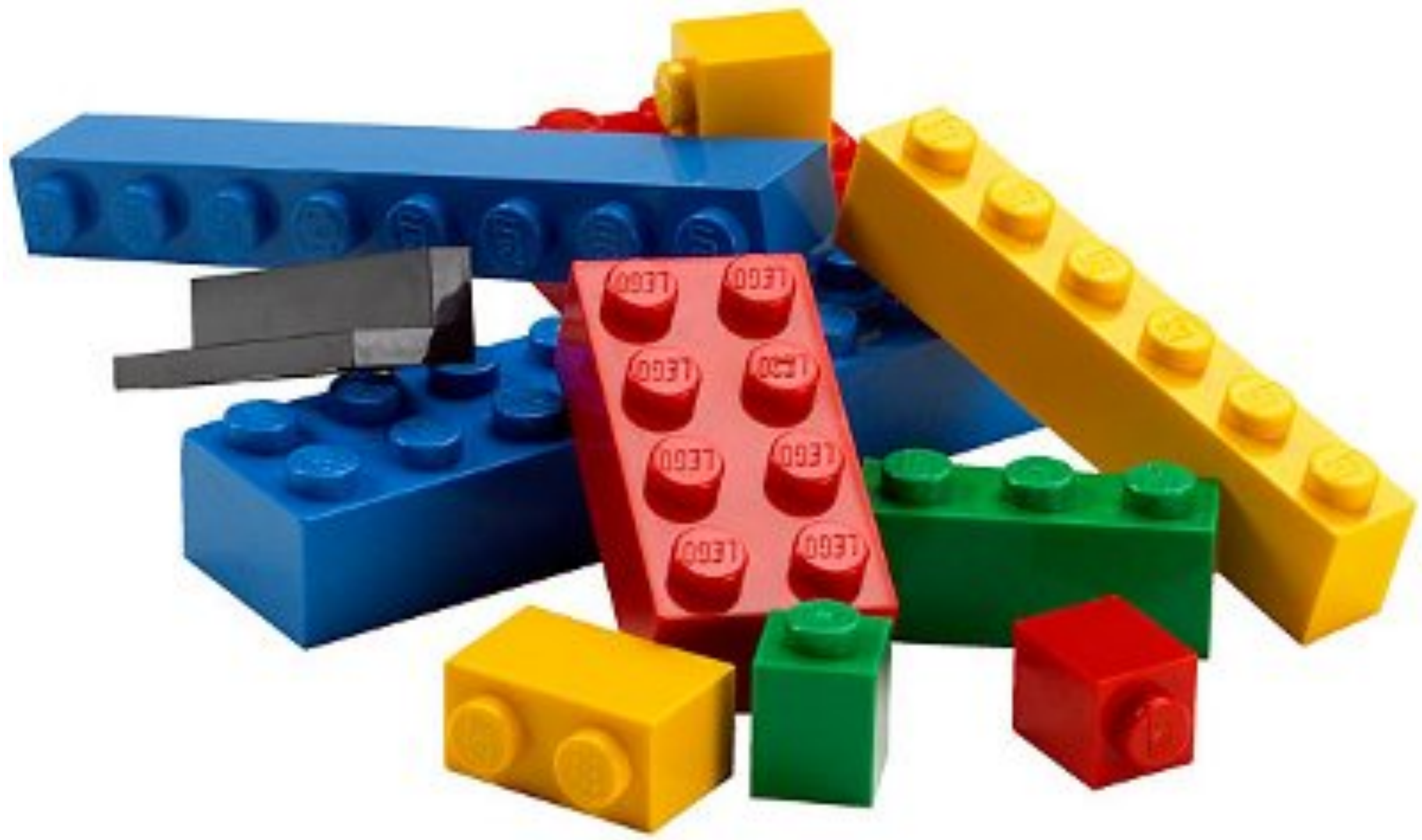
Modular design of the symplectic network

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

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

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

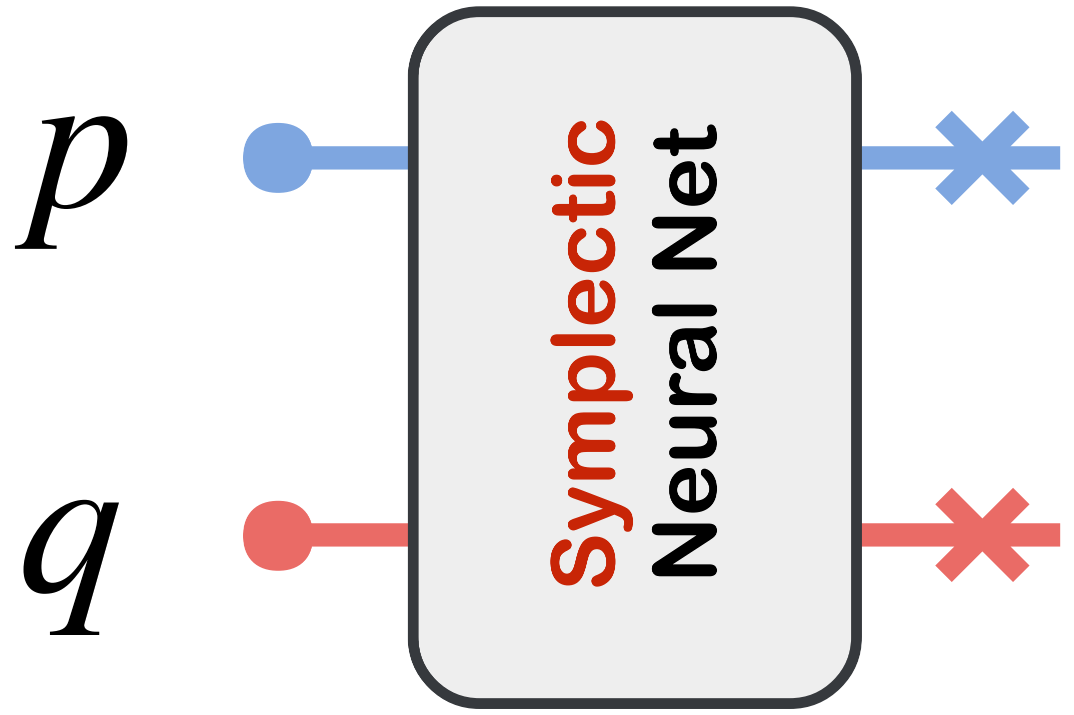
symplectic group



Compose symplectic primitives to a deep neural network

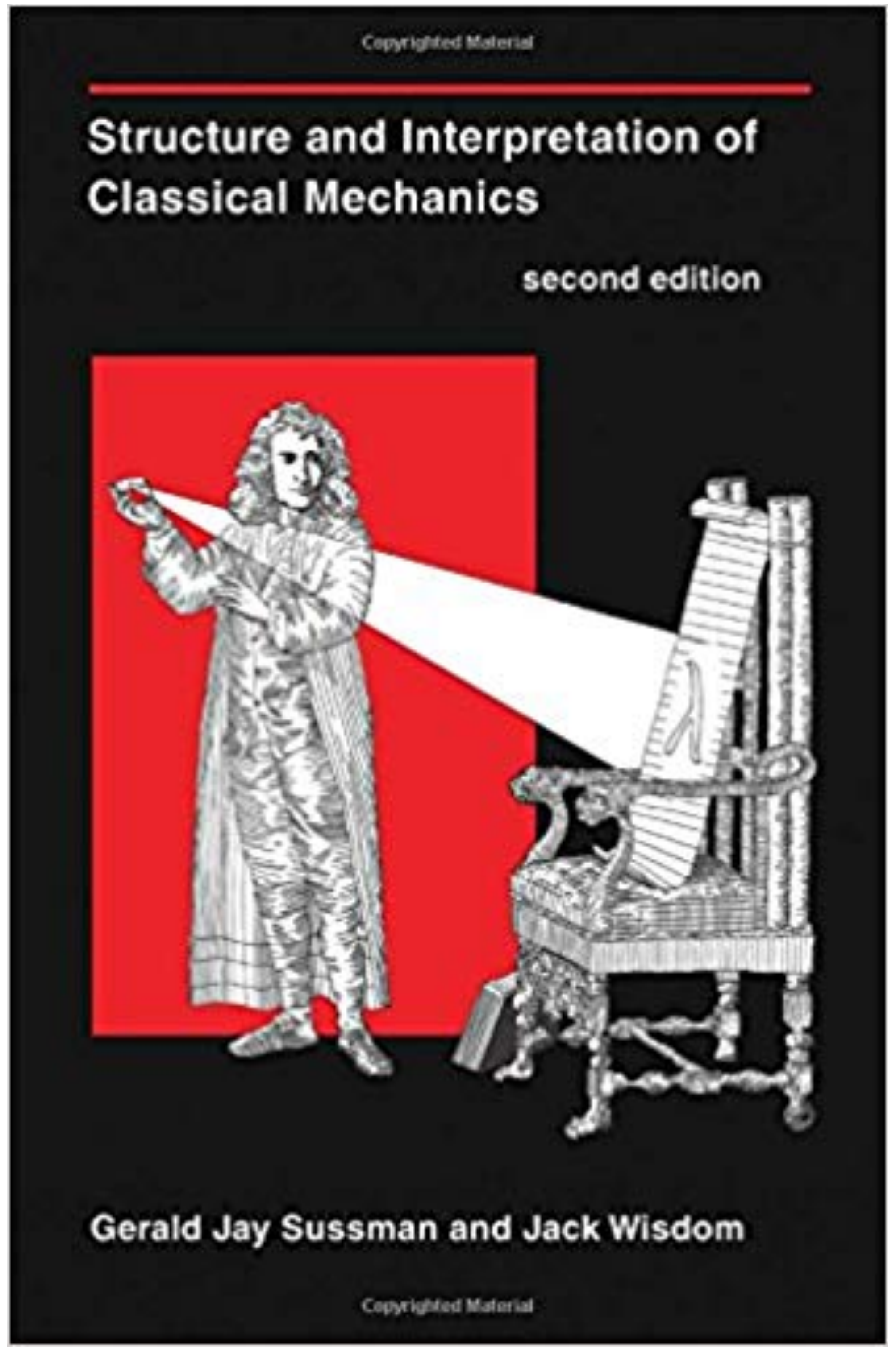
Neural symplectic primitives

- **Neural coordinate transformation**



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

$$Q = f(q)$$ invertible net



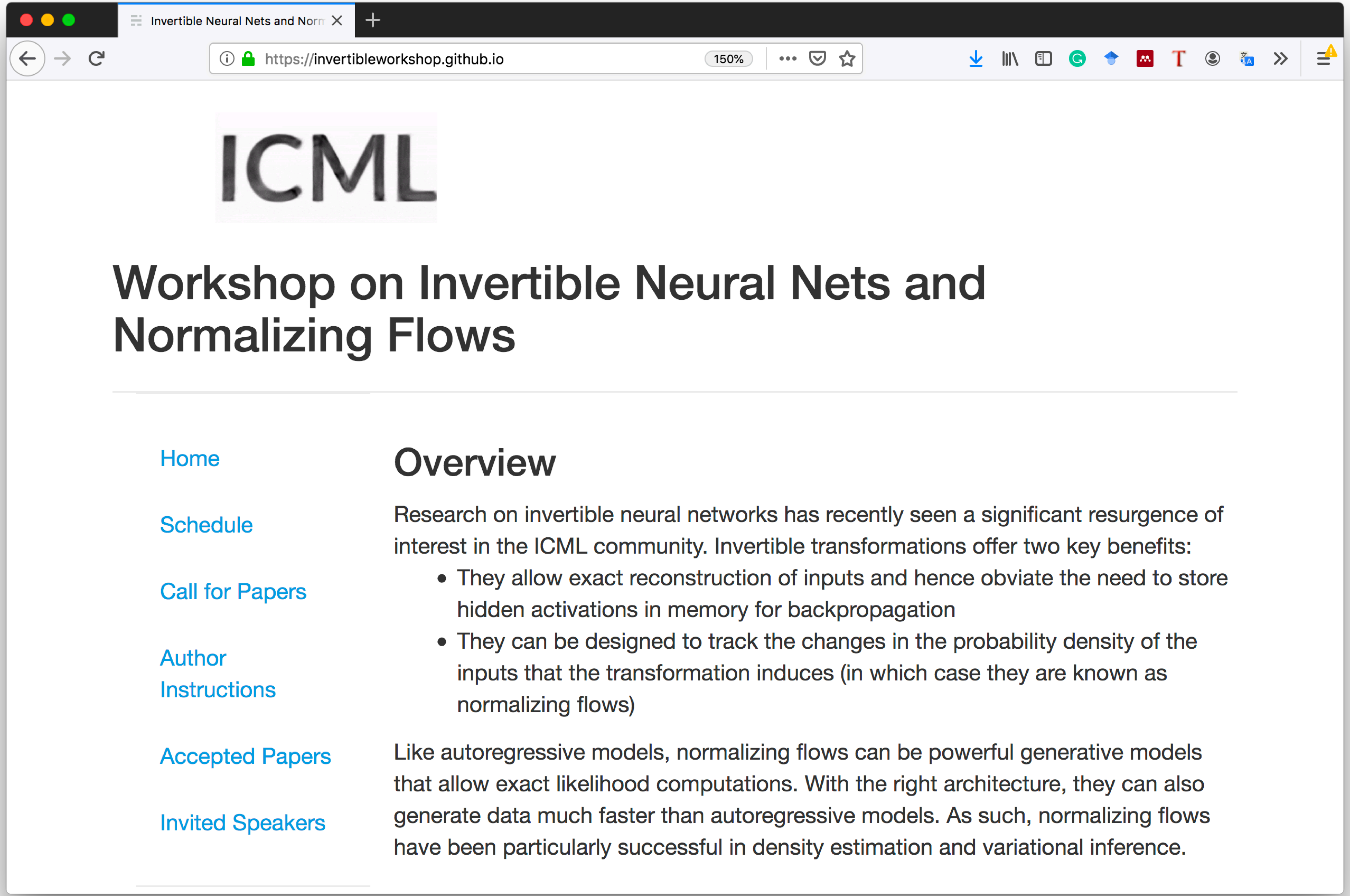
- Linear transformation: Symplectic Lie algebra

- Continuous-time flow: Symplectic generating functions

See also Bondesan, Lamacraft, 1906.04645

Neural ODE, Chen et al, 1806.07366, Monge-Ampere flow, Zhang et al 1809.10188

A digression on invertible neural networks


Invertible Neural Nets and Norm

https://invertibleworkshop.github.io

150%

ICML

Workshop on Invertible Neural Nets and Normalizing Flows

- Home
- Schedule
- Call for Papers
- Author Instructions
- Accepted Papers
- Invited Speakers

Overview

Research on invertible neural networks has recently seen a significant resurgence of interest in the ICML community. Invertible transformations offer two key benefits:

- They allow exact reconstruction of inputs and hence obviate the need to store hidden activations in memory for backpropagation
- They can be designed to track the changes in the probability density of the inputs that the transformation induces (in which case they are known as normalizing flows)

Like autoregressive models, normalizing flows can be powerful generative models that allow exact likelihood computations. With the right architecture, they can also generate data much faster than autoregressive models. As such, normalizing flows have been particularly successful in density estimation and variational inference.

Survey paper
1908.09257

Normalizing Flows: Introduction and Ideas

Ivan Kobyzev IVAN.KOBYZEV@BOREALISAI.COM
Simon Prince SIMON.PRINCE@BOREALISAI.COM
Marcus A. Brubaker MARCUS.BRUBAKER@BOREALISAI.COM

"Be still like a mountain and flow like a great river."

Lao Tzu

Eric Jang
@Google Brain

Eric Jang: Normalizing Flows in

https://blog.evjang.com/2019/07/nf-jax.html

Saturday, July 6, 2019

Normalizing Flows in 100 Lines of JAX

JAX is a great linear algebra + automatic differentiation library for fast experimentation with and teaching machine learning. Here is a lightweight example, in just 75 lines of JAX, of how to implement Real-NVP.

This post is based off of a tutorial on normalizing flows I gave at the ICML workshop on Invertible Neural Nets and Normalizing Flows. I've already written about how to implement your own flows in TensorFlow using TensorFlow Probability's Bijector API, so to make things interesting I wanted to show how to implement Real-NVP a different way.

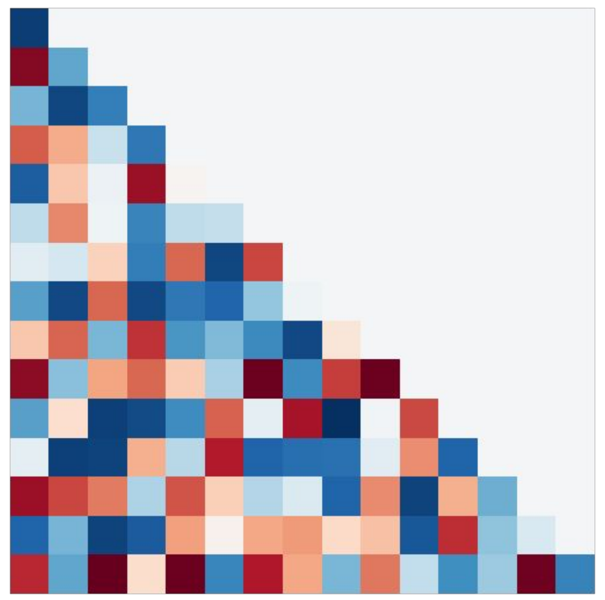
By the end of this tutorial you'll be able to reproduce this figure of a normalizing flow "bending" samples from a 2D Normal distribution to samples from the "Two Moons" dataset. Real-NVP forms the basis of a lot of flow-based architectures (as of 2019), so this is a good template to start learning from.

The essence of flows

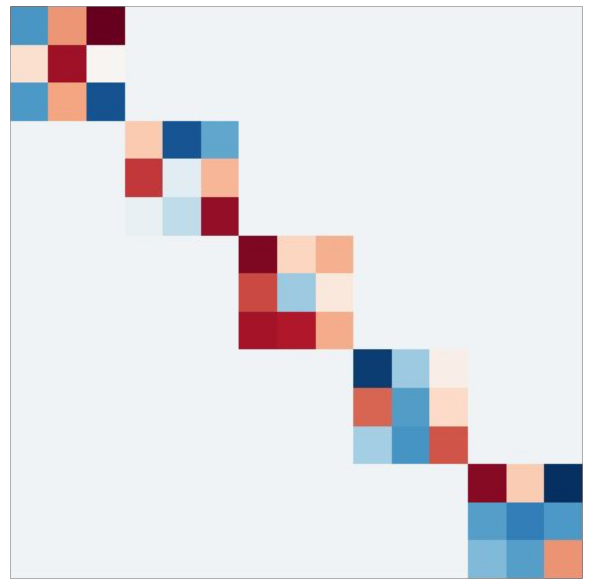
Change of variables $\boldsymbol{x} \leftrightarrow \boldsymbol{z}$

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

Design principle: Efficient Jabobian and inverse



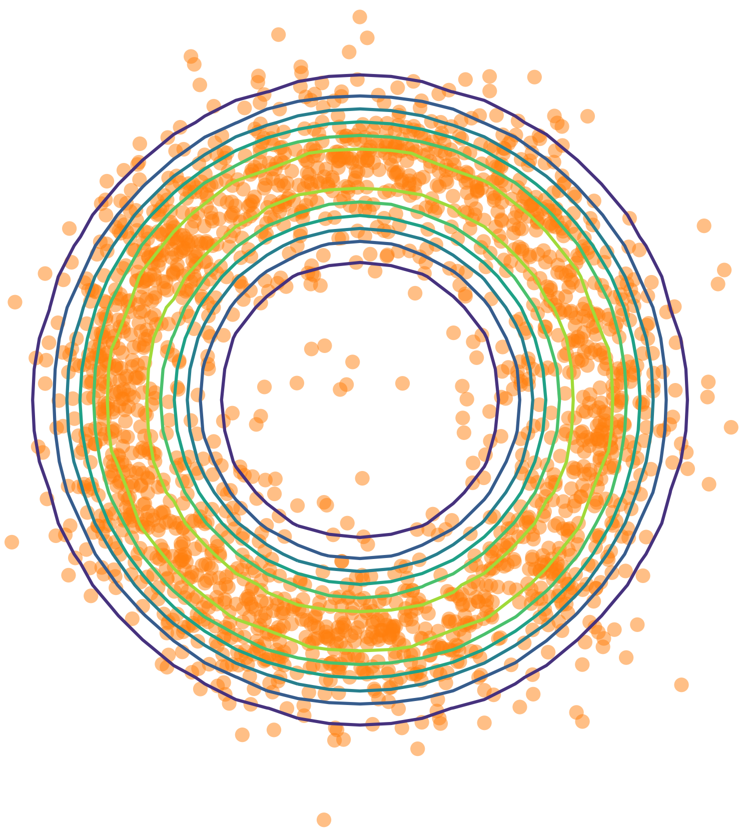
Autoregressive

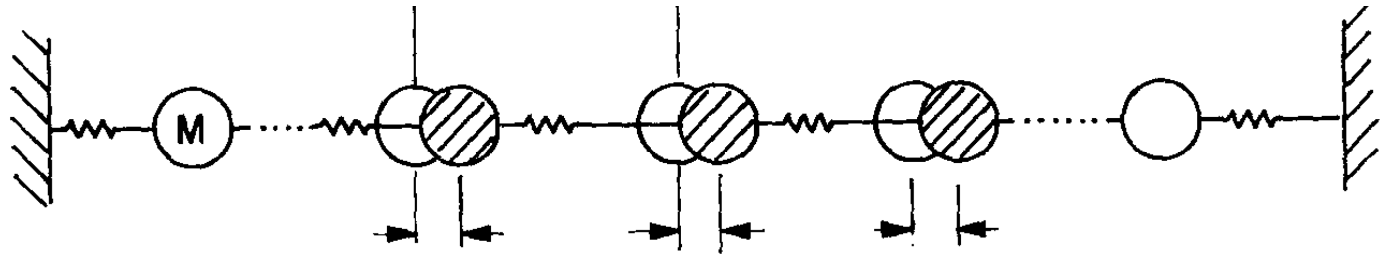


Neural RG

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

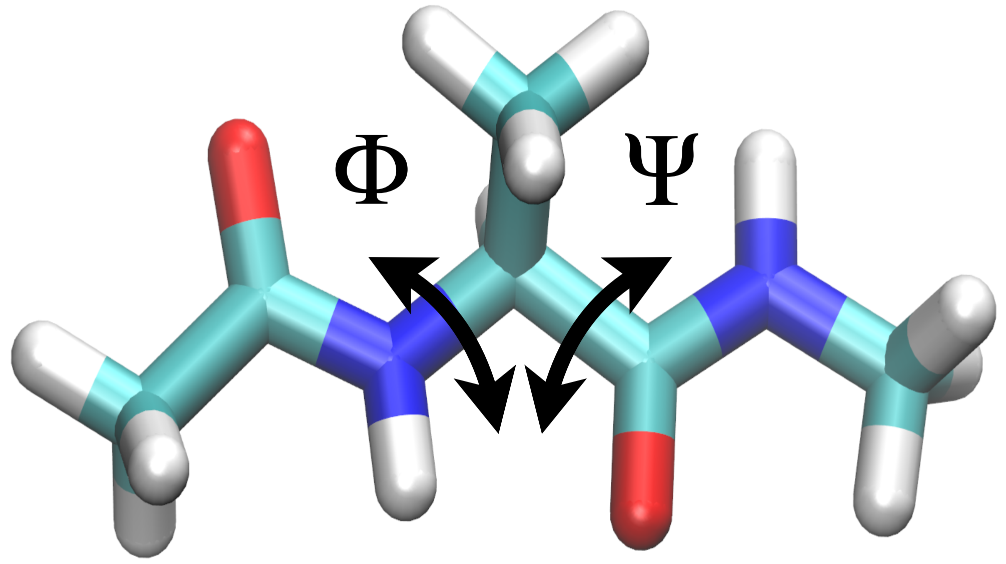
Continuous-time flow





How is this going to be useful ?

Let's play with examples!



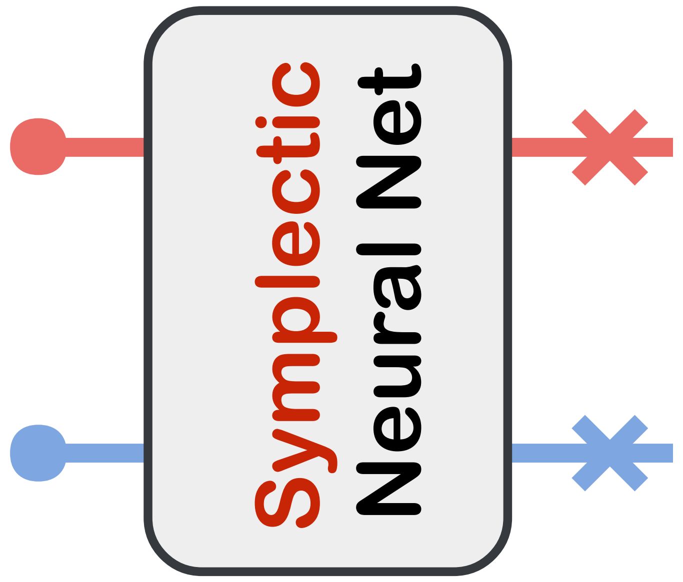
Training approaches

Variational calculation

"learn from Hamiltonian"

$$\mathscr{L} = \int d\boldsymbol{x}\, \rho(\boldsymbol{x}) \left[\ln \rho(\boldsymbol{x}) + \beta H(\boldsymbol{x}) \right]$$

$\boldsymbol{x}$



$\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \Sigma)$

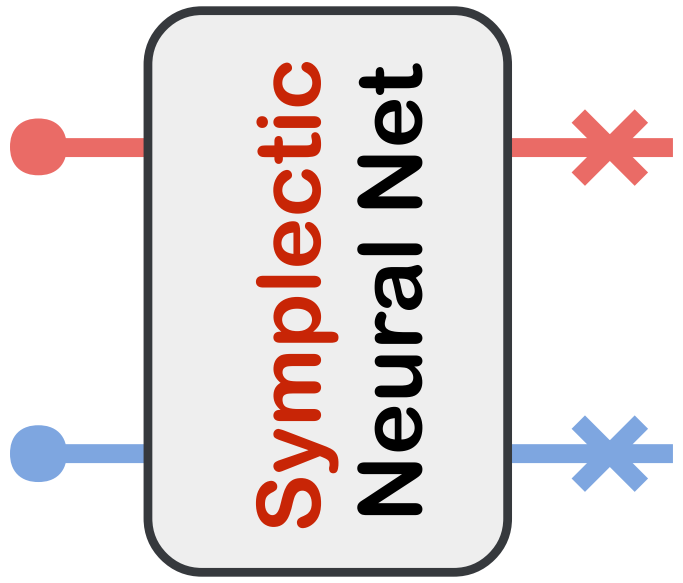
Sample in the latent space

Density estimation

"learn from data"

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

$\boldsymbol{x} \sim \text{dataset}$

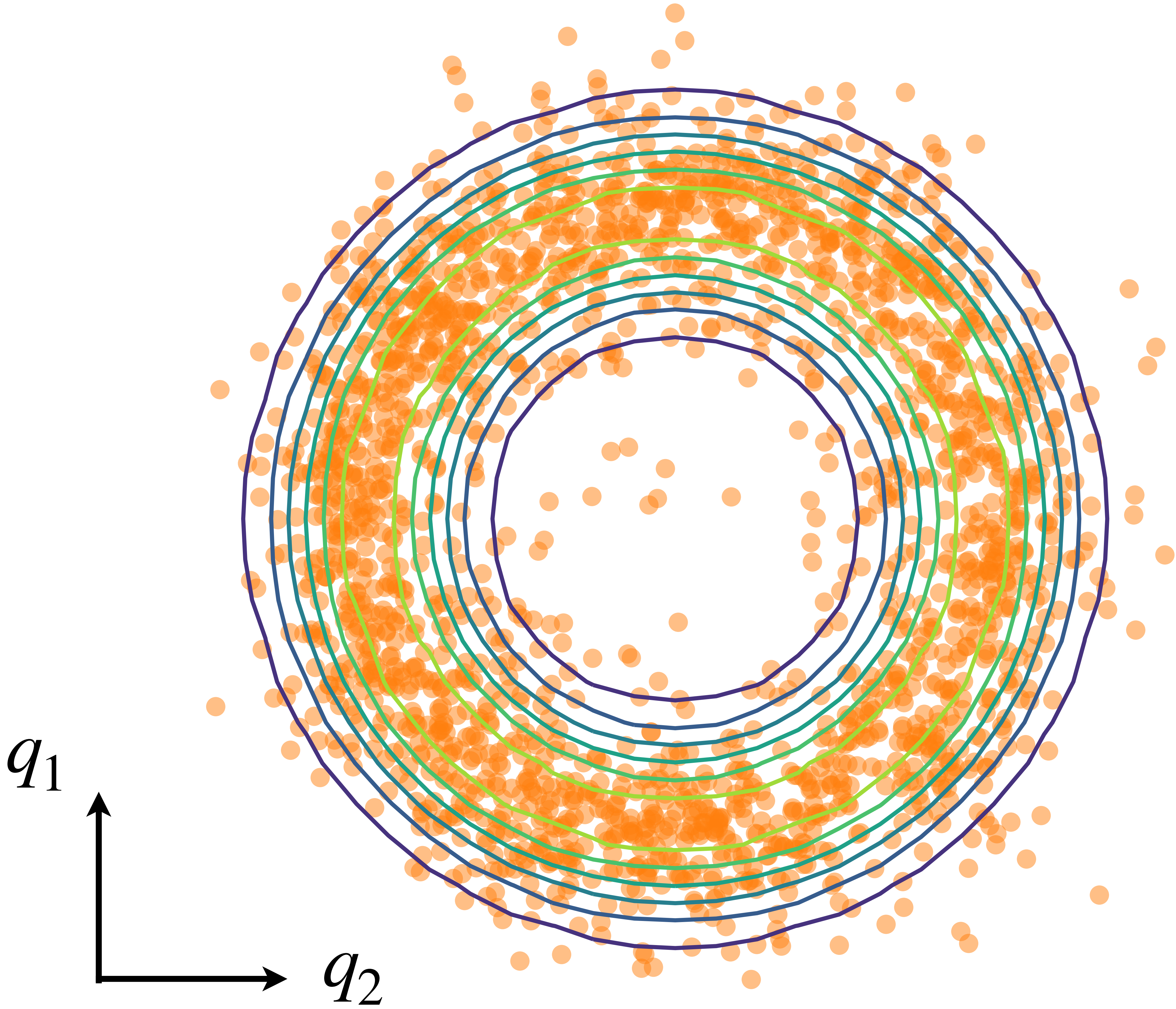


$\boldsymbol{z}$

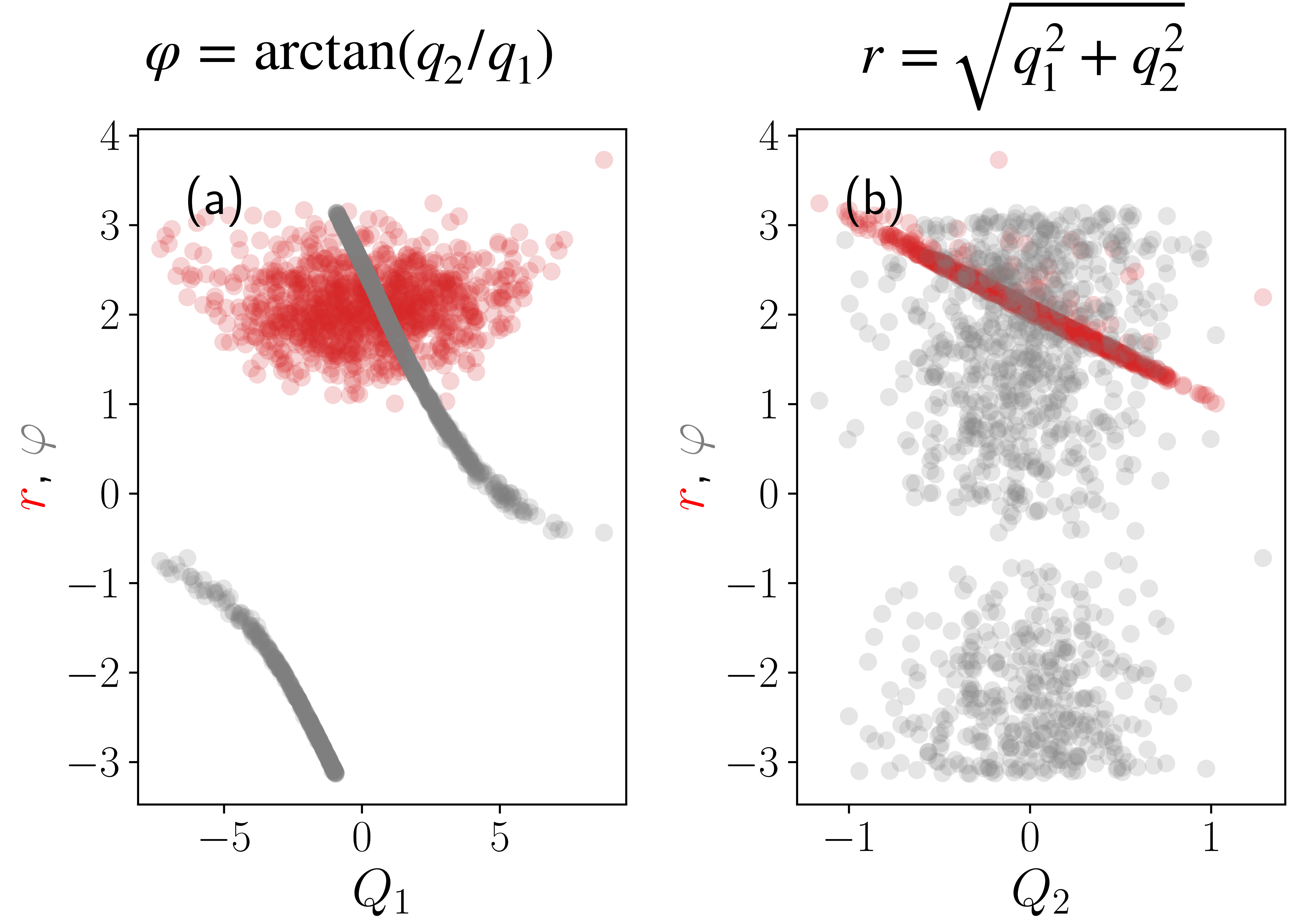
Sample from dataset in the physical space

Example 1: Ringworld

$$H = \frac{p_1^2 + p_2^2}{2} + V(q_1, q_2)$$



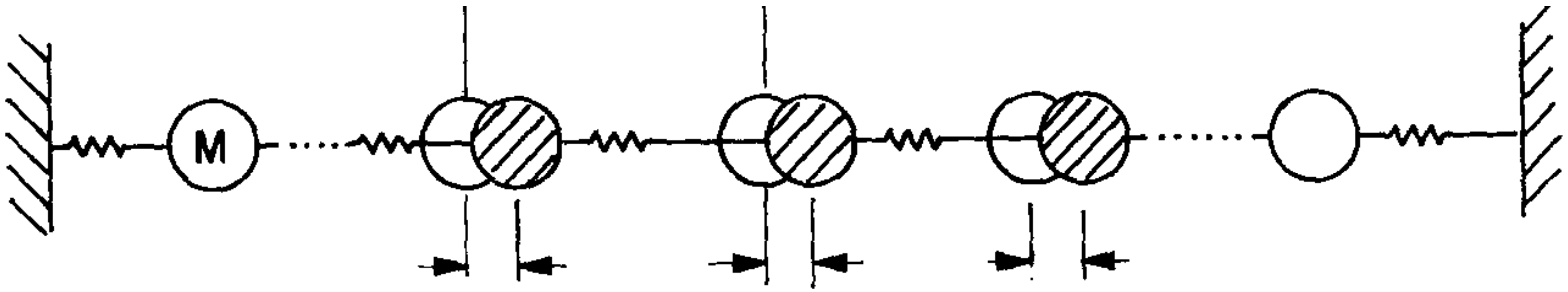
Learning the polar coordinates



Nonlinear coordinate transformation in 2d

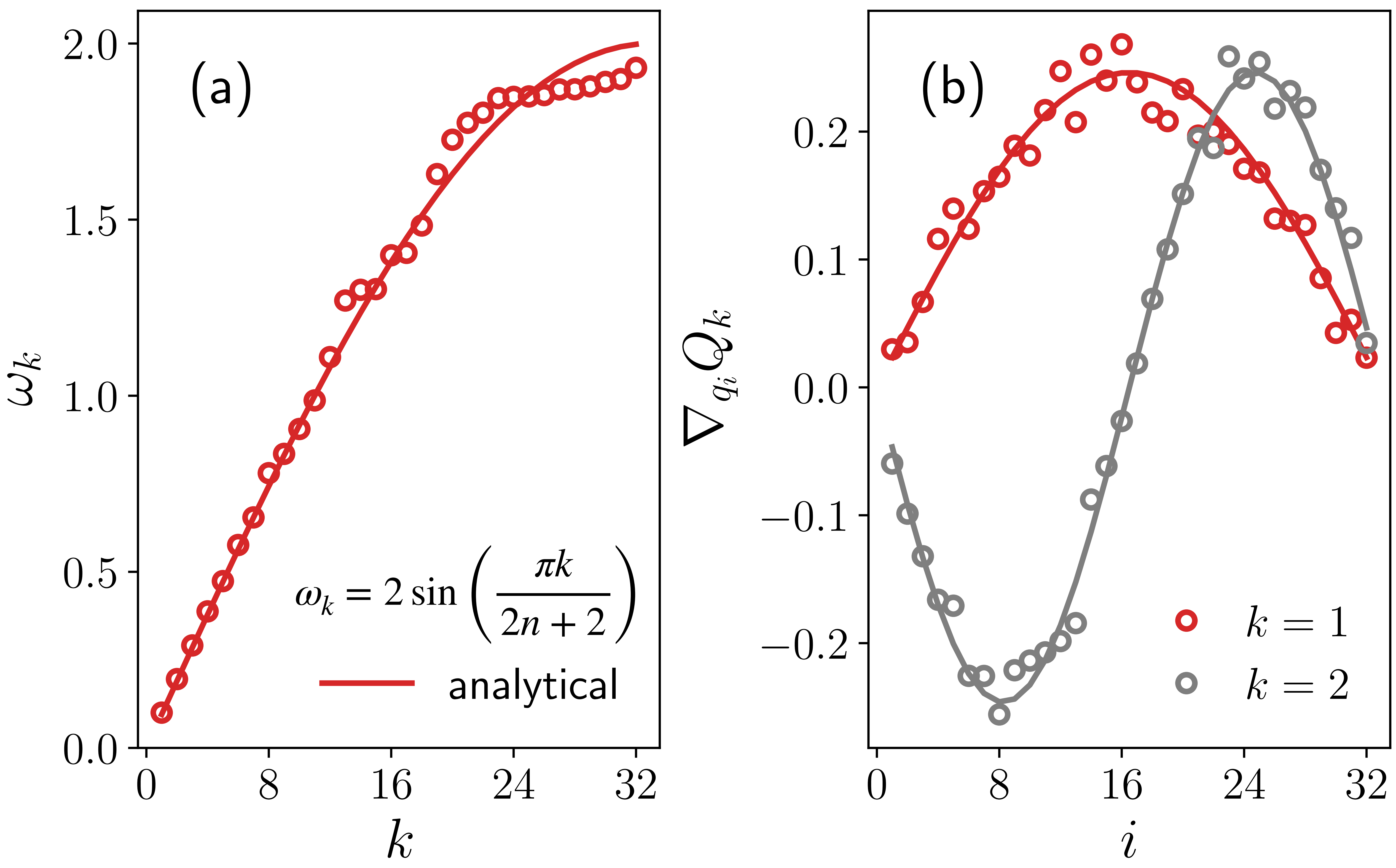
Example 2: Harmonic Chain

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



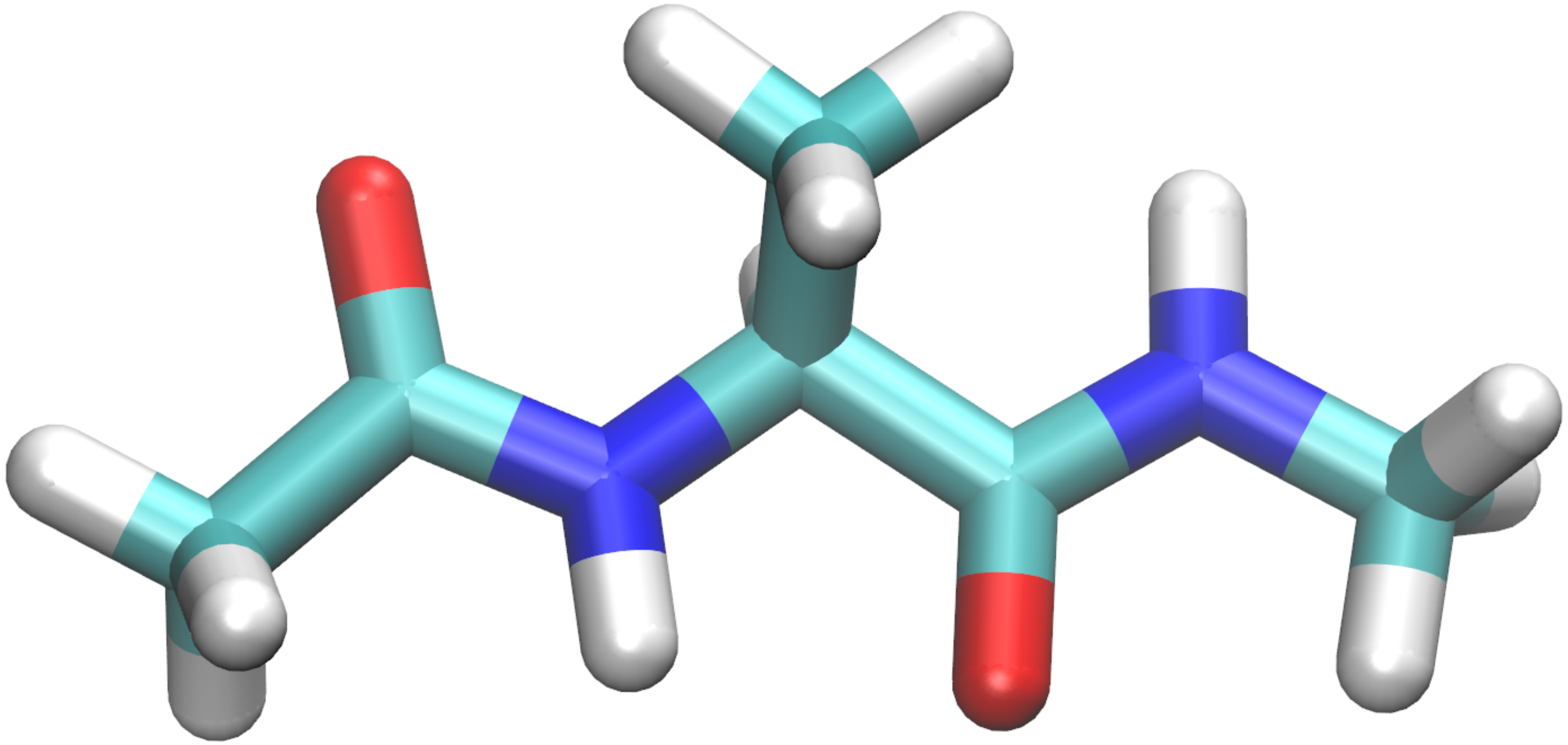
Fermi–Pasta–Ulam–Tsingou problem w/o nonlinearity

Learning normal modes



Linear coordinate transformation in high dim

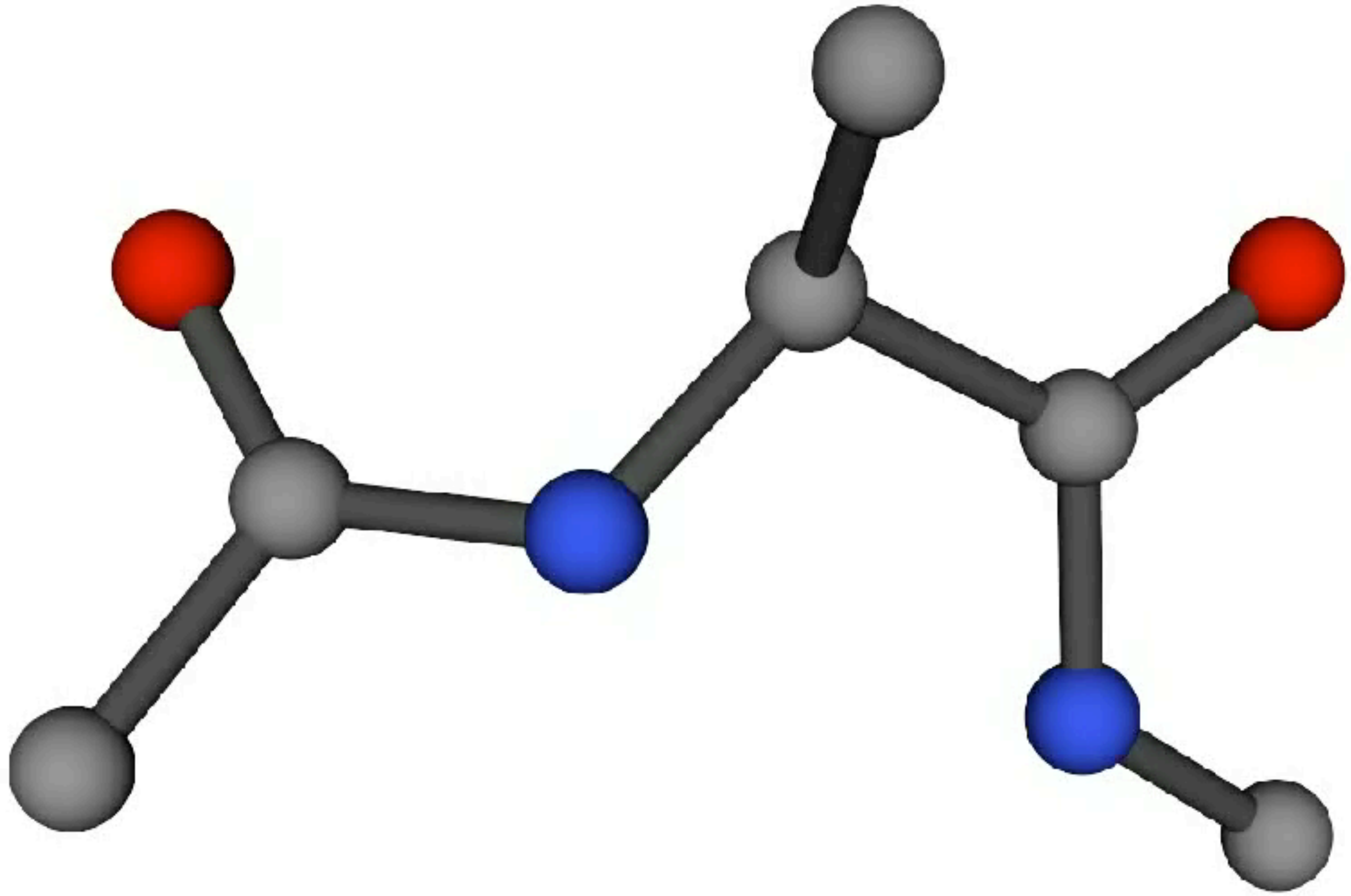
Example 3: Alanine Dipeptide



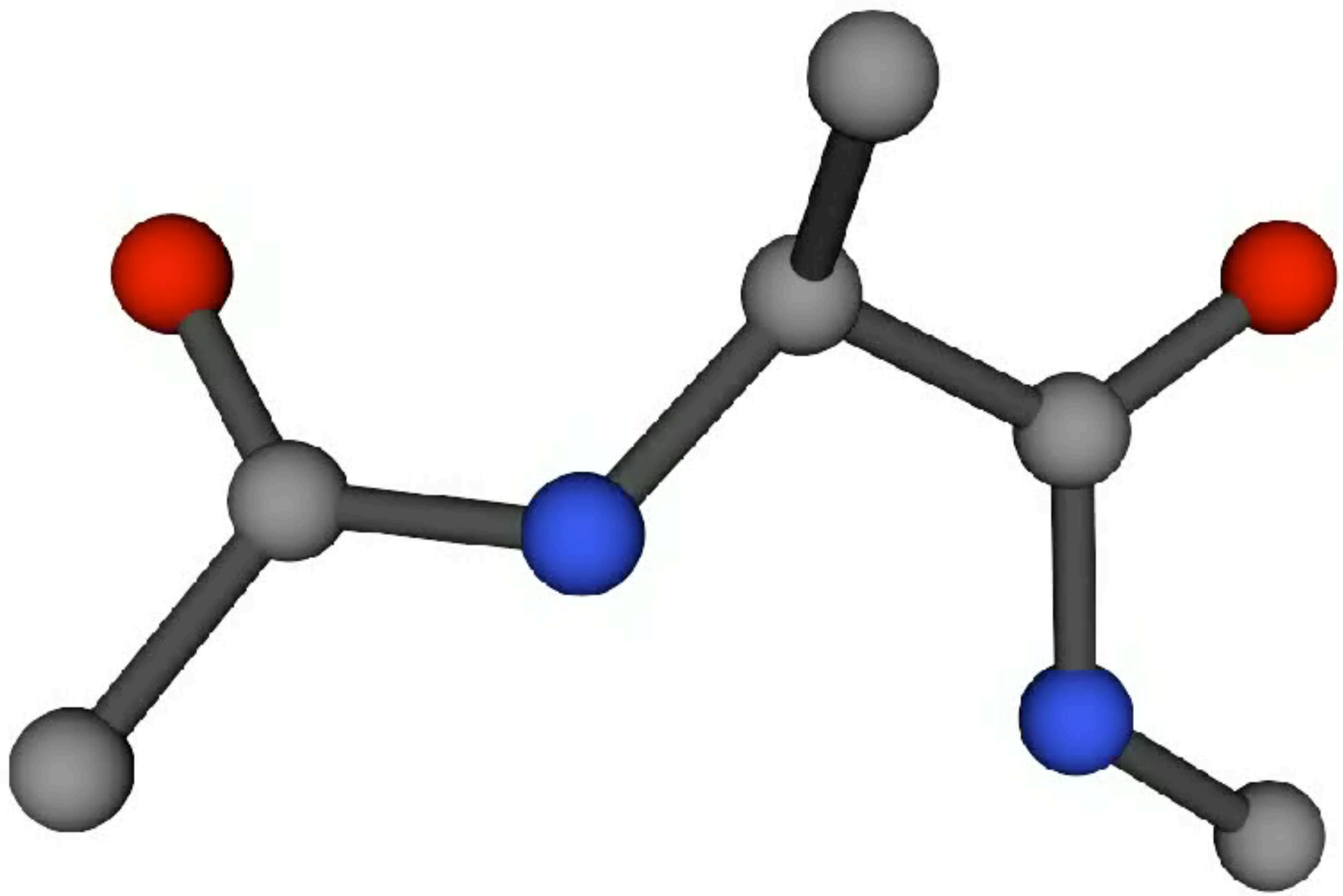
250 ns molecular dynamics simulation data at 300 K

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

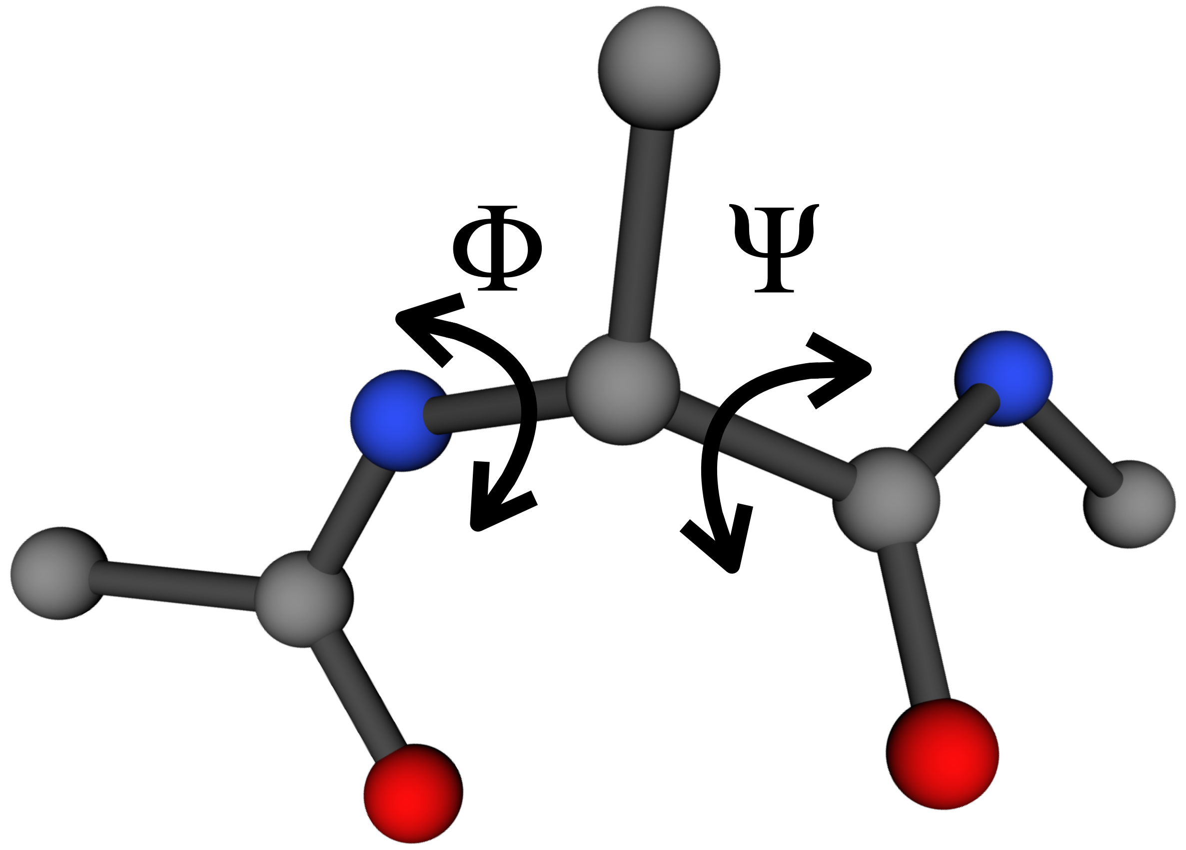
More than 3 hours of video …



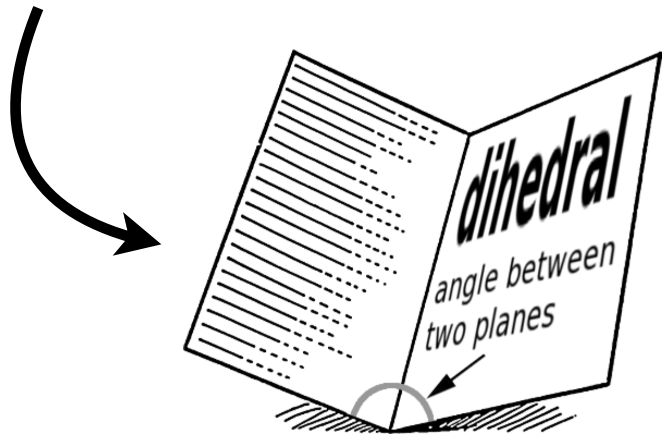
More than 3 hours of video …

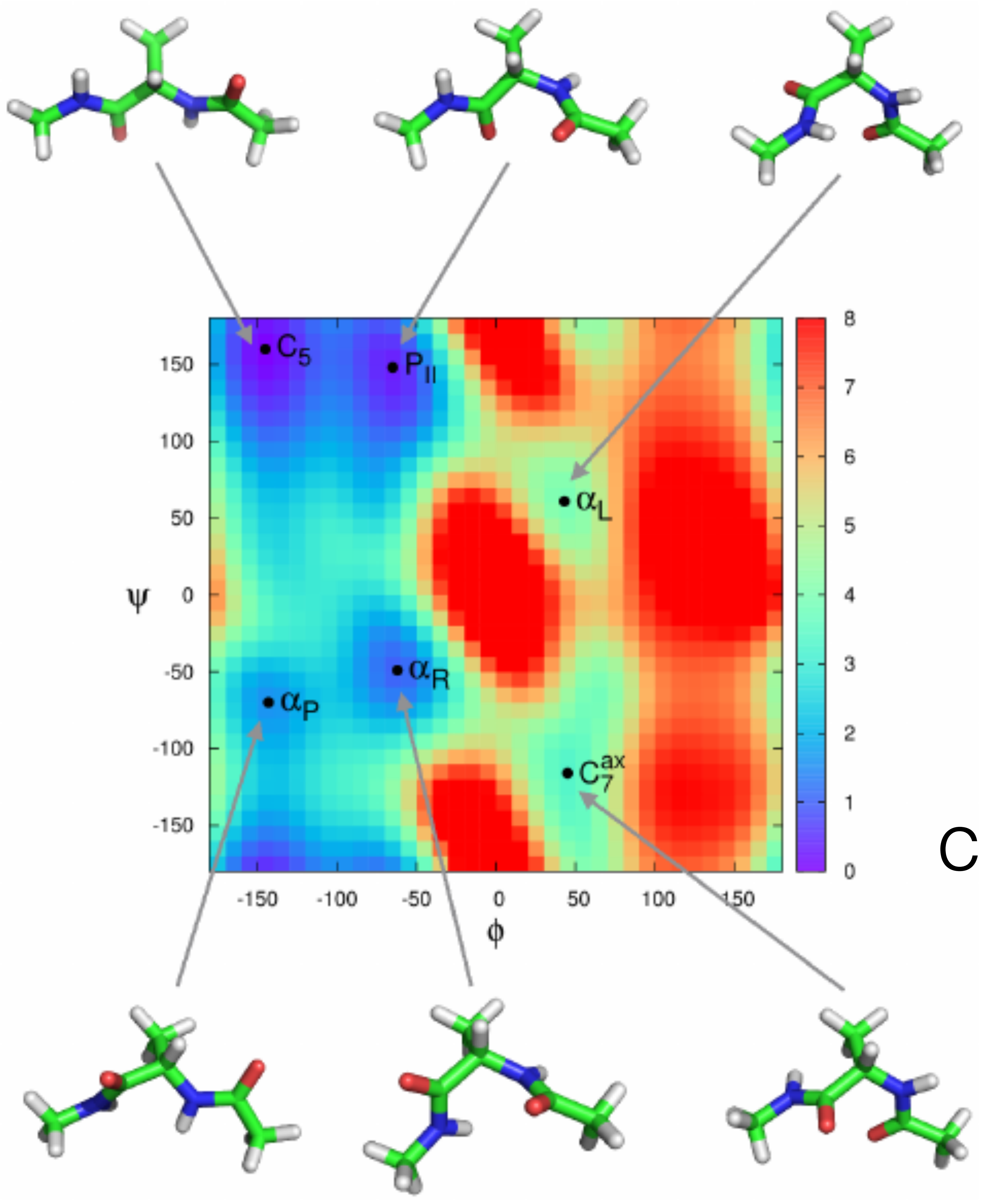


What do biologists see ?



slow motion of the
two torsion angles

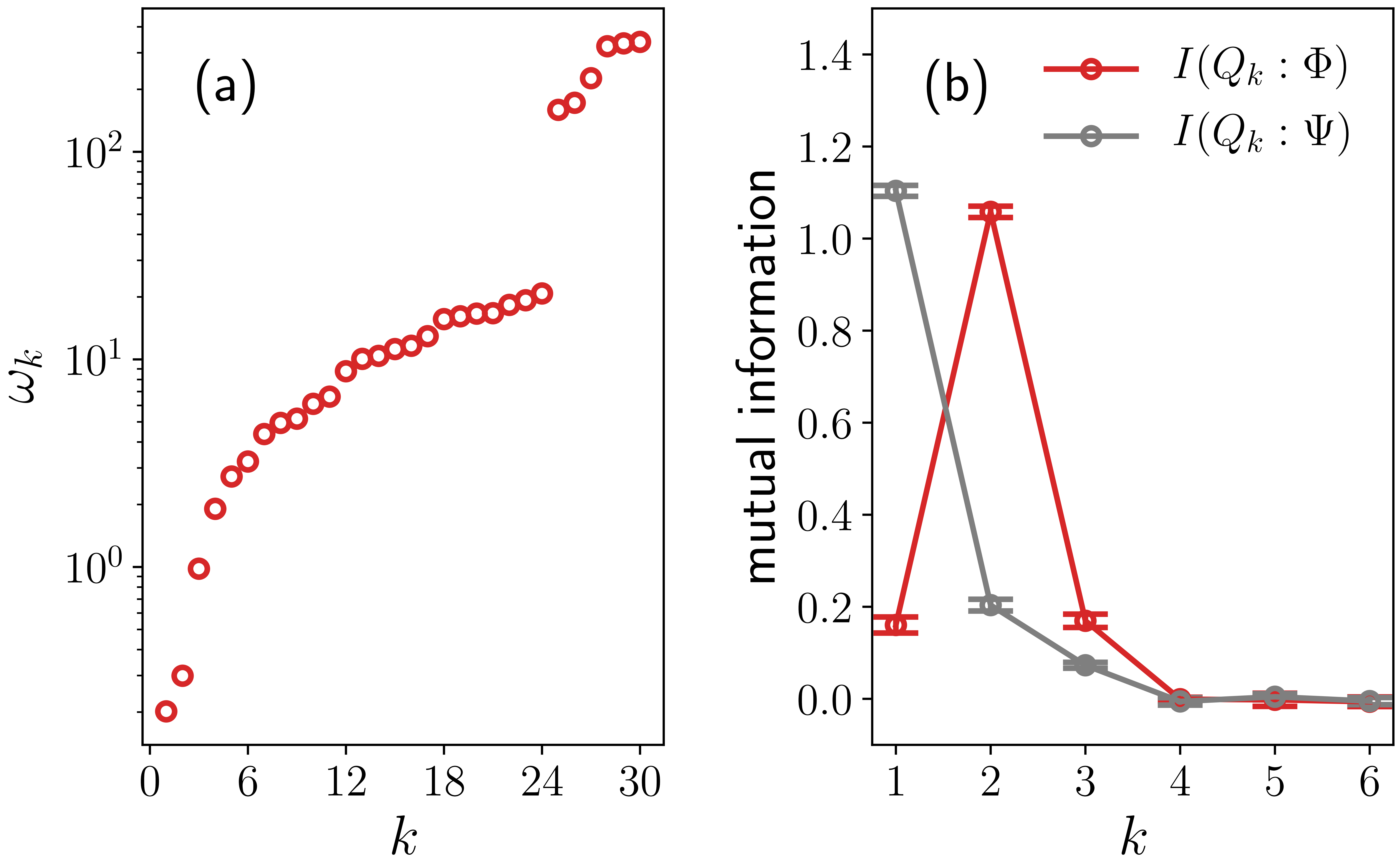




stable
conformations

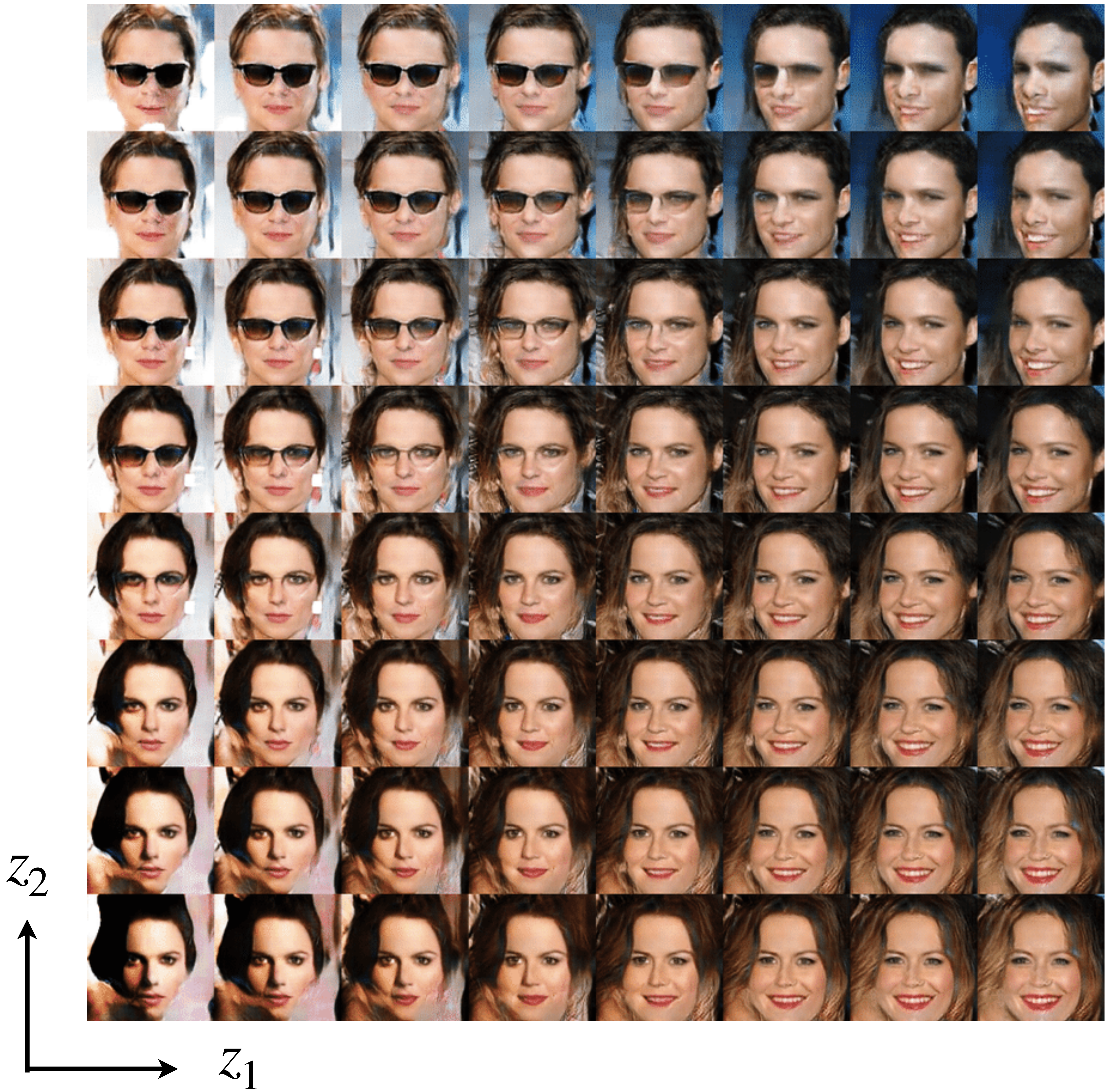
“Dimensional reduction” to manually designed collective variables

What does the neural net see ?

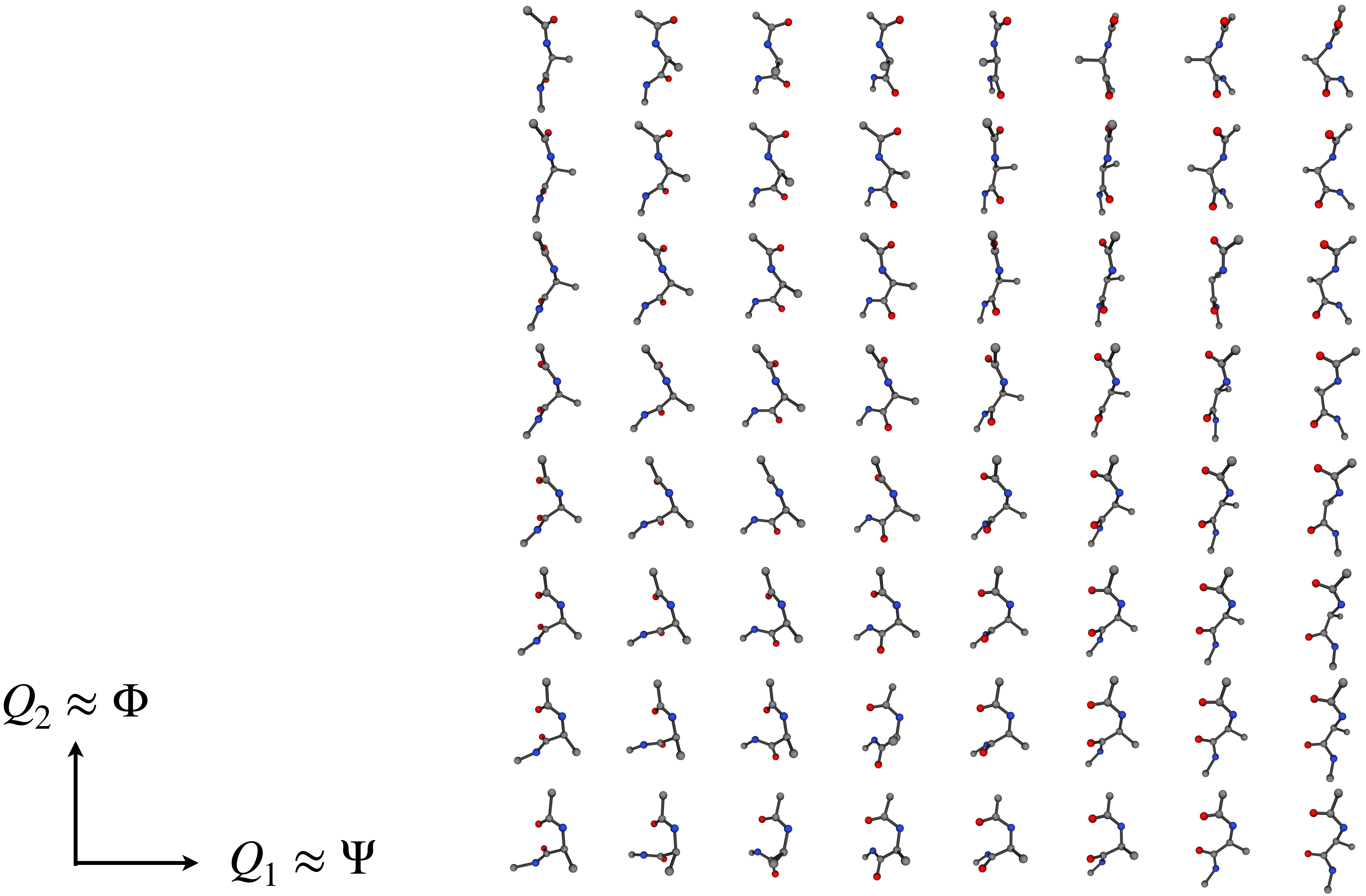


Unsupervised learning of slow & nonlinear collective variables from data

Latent space interpolation



Latent space interpolation



Example 4: MNIST

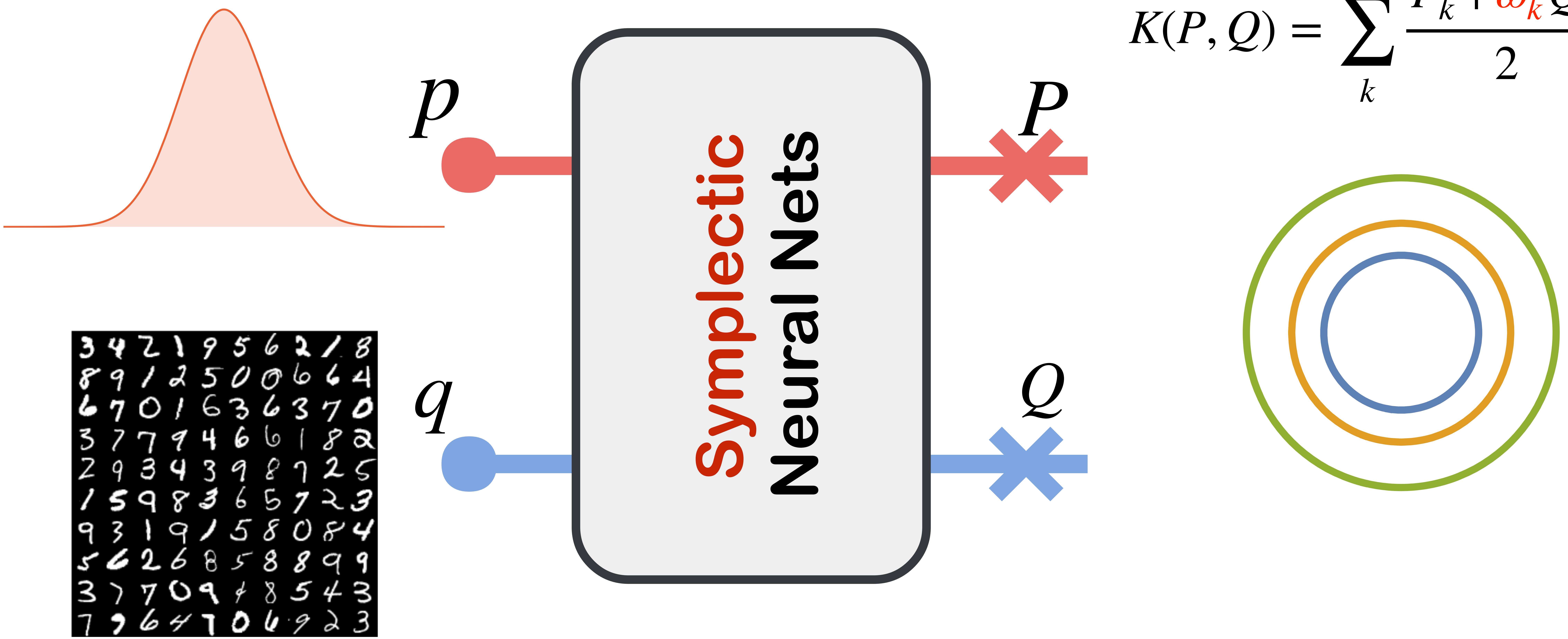
Data scientists:

“50,000 grayscale images with 28x28 pixels”

Physical Chemists:

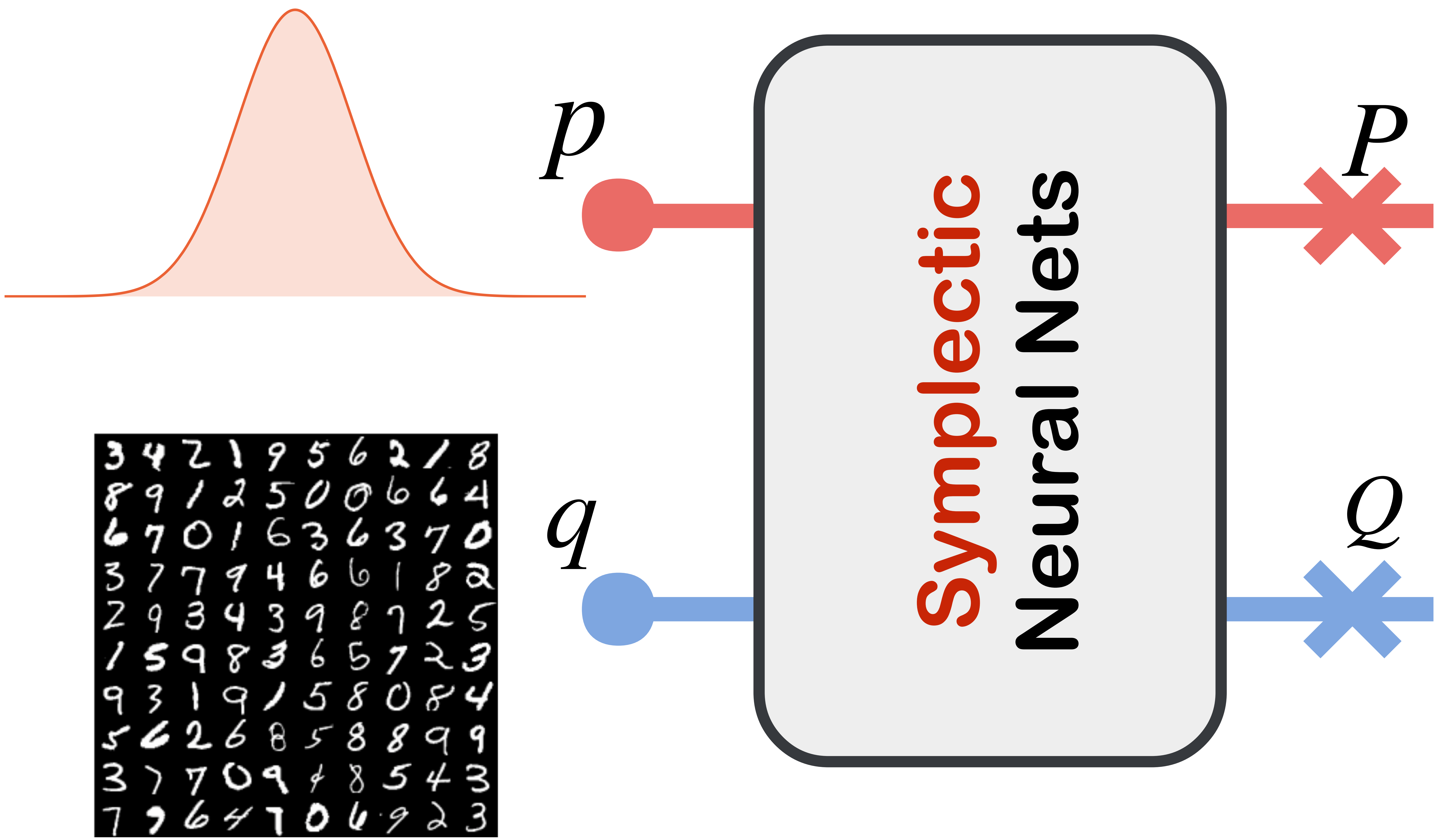
“Stable conformations of a molecule with 784 degrees of freedom”

Learning slow variables of MNIST

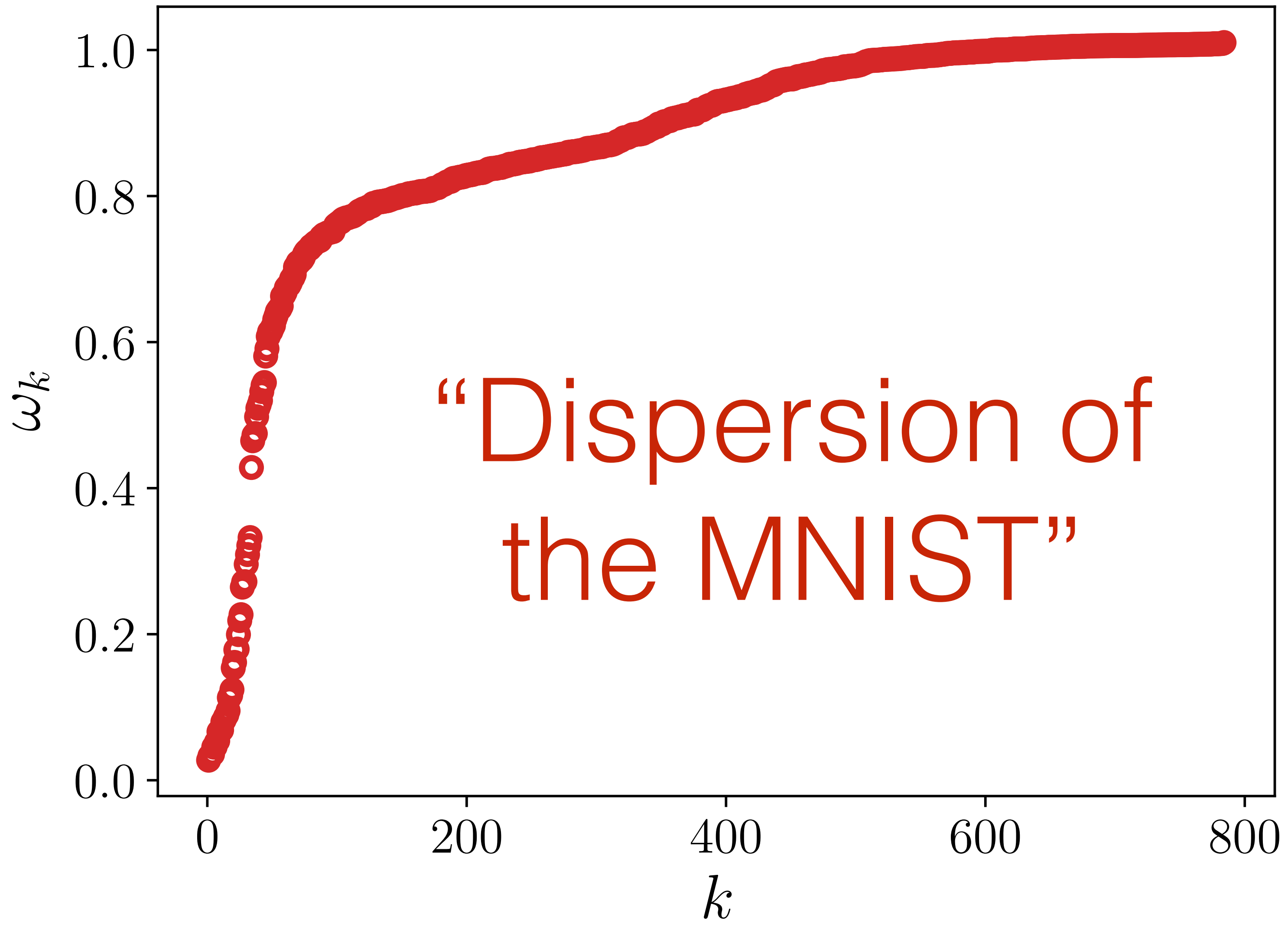


$$K(P,Q) = \sum_k \frac{P_k^2 + \omega_k^2 Q_k^2}{2}$$

Learning slow variables of MNIST

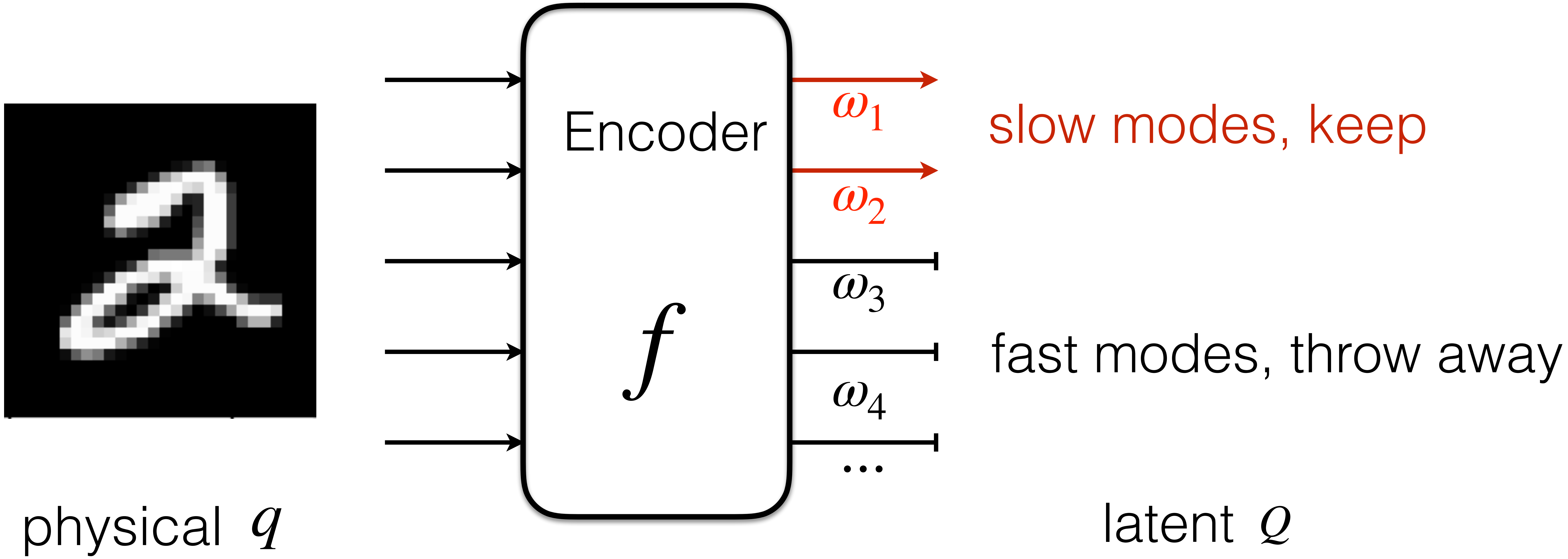


$$K(P,Q) = \sum_k \frac{P_k^2 + \omega_k^2 Q_k^2}{2}$$



Conceptual Compression

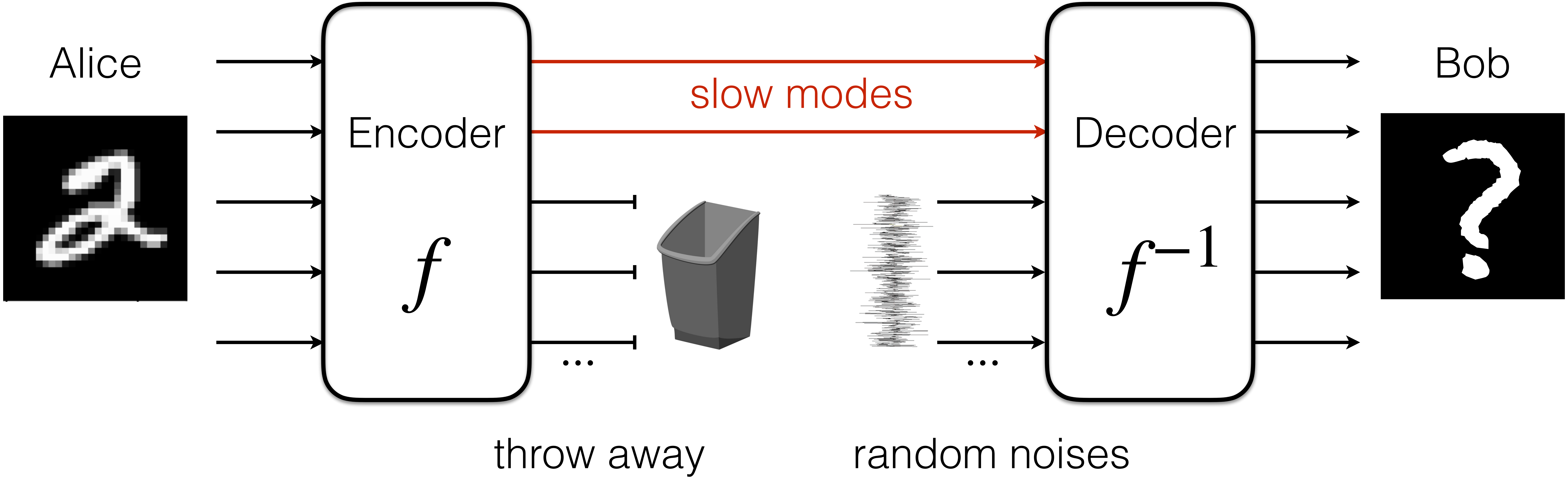
$\omega_1 < \omega_2 < \omega_3 < \omega_4 < \cdots$



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

To do list

- Even better understandings of the approach
- Actual applications to molecular dynamics and machine learning: Effective theory of slow modes, enhanced sampling, prediction…
- Applications to (near) integrable systems, nonlinear lattices with solitons, synchronization phenomena …
- More suggestions ?

Thank You!

Shuo-Hui Li 李烁辉
IOP CAS

Chen-Xiao Dong 董陈潇
IOP CAS

Linfeng Zhang 张林峰
Princeton