AI and Machine Learning in the Natural Sciences

The WASP Winter Conference 2022

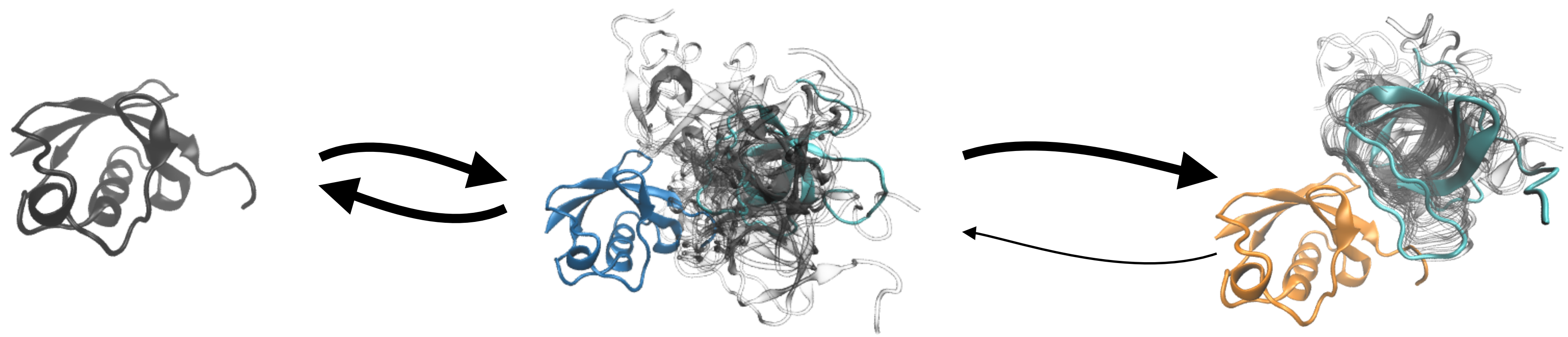
Simon Olsson
Data Science and AI, Computer Science and Engineering,
Chalmers University of Technology

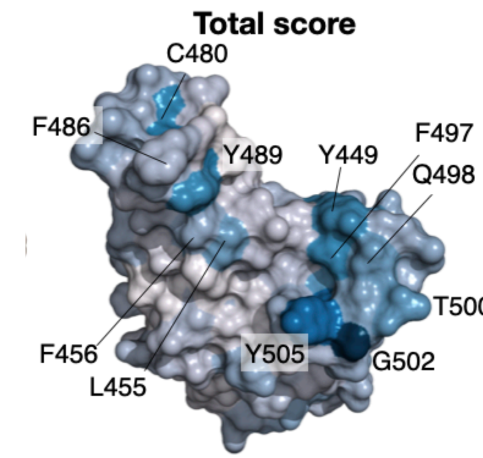
Virtual
11-13th Jan 2022

@smnlssn
simonols@chalmers.se

AI and Machine learning in the Natural Sciences — why?

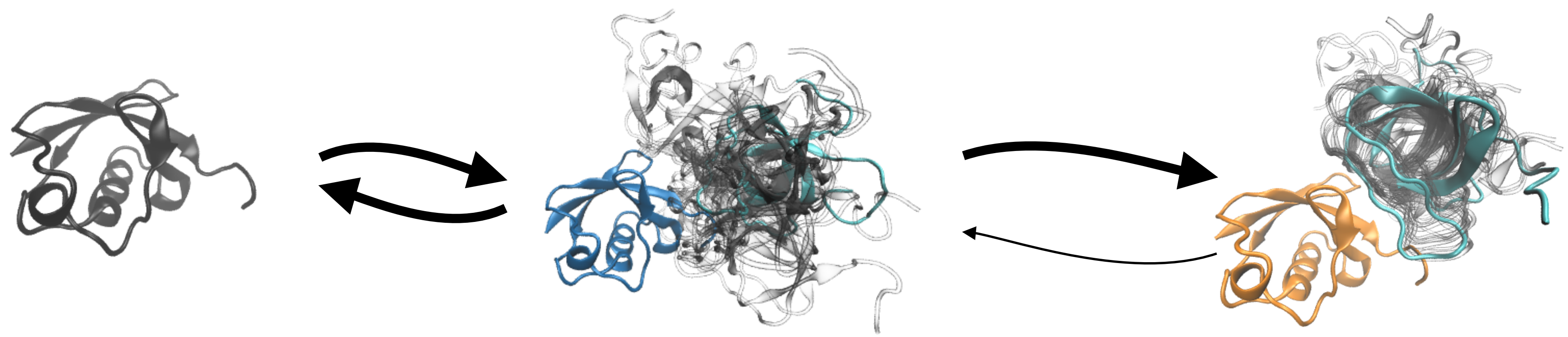
Exciting applications — Molecular Design

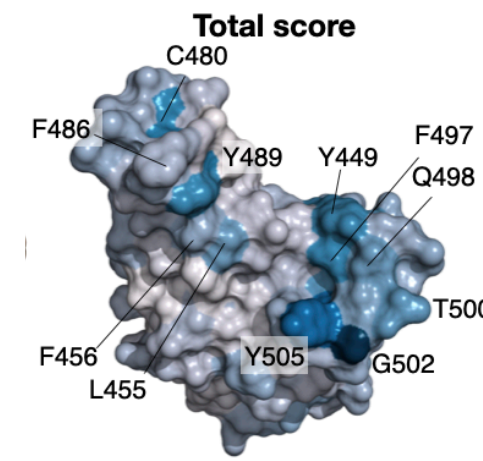




AI and Machine learning in the Natural Sciences — why?

Exciting applications — Molecular Design





Generative model

Drug, Material, Vaccine → **Property**

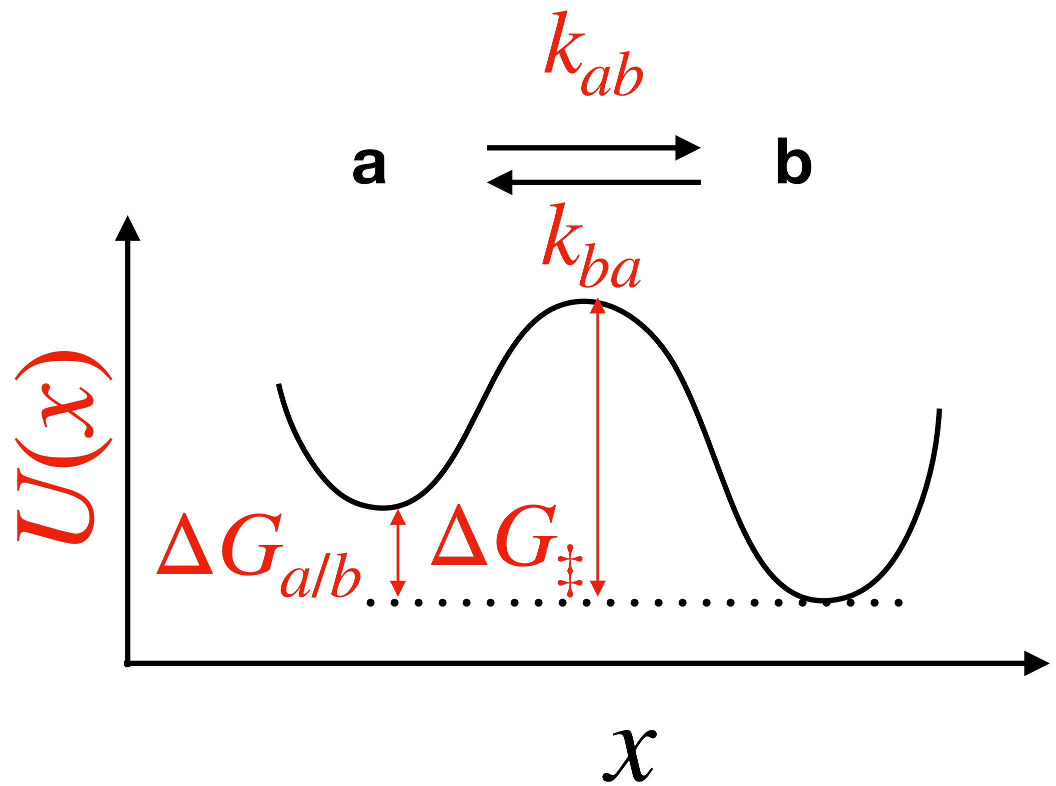
Tobias Karlsson

Sara RA, Juan VD, *[...]* <u>SO</u>, RM — ICML RL4RL Workshop (2021) doi: 10.33774/chemrxiv-2021-9w3tc

AI and Machine learning in the Natural Sciences — why?

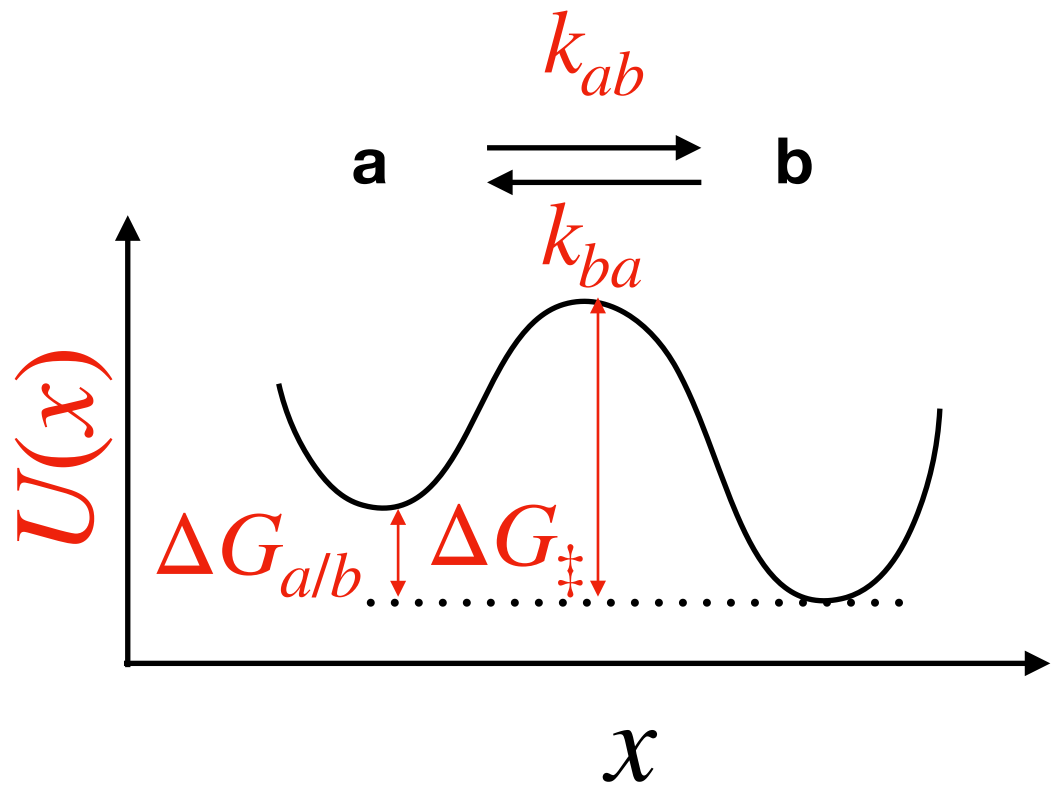
Exciting applications — Efficient simulations



→ **Dynamics/kinetics**

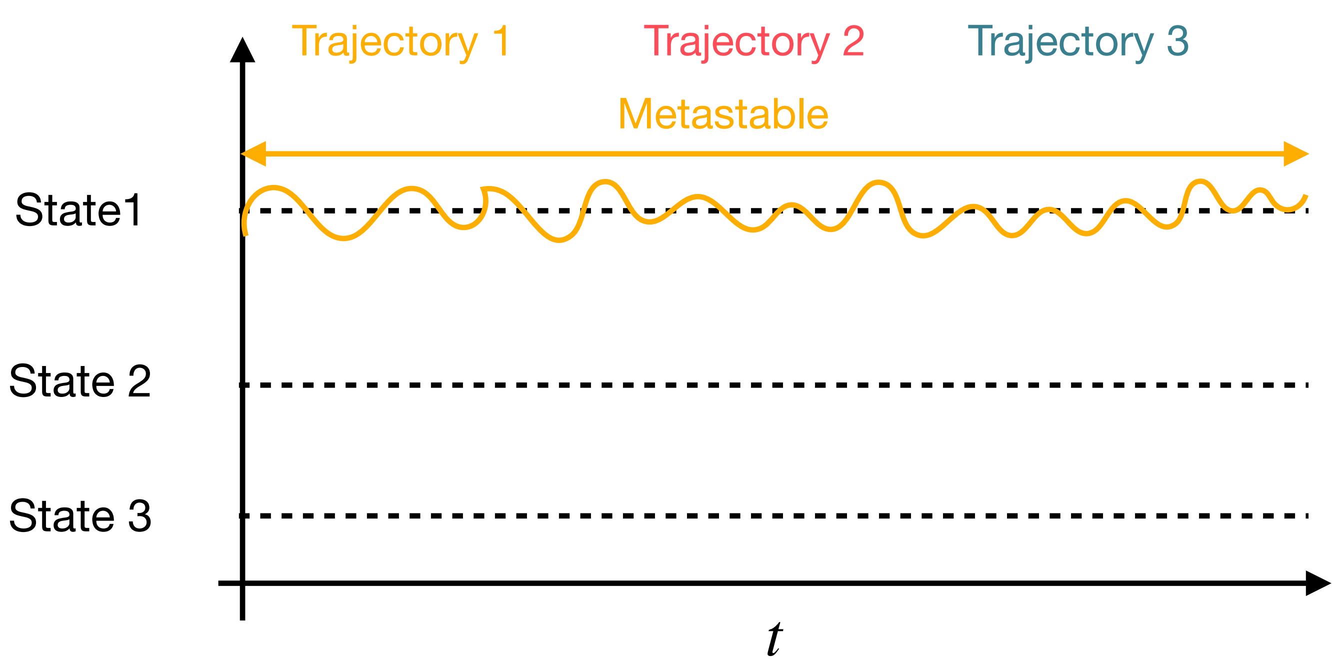
→ **Thermodynamics**

Exciting applications — Efficient simulations



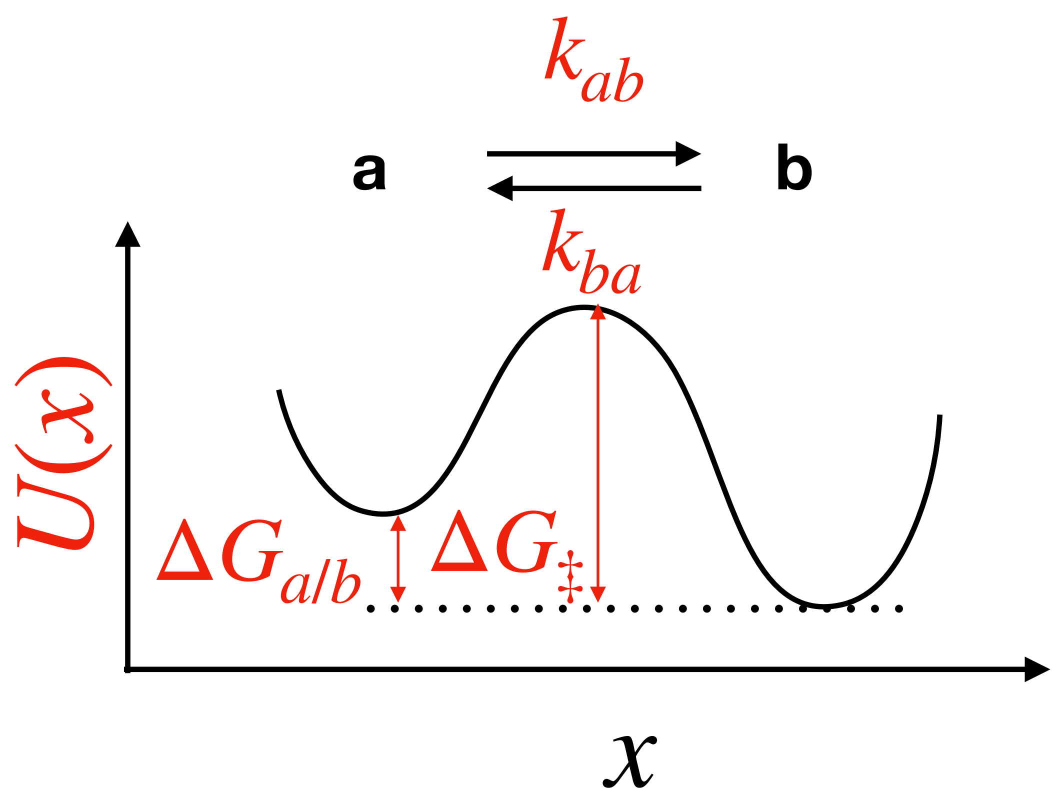
→ **Dynamics/kinetics**

→ **Thermodynamics**



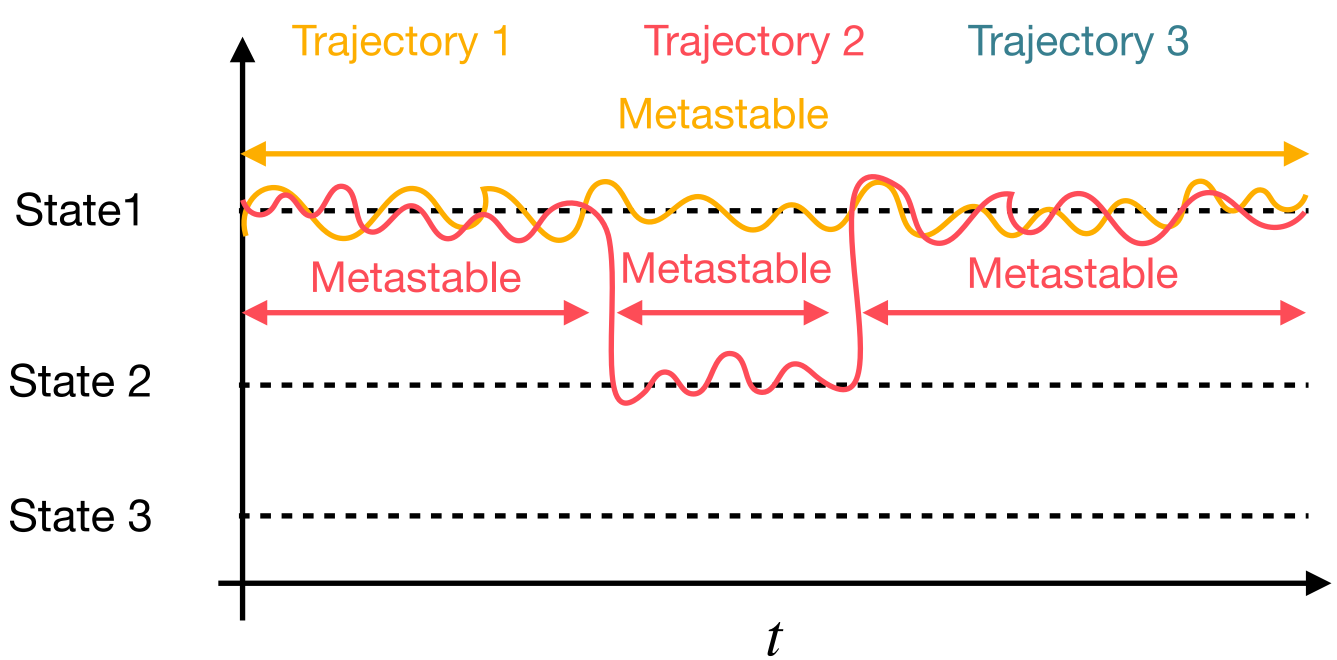
AI and Machine learning in the Natural Sciences — why?

Exciting applications — Efficient simulations



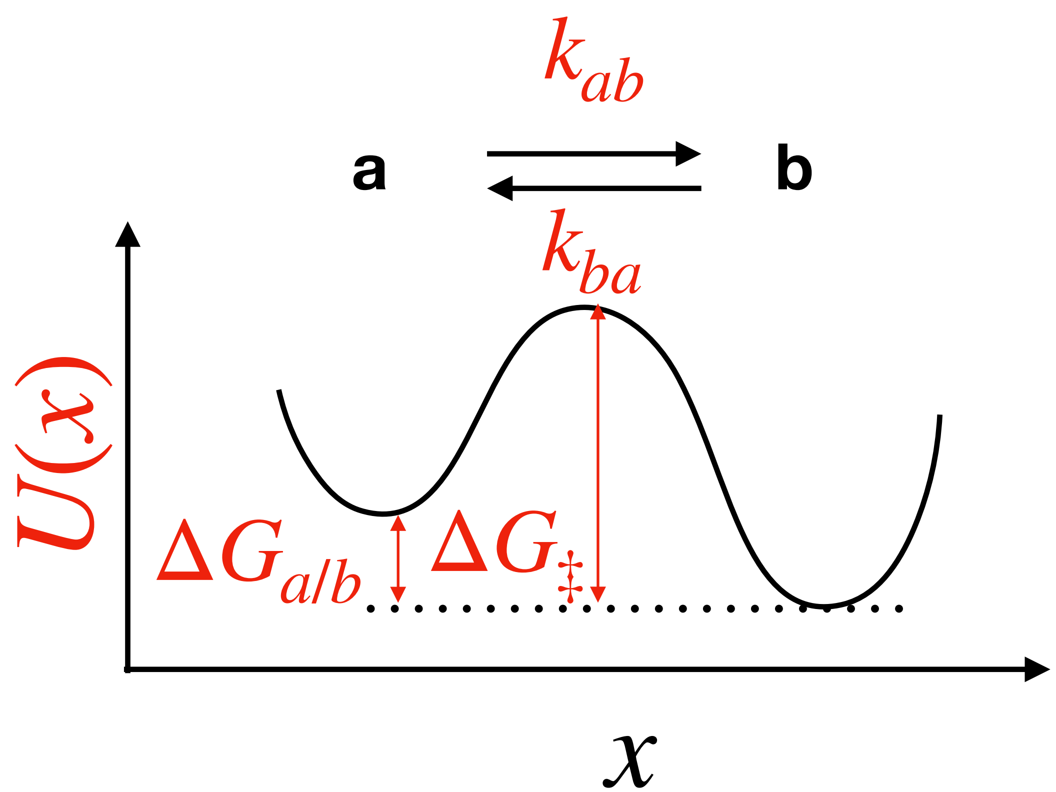
→ **Dynamics/kinetics**

→ **Thermodynamics**



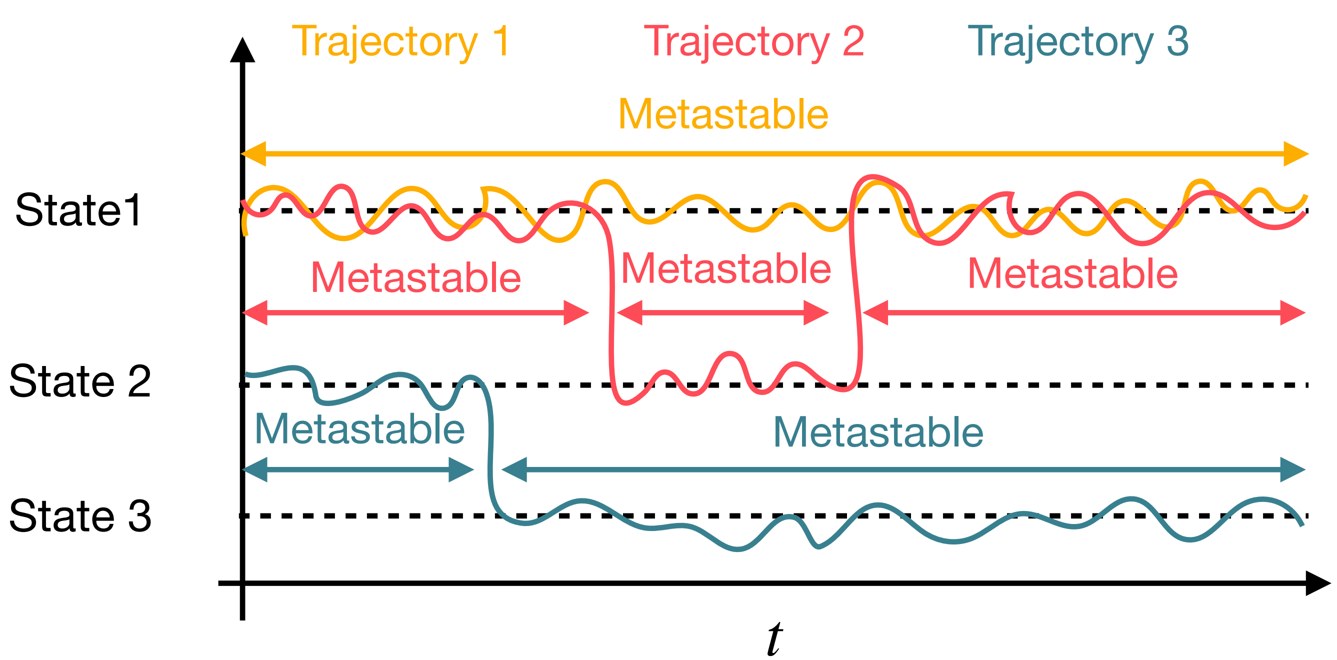
AI and Machine learning in the Natural Sciences — why?

Exciting applications — Efficient simulations



→ **Dynamics/kinetics**

→ **Thermodynamics**



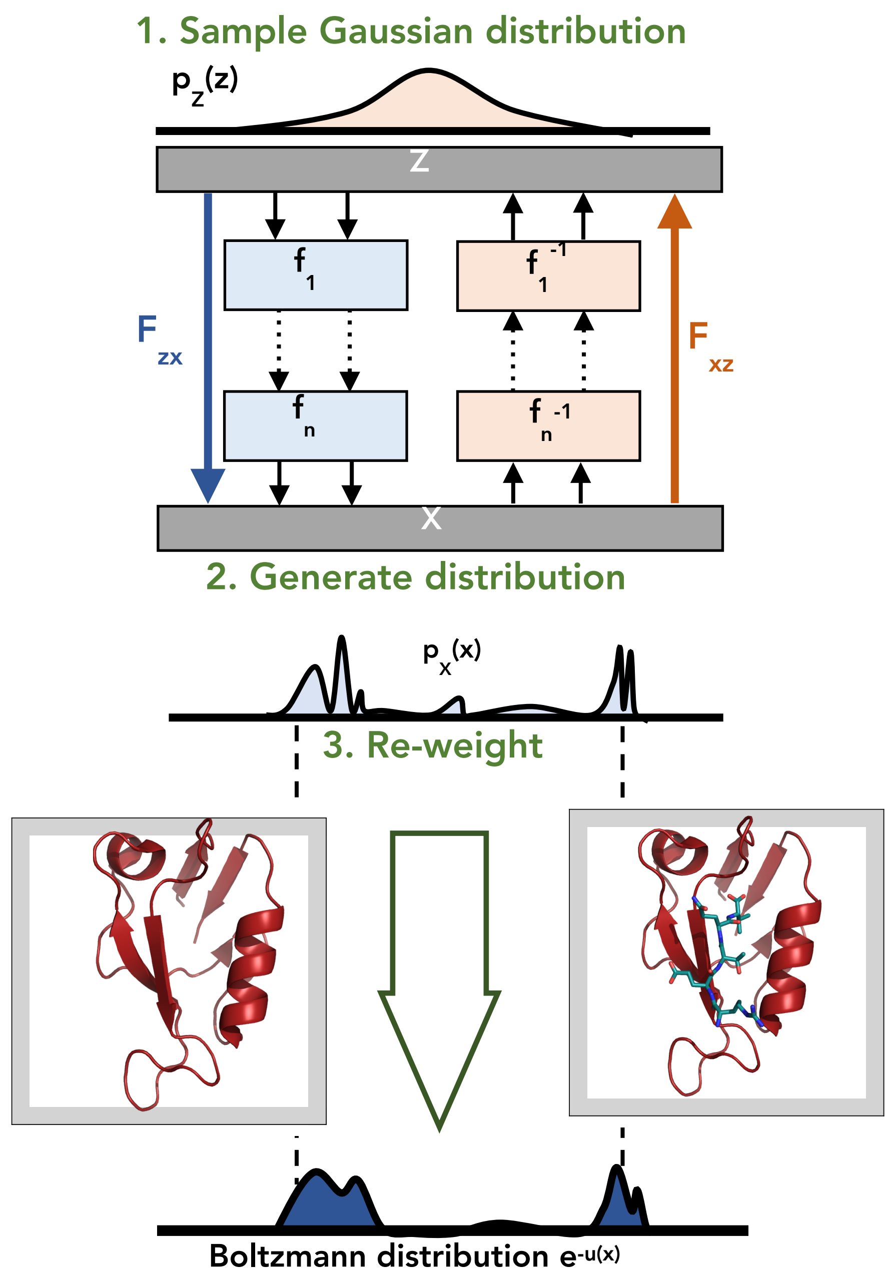
<u>Non-IID data</u>

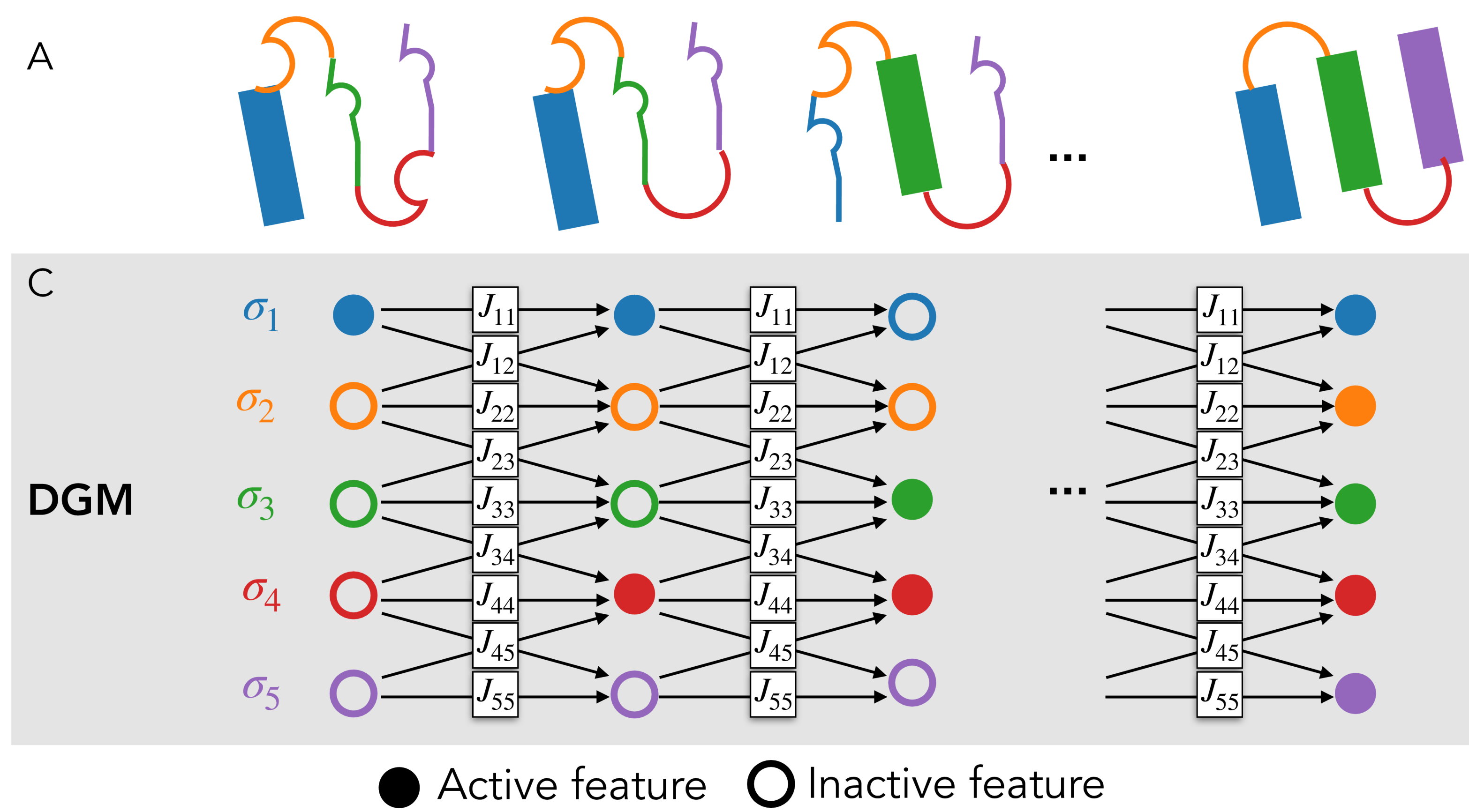
Juan Viguera Diez

Christopher Kolloff

AI and Machine learning in the Natural Sciences — why?

Exciting applications — Efficient simulations





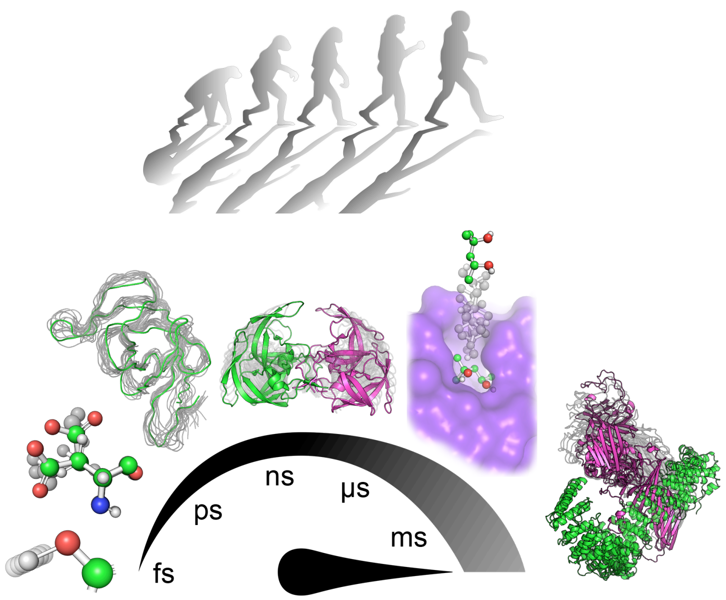
Noé F*†, **Olsson S†,** Köhler J†, Wu H. Science (2019)**.** 365, eaaw1147. doi:10.1126/science.aaw1147.

Olsson & *Noé* Proc. Natl. Acad. Sci. U.S.A. (2019), 116(30), pp. 15001-15006

AI and Machine learning in the Natural Sciences — why?

Challenging theory

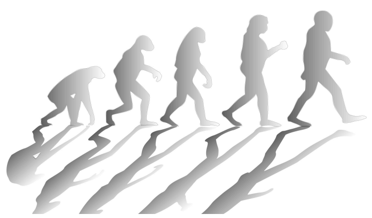
Generative processes

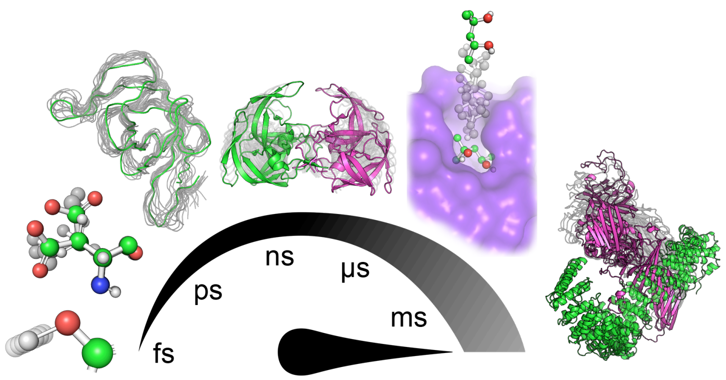


AI and Machine learning in the Natural Sciences — why?

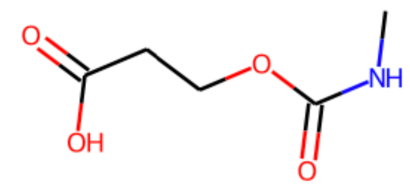
Challenging theory

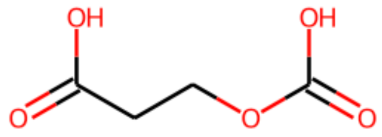
Generative processes

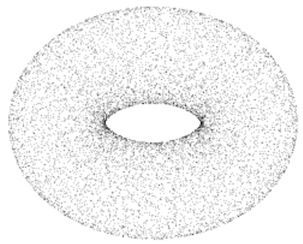




Non-Euclidean data



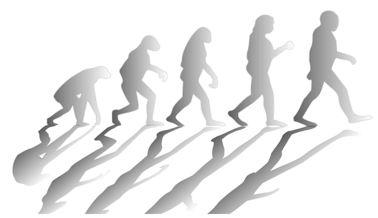


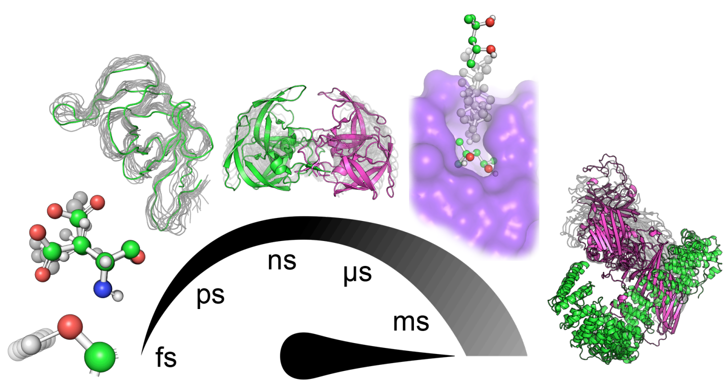


AI and Machine learning in the Natural Sciences — why?

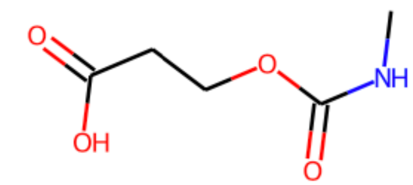
Challenging theory

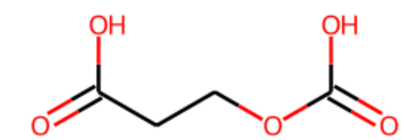
Generative processes

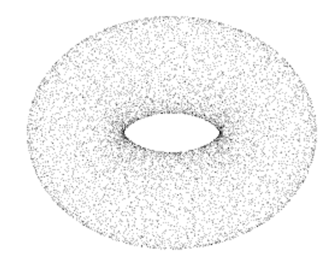




Non-Euclidean data







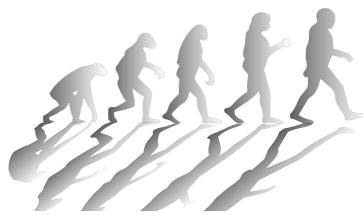
Domain adaptation/transfer learning

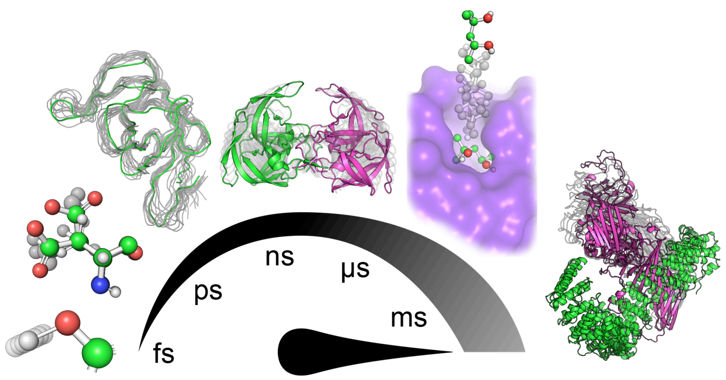
$$\mathbb{E}_{x\sim\mathcal{M}_1}[\,|f(x,M_1)-M_1(x)|_2] \approx \mathbb{E}_{x\sim\mathcal{M}_2}[\,|f(x,M_2)-M_2(x)|_2]$$

AI and Machine learning in the Natural Sciences — why?

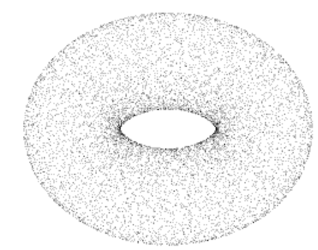
Challenging theory

Generative processes





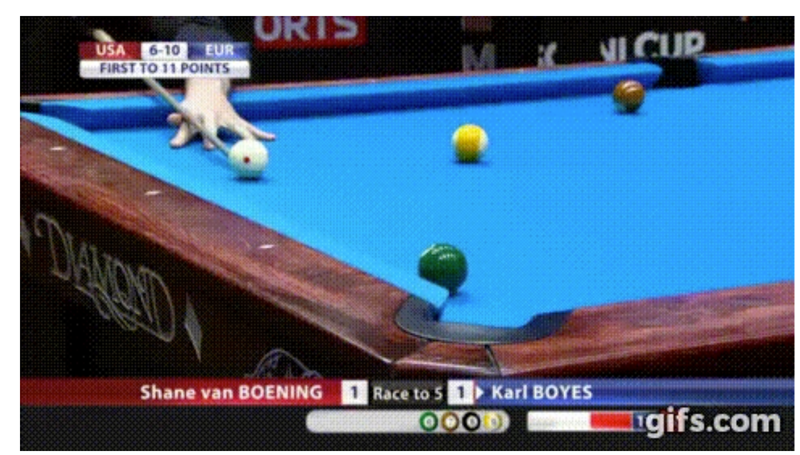
Non-Euclidean data



Domain adaptation/transfer learning

$$\mathbb{E}_{x\sim\mathcal{M}_1}[\,|f(x,M_1)-M_1(x)|_2] \approx \mathbb{E}_{x\sim\mathcal{M}_2}[\,|f(x,M_2)-M_2(x)|_2]$$

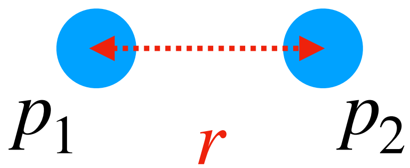
Causal modeling



What do know about our data?

Potential energy:

$$U(p_1, p_2) = (x_1 - x_2)^2 + (y_1 - y_2)^2$$
$$= r^2$$



Force:

$$F = -\nabla U(p_1, p_2)$$
$$= -\left[\left(\frac{\partial U}{x_1}, \frac{\partial U}{y_1}\right), \left(\frac{\partial U}{x_2}, \frac{\partial U}{y_2}\right)\right]$$

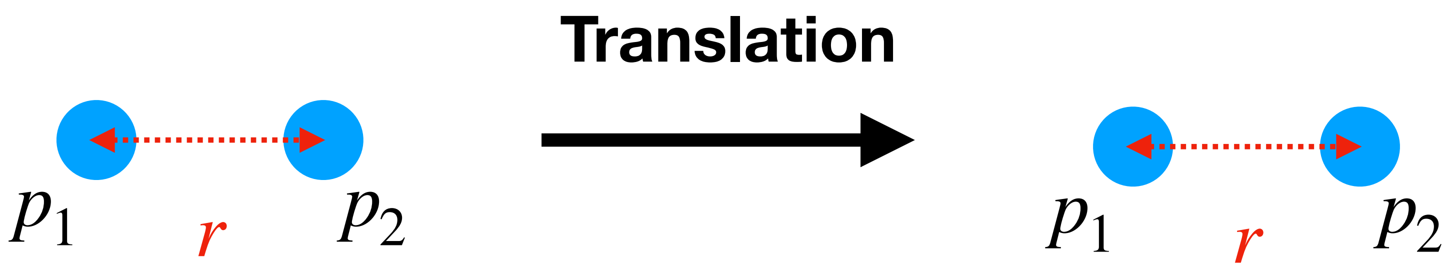
What do know about our data?

Potential energy:

$$U(p_1, p_2) = (x_1 - x_2)^2 + (y_1 - y_2)^2$$
$$= r^2$$

Force:

$$F = -\nabla U(p_1, p_2)$$
$$= -\left[\left(\frac{\partial U}{x_1}, \frac{\partial U}{y_1}\right), \left(\frac{\partial U}{x_2}, \frac{\partial U}{y_2}\right)\right]$$



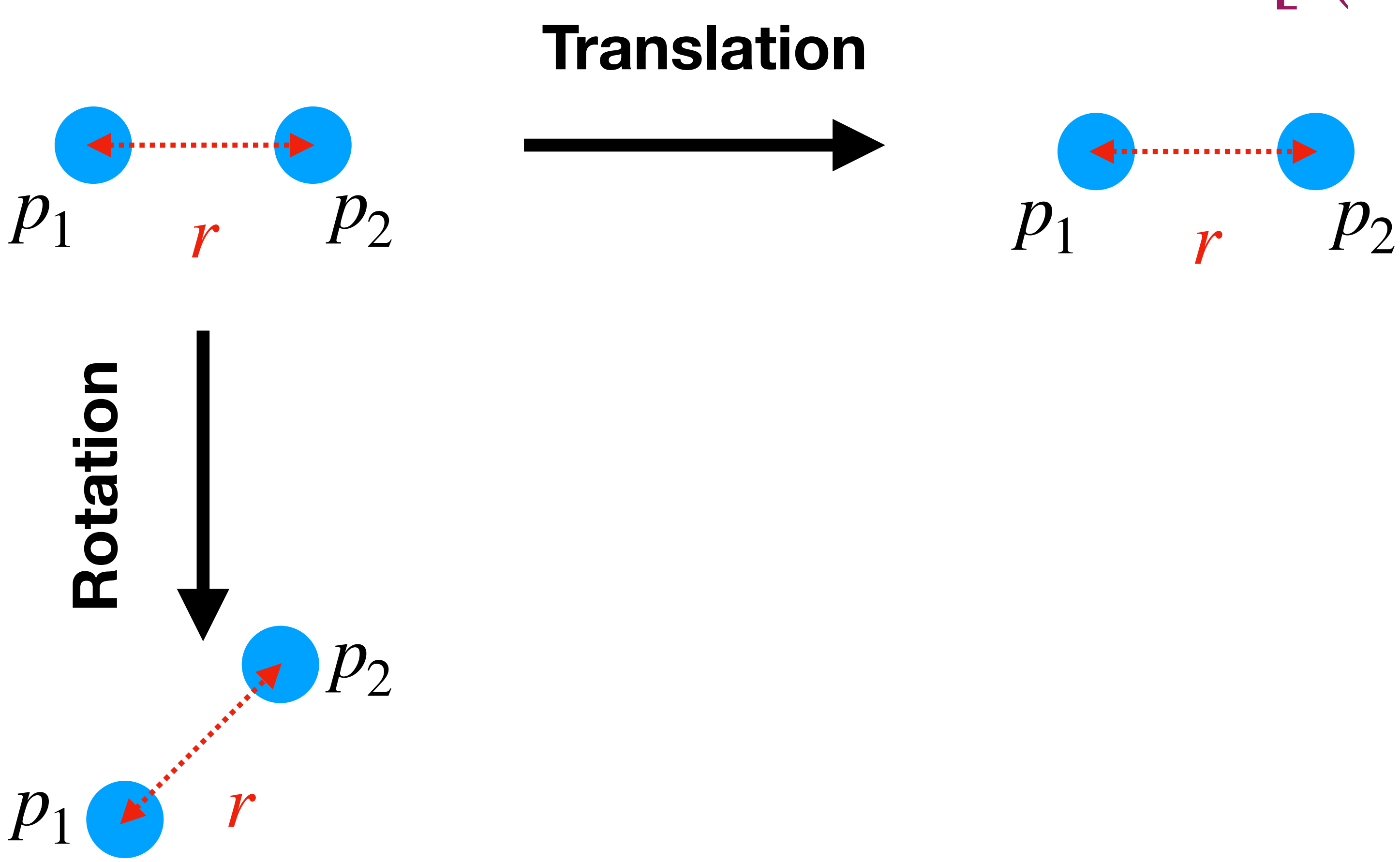
What do know about our data?

Potential energy:

$$U(p_1, p_2) = (x_1 - x_2)^2 + (y_1 - y_2)^2$$
$$= r^2$$

Force:

$$F = -\nabla U(p_1, p_2)$$
$$= -\left[\left(\frac{\partial U}{x_1}, \frac{\partial U}{y_1}\right), \left(\frac{\partial U}{x_2}, \frac{\partial U}{y_2}\right)\right]$$



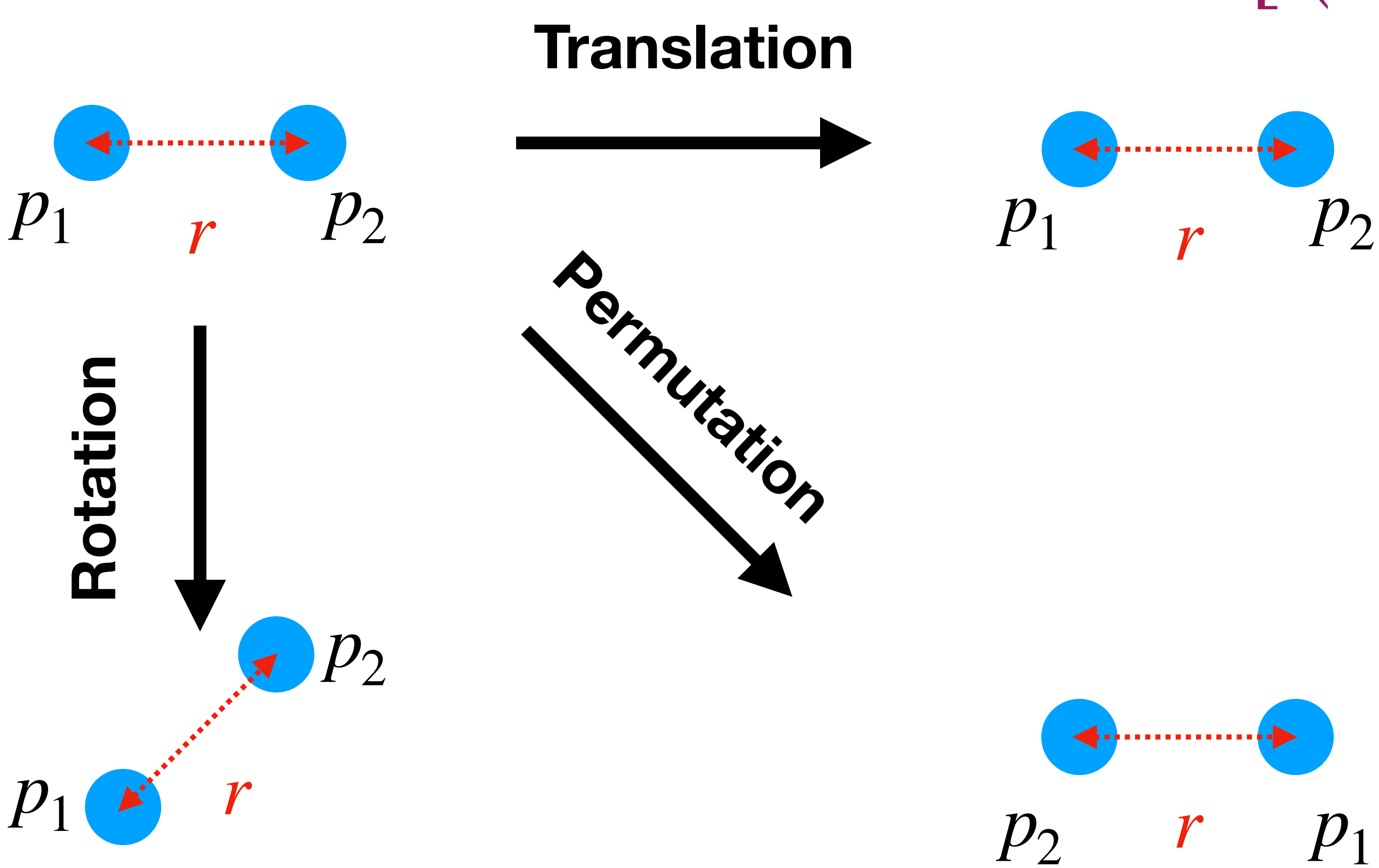
What do know about our data?

Potential energy:

$$U(p_1, p_2) = (x_1 - x_2)^2 + (y_1 - y_2)^2$$
$$= r^2$$

Force:

$$F = -\nabla U(p_1, p_2)$$
$$= -\left[\left(\frac{\partial U}{x_1}, \frac{\partial U}{y_1}\right), \left(\frac{\partial U}{x_2}, \frac{\partial U}{y_2}\right)\right]$$



What happens to the potential energy — and what about the force?

Noethers theorem — informally

"If a system has a continuous symmetry property, then there are corresponding quantities whose values are conserved in time."

Emmy Noether

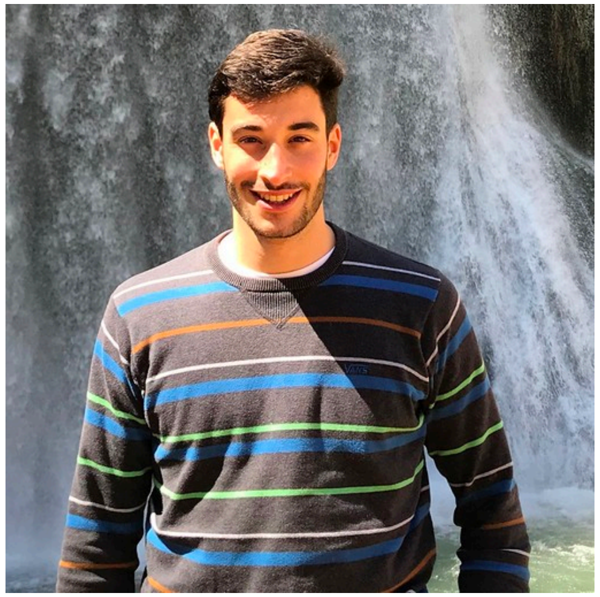
Noethers theorem — informally

"If a system has a continuous symmetry property, then there are corresponding quantities whose values are conserved in time."

Symmetry ⇔ Conservation Law

Emmy Noether

A transferable Boltzmann Generator for small molecules



Juan Viguera Diez

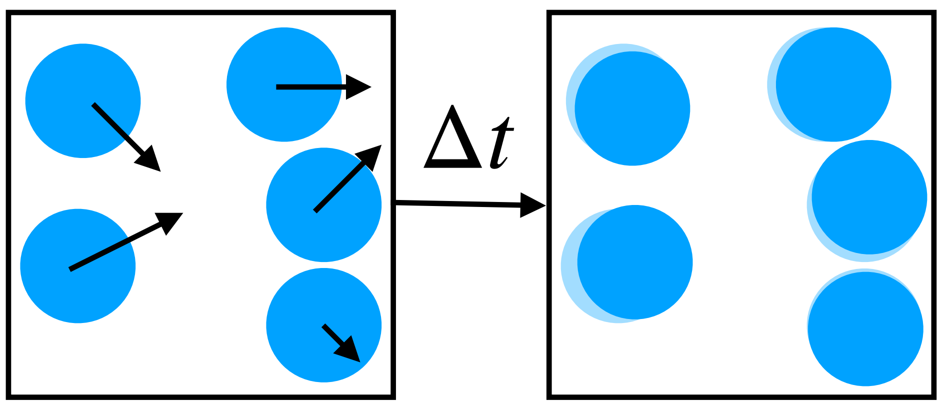
Sara Romeo Atance

Viguera Diez J, Romeo Atance S, Engkvist O, Mercado R, Olsson S*. ***ELLIS Machine Learning for Molecule Discovery Workshop*** *2021*

Challenge: $\mathbf{x} \sim p(\mathbf{x}) \propto \exp\left(-\beta U(\mathbf{x})\right)$

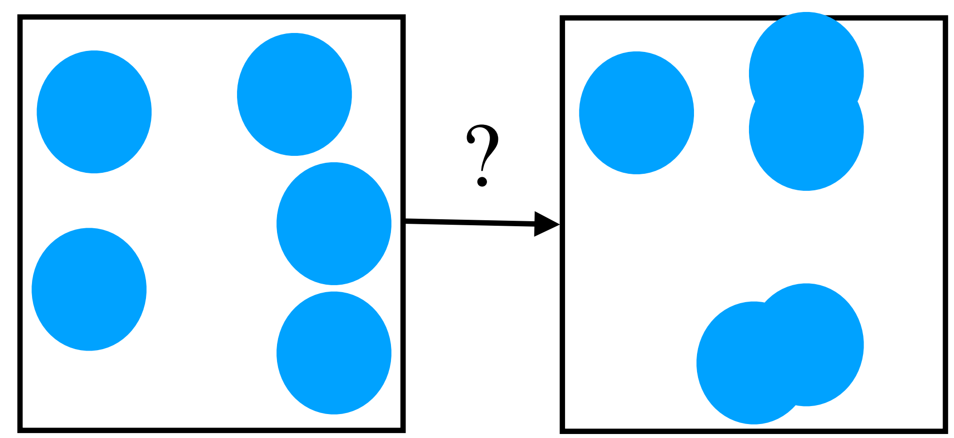
Challenge: $\mathbf{x} \sim p(\mathbf{x}) \propto \exp\left(-\beta U(\mathbf{x})\right)$

Molecular dynamics



Correlated samples = *Slow mixing*

Monte Carlo



Low-acceptance or
highly correlated samples

Deep Generative Models

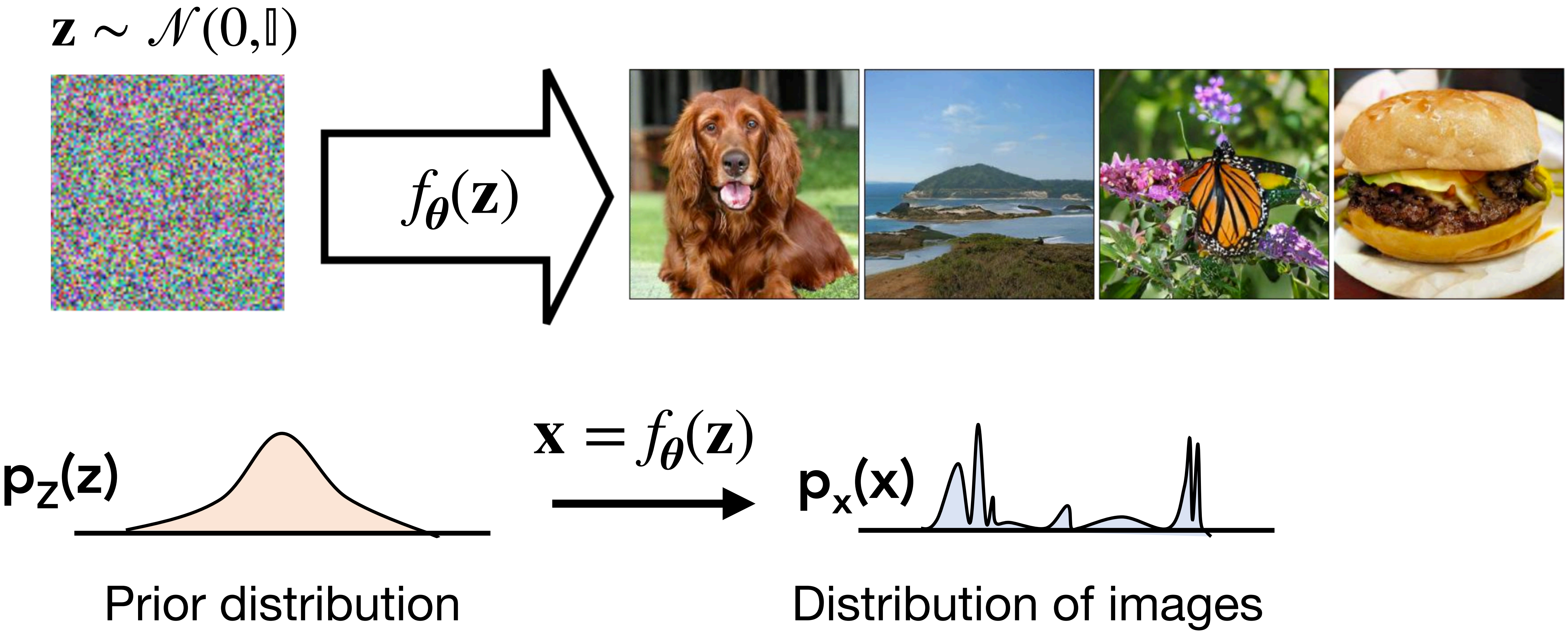
$\mathbf{z} \sim \mathcal{N}(0, \mathbb{I})$

$f_{\theta}(\mathbf{z})$

GAN (Goodfellow et al., 2014)

BigGAN (Brock et al., 2019)

Deep Generative Models



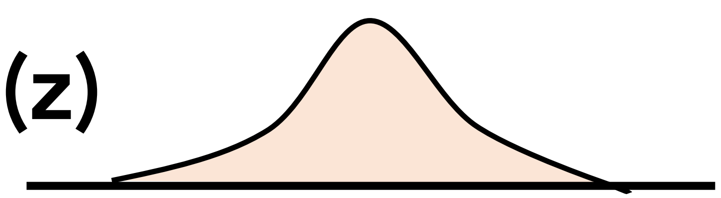
GAN (Goodfellow et al., 2014)

BigGAN (Brock et al., 2019)

Deep Generative Models

$\mathbf{z} \sim \mathcal{N}(0, \mathbb{I})$

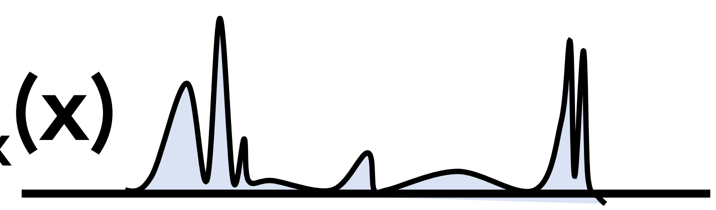
$p_Z(z)$



Prior distribution

$\mathbf{x} = f_\theta(\mathbf{z})$

$p_x(x)$

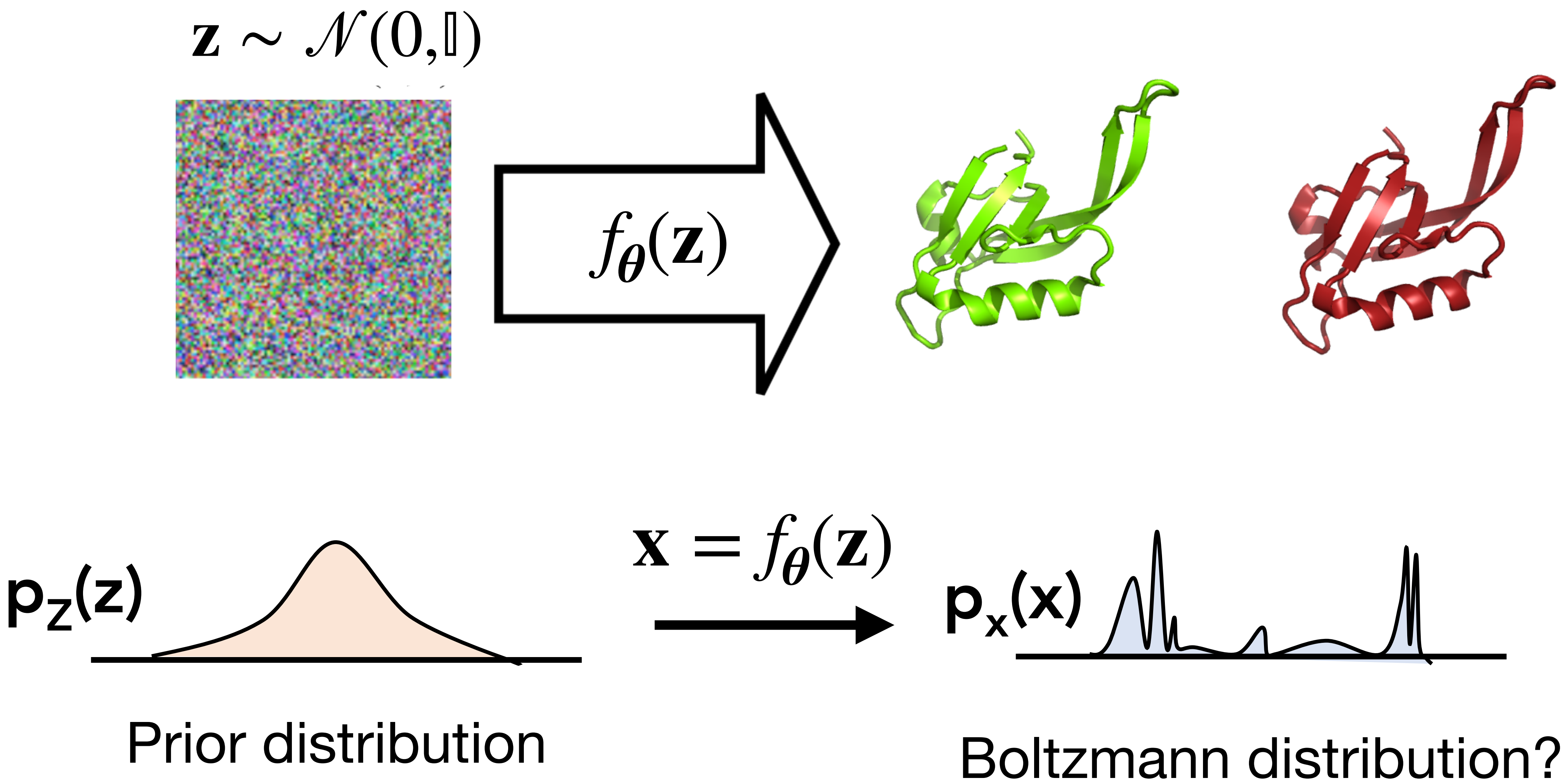


Distribution of images

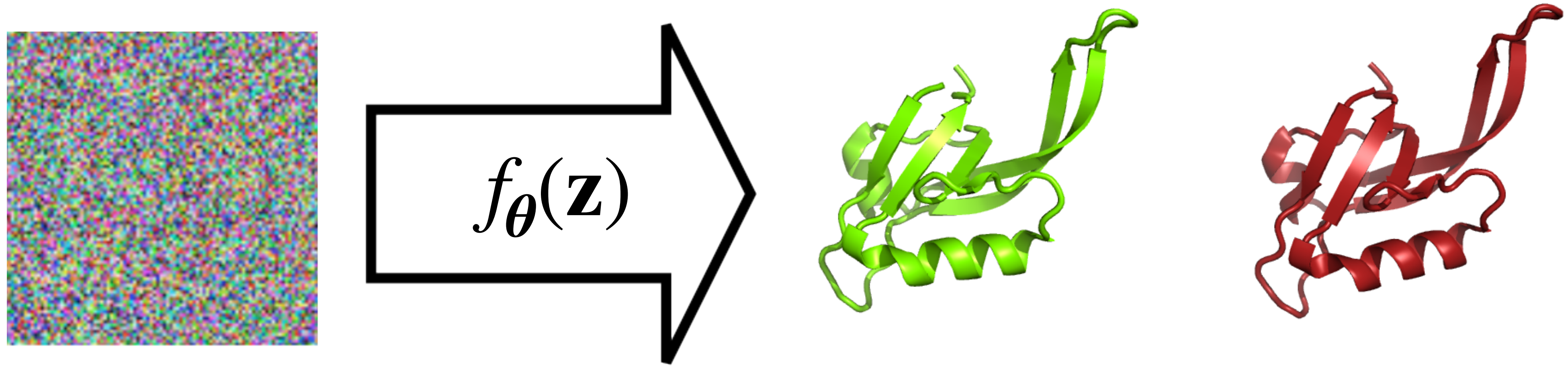
What if $p_{\mathbf{x}}(\mathbf{x})$ was the Boltzmann distribution?

GAN (Goodfellow et al., 2014)

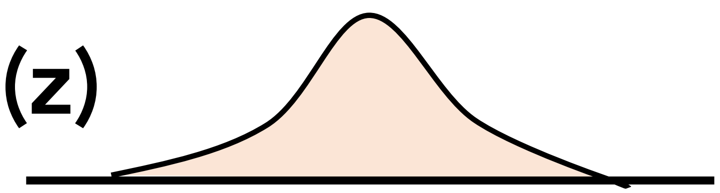
BigGAN (Brock et al., 2019)


$\mathbf{z} \sim \mathcal{N}(0,\mathbb{I})$
$f_\theta(\mathbf{z})$
$\mathbf{x} = f_\theta(\mathbf{z})$
$p_Z(z)$
$p_x(x)$
Prior distribution
Boltzmann distribution?

$\mathbf{z} \sim \mathcal{N}(0, \mathbb{I})$

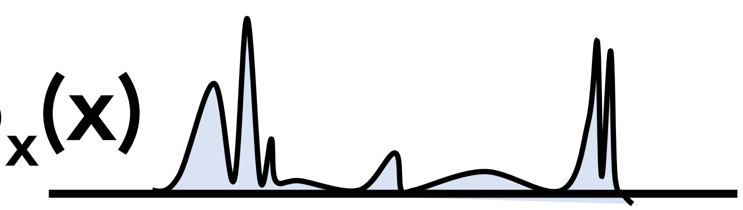


$p_Z(z)$



$\mathbf{x} = f_\theta(\mathbf{z})$

$p_x(x)$

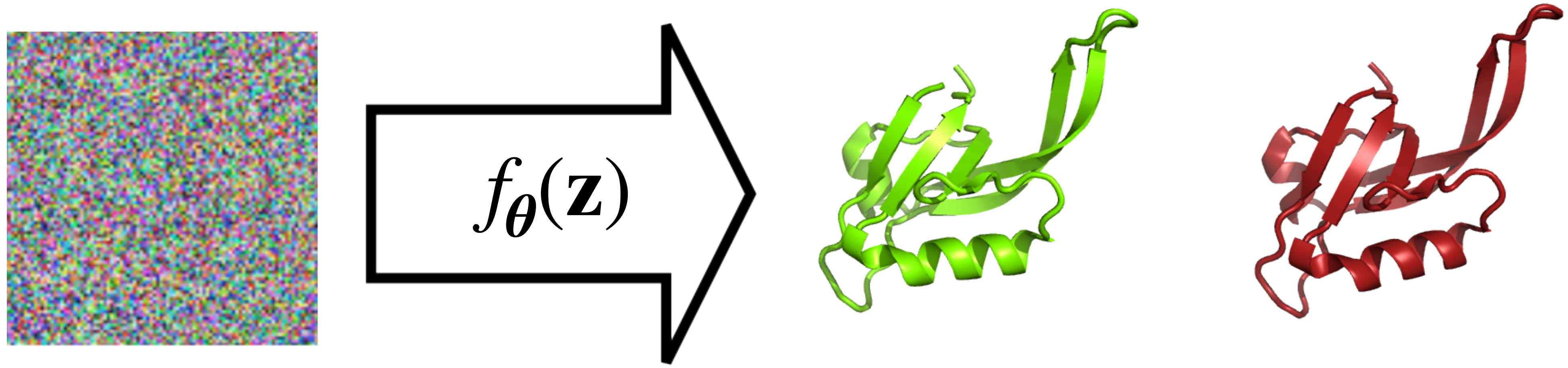


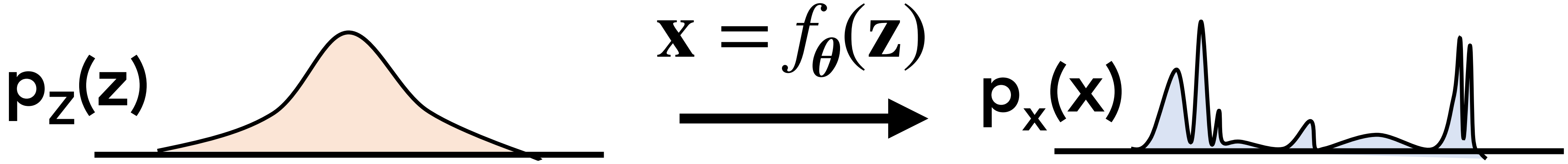
Prior distribution

Boltzmann distribution?

<u>Problem: We do not have a lot of data!</u>

$\mathbf{z} \sim \mathcal{N}(0, \mathbb{I})$





Prior distribution

Boltzmann distribution?

<u>Problem: We do not have a lot of data!</u>

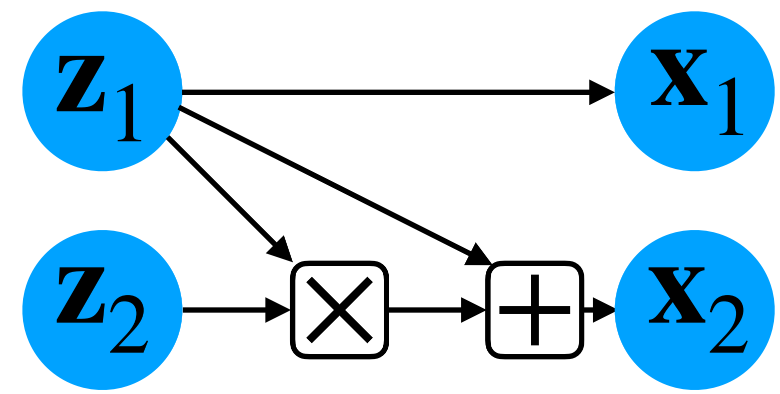
But we do have $U(\mathbf{x})$

If $f_\theta(\mathbf{z})$ satisfies + cheap to compute

$$\mathbf{x} = f_\theta(\mathbf{z}) \qquad J(\mathbf{z})$$

$$\mathbf{z} = f_\theta^{-1}(\mathbf{x}) \qquad J^{-1}(\mathbf{x})$$

RealNVP



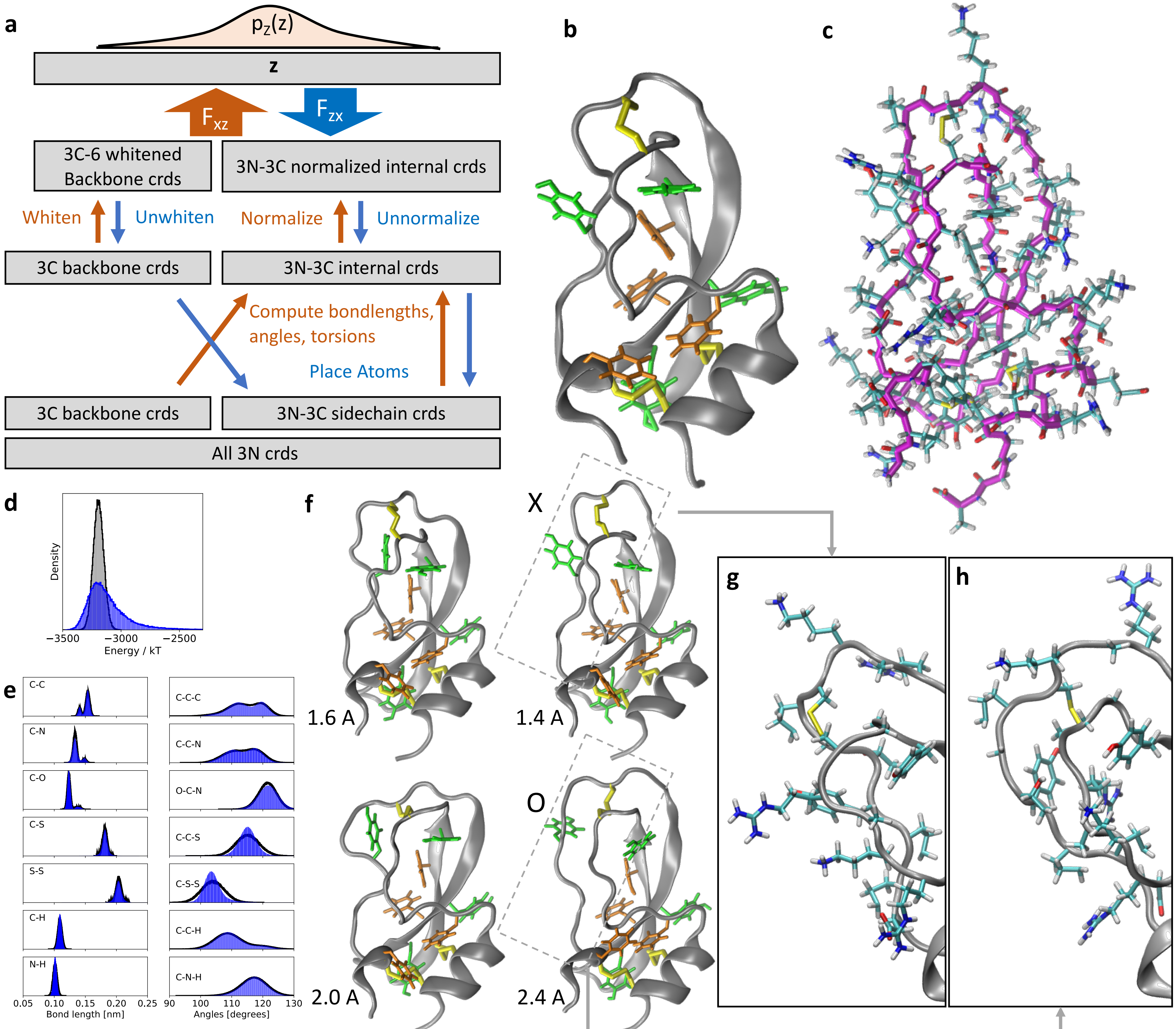
Dinh, Sohl-Dickstein & Bengio *arXiv:1605.08803* (2016).

We can optimize

$$J_{KL} = \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[U(f_\theta(\mathbf{z})) - \log \det J(\mathbf{z})\right]$$

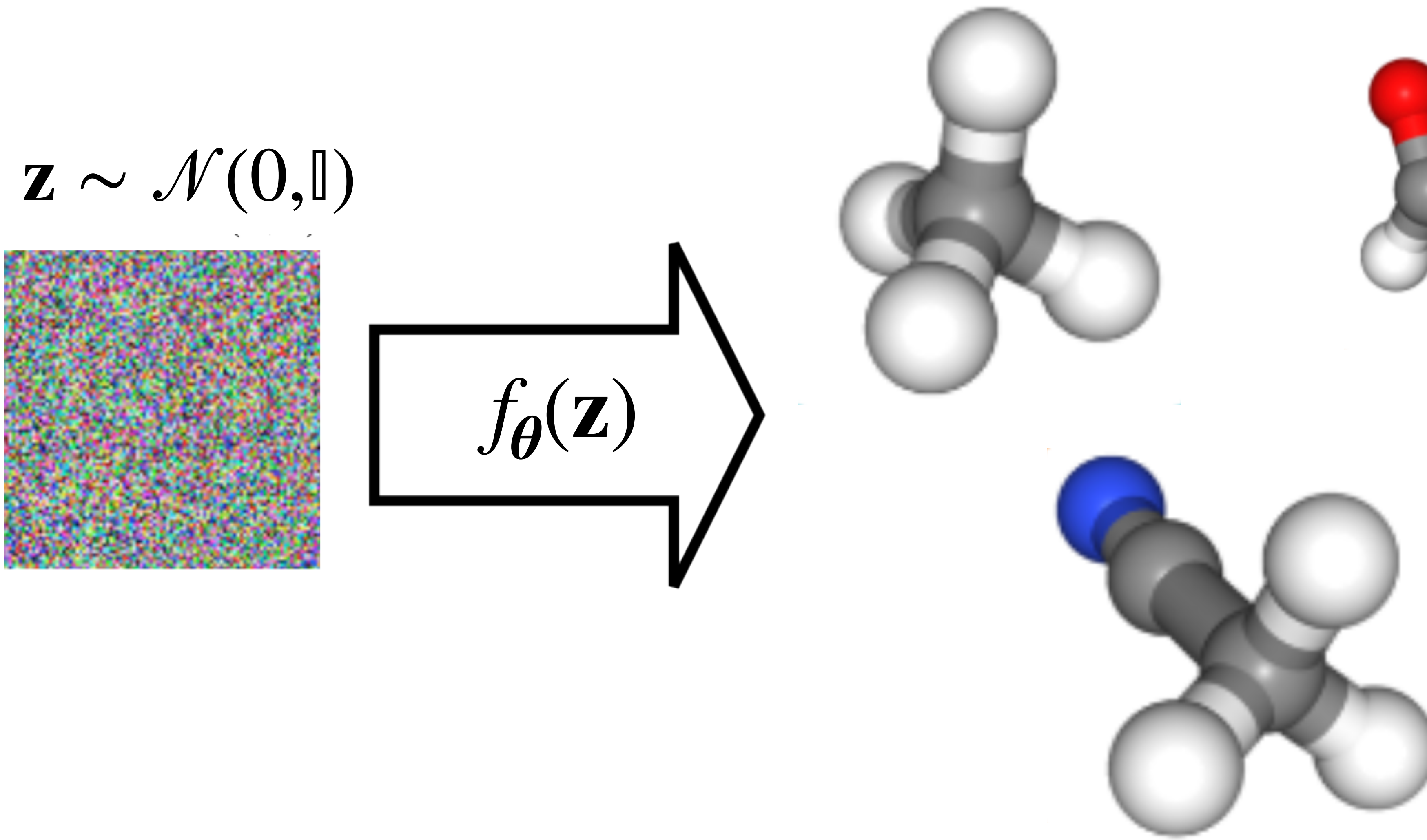
Potential energy
(‘enthalpy’)

Smearing of prior distribution
(‘entropy’)



a
$p_z(z)$
z
F_{xz}
F_{zx}
3C-6 whitened Backbone crds
3N-3C normalized internal crds
Whiten
Unwhiten
Normalize
Unnormalize
3C backbone crds
3N-3C internal crds
Compute bondlengths, angles, torsions
Place Atoms
3C backbone crds
3N-3C sidechain crds
All 3N crds
b
c
d
Density
−3500
−3000
−2500
Energy / kT
e
C-C
C-N
C-O
C-S
S-S
C-H
N-H
0.05
0.10
0.15
0.20
0.25
Bond length [nm]
C-C-C
C-C-N
O-C-N
C-C-S
C-S-S
C-C-H
C-N-H
90
100
110
120
130
Angles [degrees]
f
X
O
1.6 A
1.4 A
2.0 A
2.4 A
g
h

How can we build a Boltzmann Generator which works for multiple molecules?



How can we build a Boltzmann Generator which works for multiple molecules?

Use an auto-regressive structure,

$$p_{\boldsymbol{\theta},M}(\mathbf{x}) = p(\mathbf{x}_1)\prod_{i=2}^{N} p(\mathbf{x}_i \mid \mathbf{x}_{<i})$$

How can we build a Boltzmann Generator which works for multiple molecules?

Use an auto-regressive structure,

$$p_{\boldsymbol{\theta},M}(\mathbf{x}) = p(\mathbf{x}_1)\prod_{i=2}^{N} p(\mathbf{x}_i \mid \mathbf{x}_{<i})$$

Transferable kernel,

$$p_{\boldsymbol{\theta},M}(\mathbf{x}) = \prod_{i=2}^{N} p_{\boldsymbol{\theta}}(\mathbf{x}_i \mid \boldsymbol{C}_i)$$

With $\mathbf{C}_i = \left(\mathbf{M}, \mathbf{A}_i, \mathsf{R}_{<i}, \mathbf{F}_i\right)$

How can we build a Boltzmann Generator which works for multiple molecules?

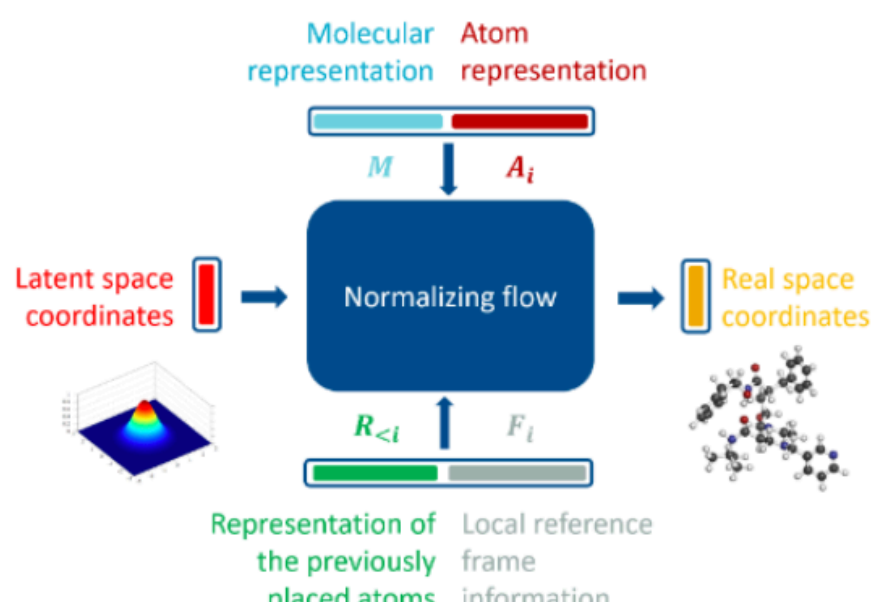
Use an auto-regressive structure,

$$p_{\boldsymbol{\theta},M}(\mathbf{x}) = p(\mathbf{x}_1)\prod_{i=2}^{N} p(\mathbf{x}_i \mid \mathbf{x}_{<i})$$

Transferable kernel,

$$p_{\boldsymbol{\theta},M}(\mathbf{x}) = \prod_{i=2}^{N} p_{\boldsymbol{\theta}}(\mathbf{x}_i \mid \boldsymbol{C}_i)$$

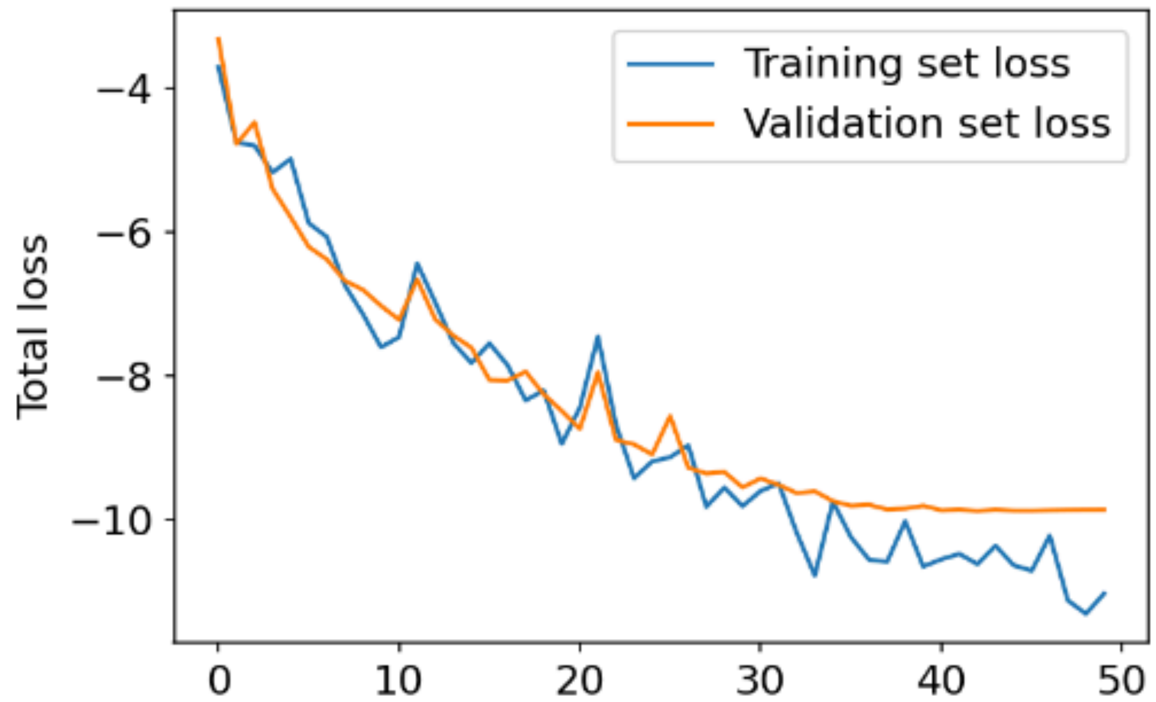
With $\mathbf{C}_i = \left(\mathbf{M}, \mathbf{A}_i, \mathbf{R}_{<i}, \mathbf{F}_i\right)$

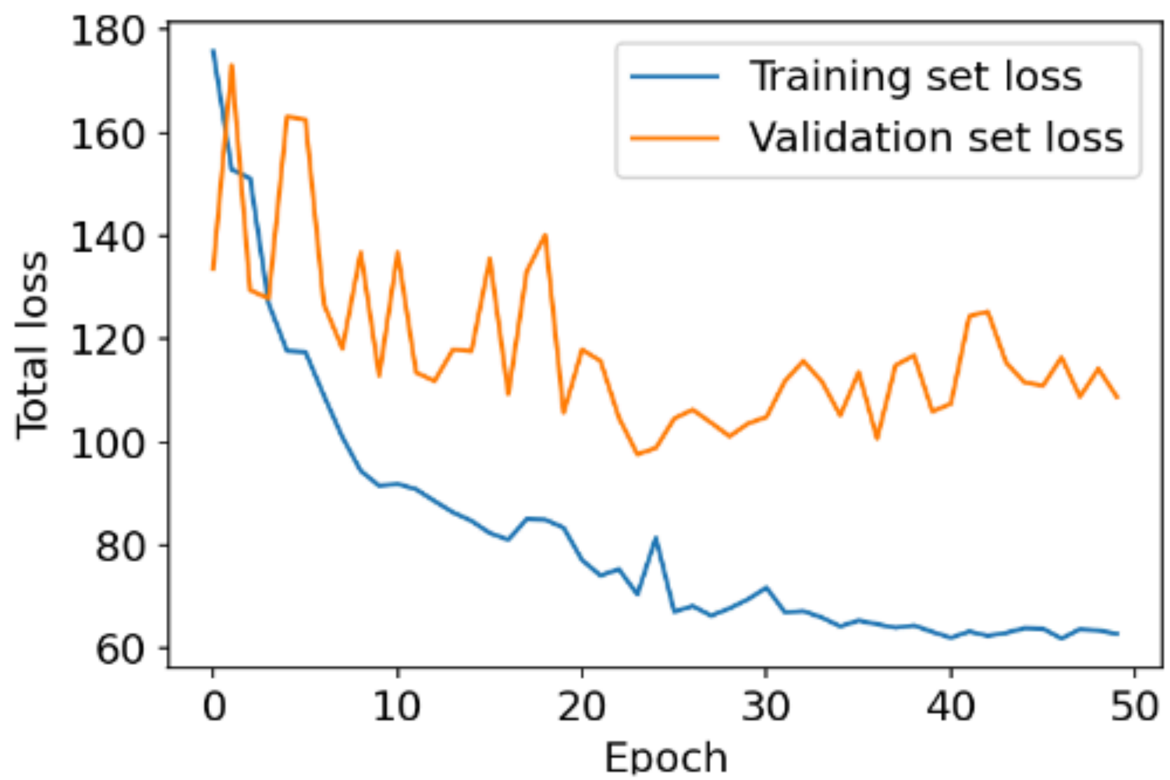


Viguera Diez J, Romeo Atance S, Engkvist O, Mercado R, Olsson S*.
***ELLIS Machine Learning for Molecule Discovery Workshop** 2021*

Schütt, K. T., Kindermans, P.-J., Sauceda, H. E., Chmiela, S.,Tkatchenko, A., and Müller, K.-R. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions, 2017.

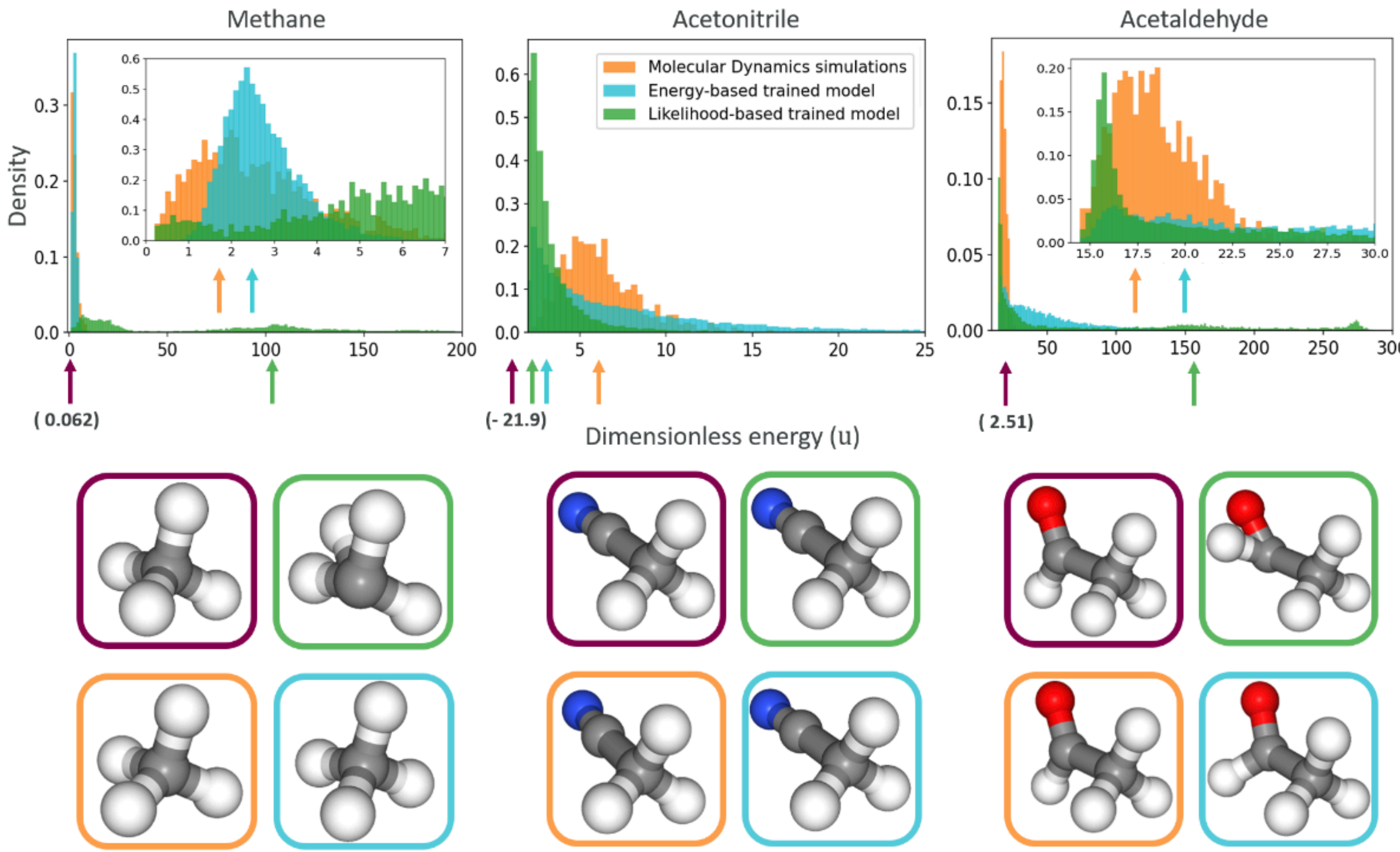
Transferability





Viguera Diez J, Romeo Atance S, Engkvist O, Mercado R, Olsson S*.
ELLIS Machine Learning for Molecule Discovery Workshop *2021*

Transferability



Viguera Diez J, Romeo Atance S, Engkvist O, Mercado R, Olsson S*.
ELLIS Machine Learning for Molecule Discovery Workshop *2021*

Acknowledgements

Juan Viguera Diez

Tobias Karlsson

Christopher Kolloff

Vladimir Pastukhov

Azadeh Karimisefat & Dipti Aswal
Enmin Su & Wenli Gao

WASP | WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM