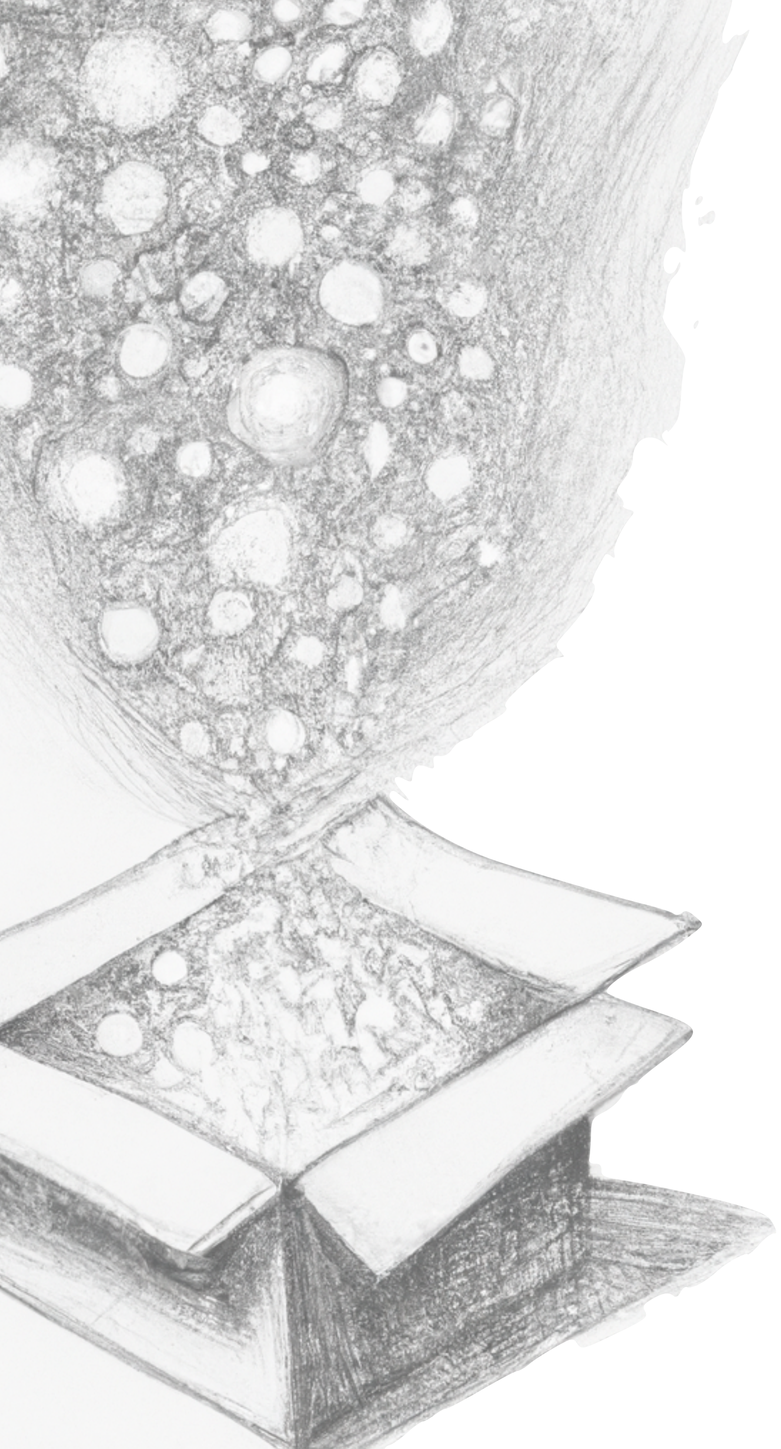


[Preliminary draft version]

An introduction to generative modeling

with applications in physics

Siddharth Mishra-Sharma

(smsharma.io/@kdqg1)

NSF AI Institute for Artificial Intelligence and Fundamental Interactions (IAIFI)

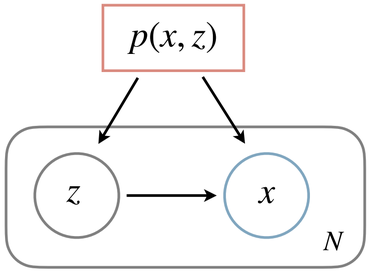
IAIFI Summer School

August 8, 2023

Outline

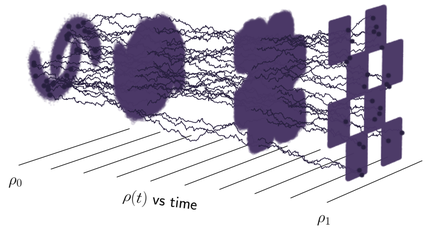
Why (deep) generative modeling?

What is it, and what can it do for you?



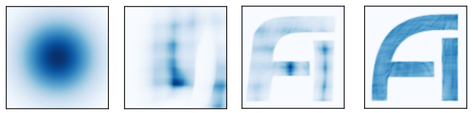
Variational auto encoders

Latent-variable modeling, and compression is all you need



Diffusion models

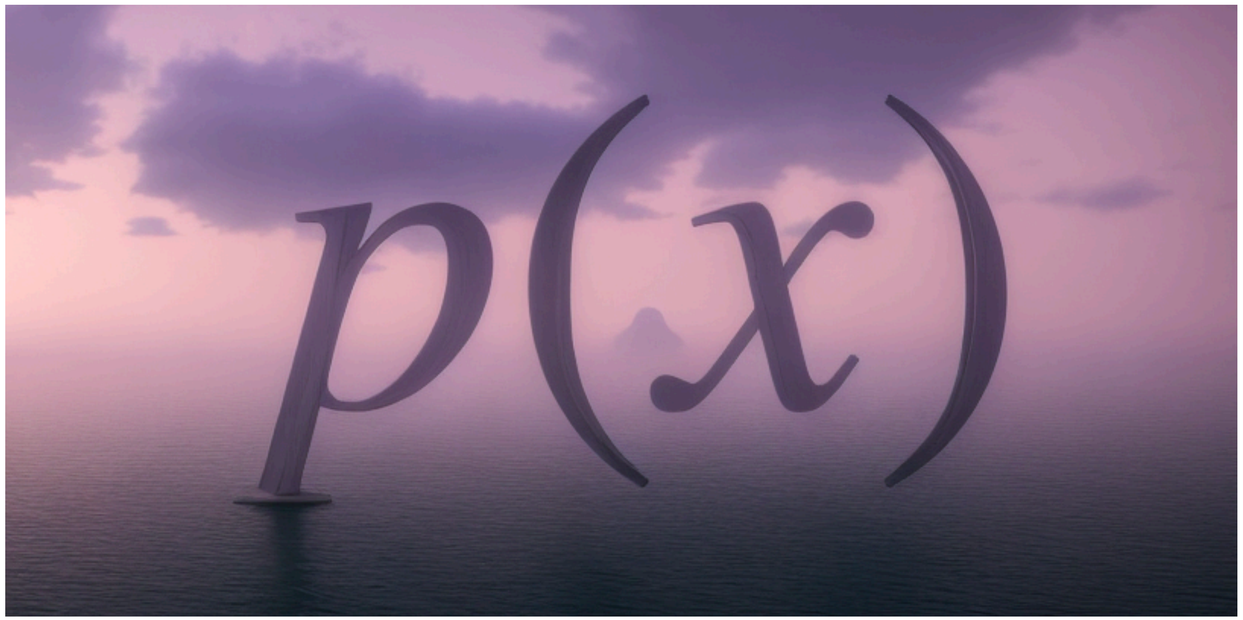
Models based on iterative refinement



Normalizing flows (and some other models)

Invertible transformations

The *data distribution*



We often work with data distributions in physics

(modeled or empirical)

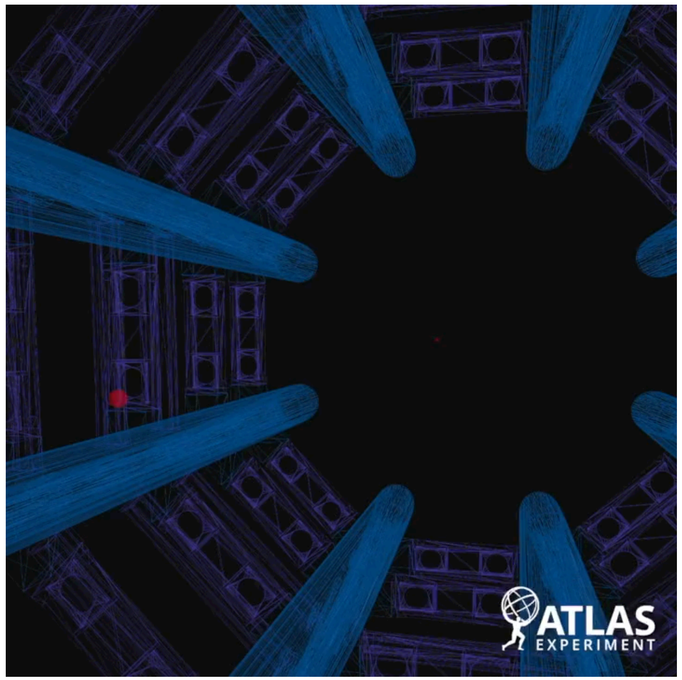
Simulators

$$x \sim p(x)$$

Simulators are ubiquitous: *they prescribe a way to sample from the data distribution*

Collider data

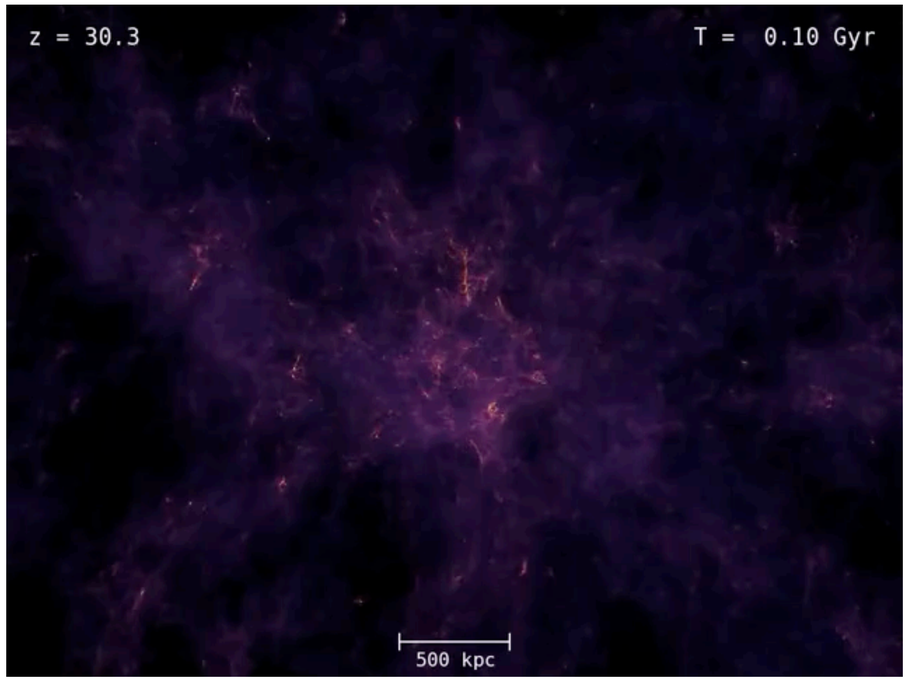
particles $\sim p$(particles)



[C. Cesarotti with ATLAS]

Cosmology data

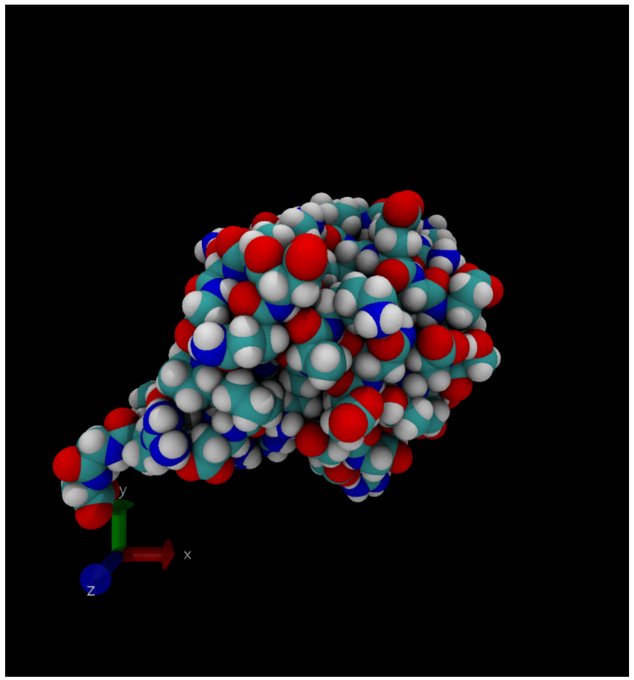
particles $\sim p$(particles)



[Aquarius simulation]

Molecular dynamics

configurations $\sim p$(configurations)



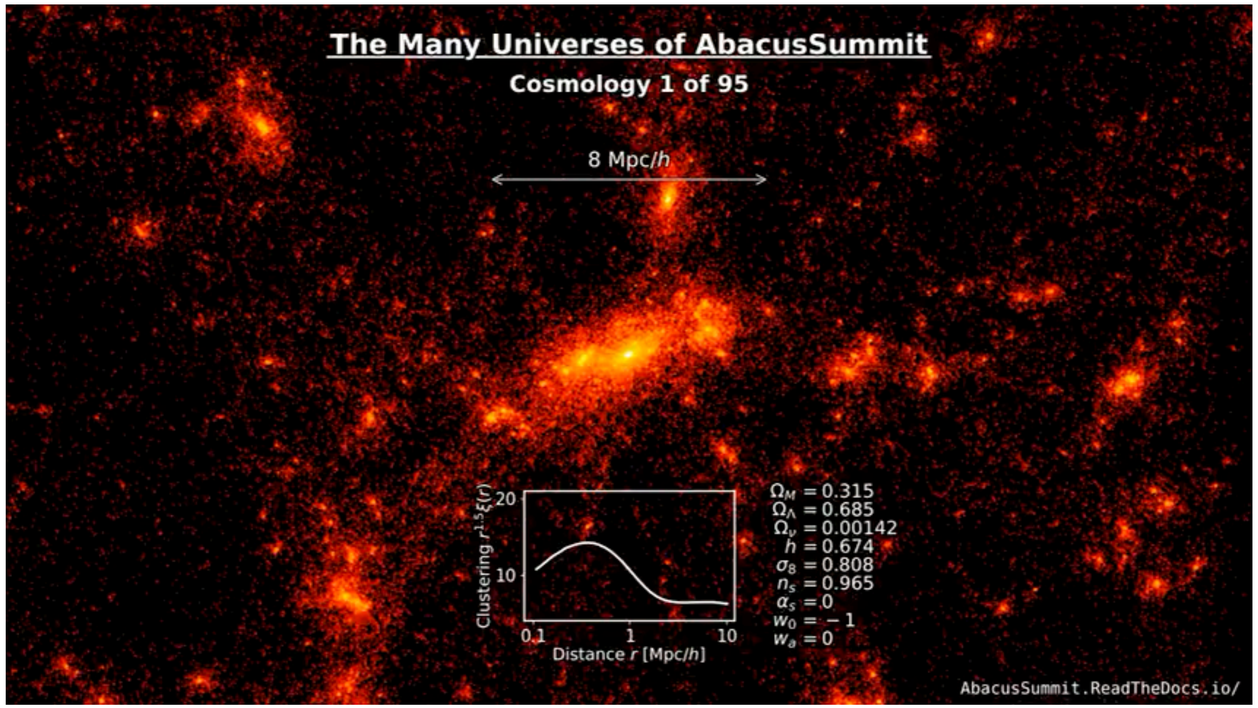
[E. Cances et al]

Conditional simulators

Conditional simulations *sample from the likelihood* $p(x \mid \theta)$

Cosmology data

$$\text{particles} \sim p(\text{particles} \mid \{\Omega_m, \sigma_8\})$$



[Abacus Summit]

$$x \sim p(x; \mathscr{M})$$

Model

or

$$x \sim p(x \mid \theta)$$

Model parameters

Are simulators all you need?

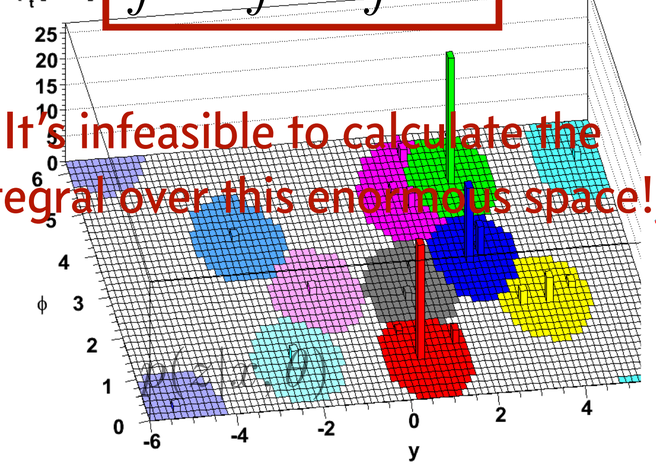
Latent variables

Observables | Detector interactions | Shower splittings | Parton-level momenta | Theory parameters

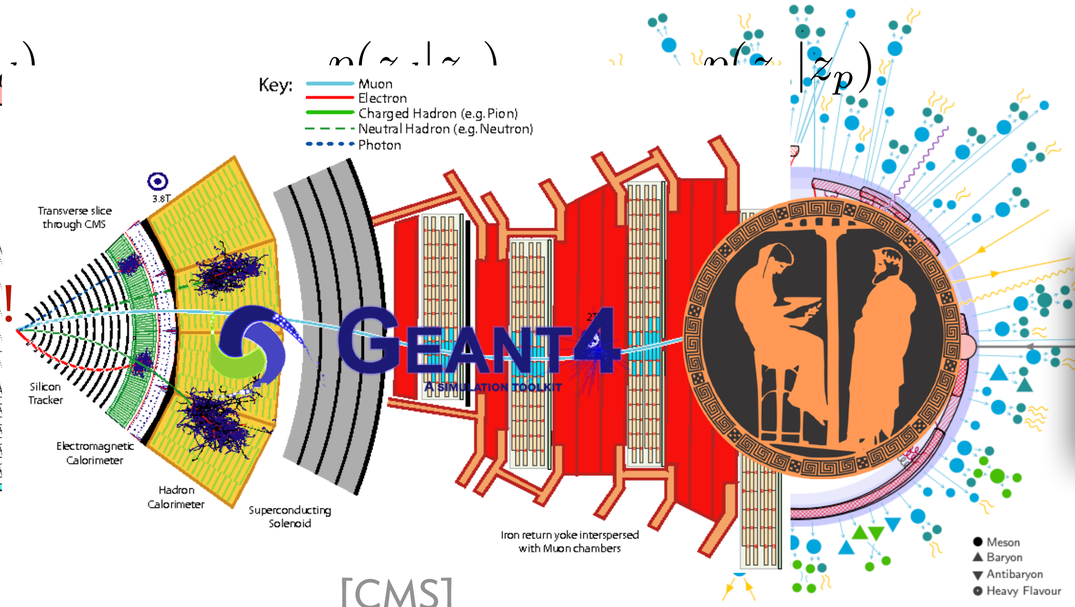
$x \leftarrow z_d \leftarrow z_s \leftarrow z_p \leftarrow \theta$

$$p(x|\theta) = \int \mathrm{d}z_d \int \mathrm{d}z_s \int \mathrm{d}z_p \; p(x|z_d) \; p(z_d|z_s) \; p(z_s|z_p) \; p(z_p|\theta)$$

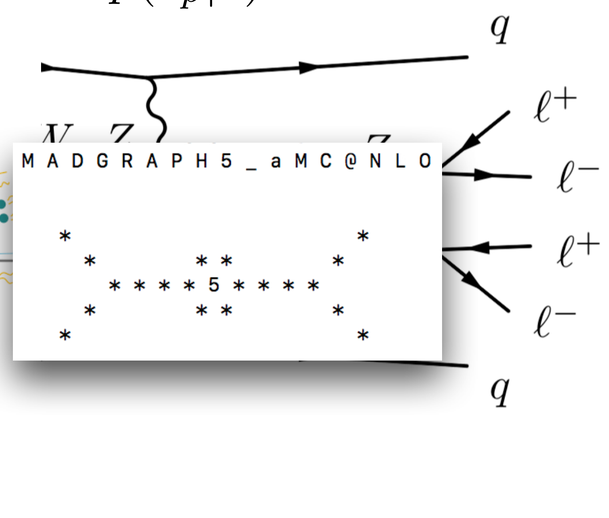
It's infeasible to calculate the integral over this enormous space!



[M. Cacciari, G. Salam, G. Soyez 0802.1189]



[CMS]



[Pythia]

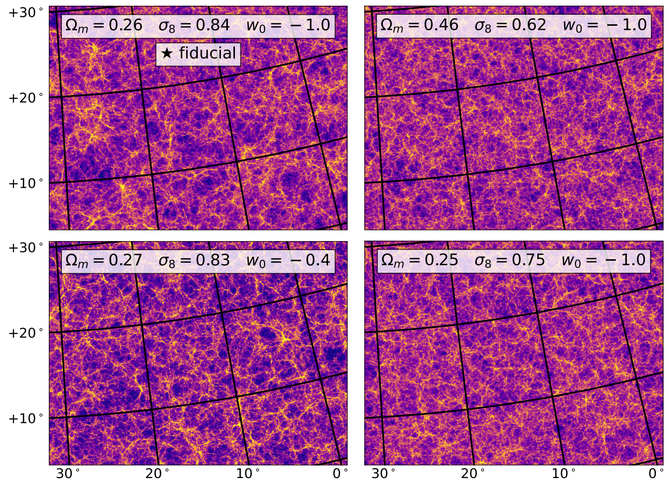
Prediction (simulation) — Inference

What could we do with $p(x)$?

Produce samples

for downstream applications: a fast simulator/emulator

$$x \sim p(x)$$

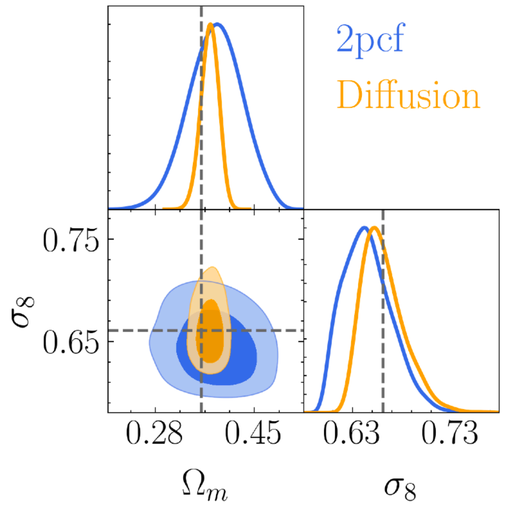


[Kacprzak et al 2022]

Evaluate likelihood

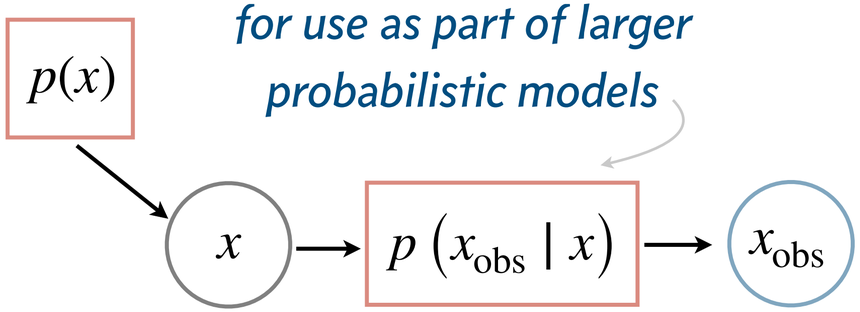
for model selection, parameter inference, outlier detection, …

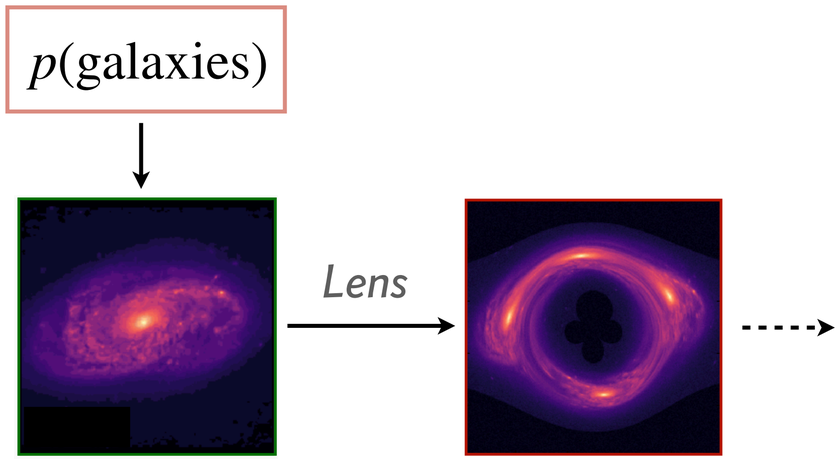
$$p(\theta \mid x) = \frac{p(x \mid \theta) \cdot p(\theta)}{p(x)}$$



Encode complex priors

for use as part of larger probabilistic models

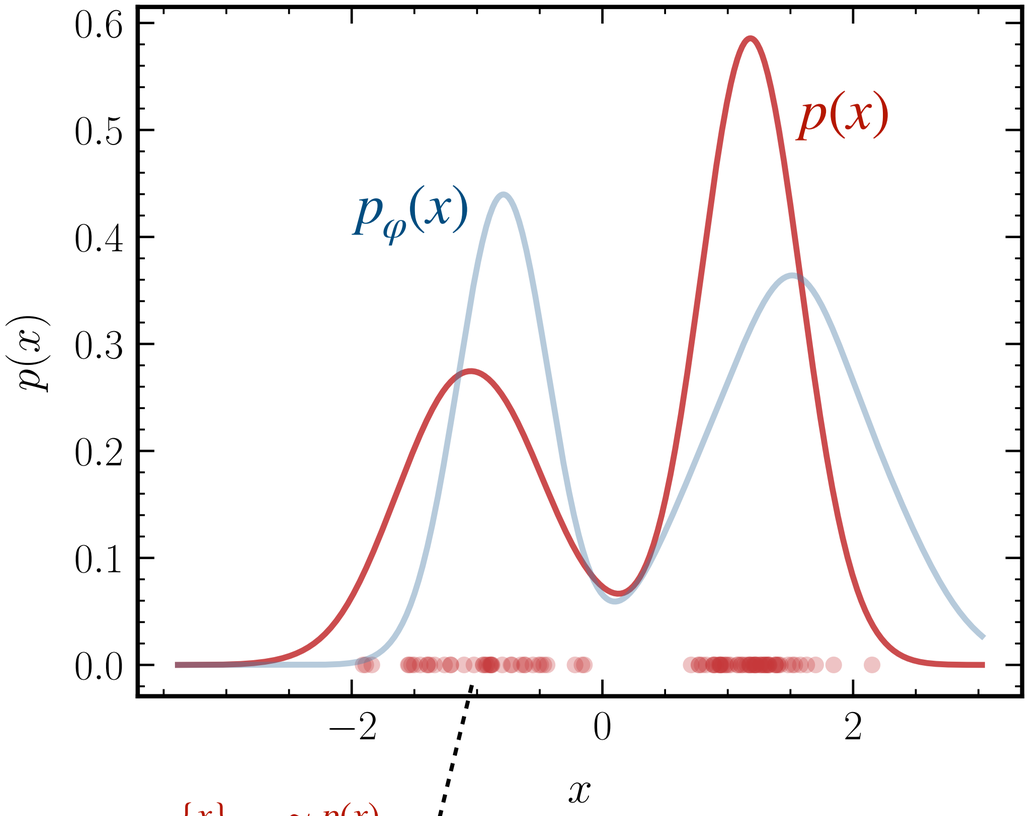




Generative modeling can efficiently enable these for a wide variety of scientific data/models!

Learning the data distribution

I'm sold! *How do I learn a generative model for my data?*



1. Ingredients:

- A parameterized distribution $p_\varphi(x)$
- Samples from the data distribution $\{x\}_{\text{train}} \sim p(x)$

(empirical or simulated)

2. Maximize the likelihood of the model under the training data samples

$$\hat{\phi} = \arg\max_{\varphi} \left[\log p_\varphi \left(\{x\}_{\text{train}} \right) \right]$$

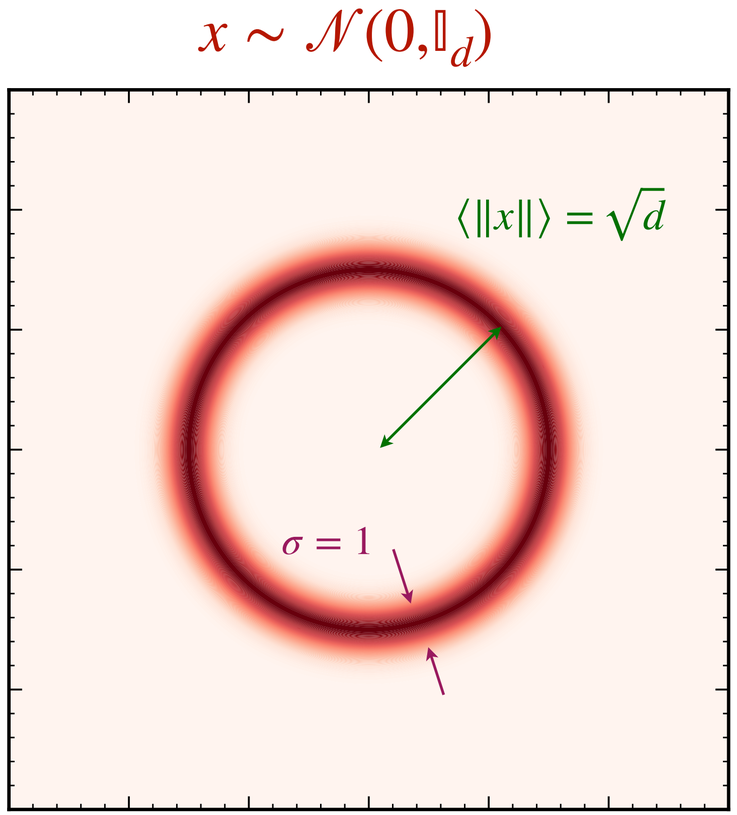
3. $p_{\hat{\varphi}}(x)$ 👏🎉🏝️

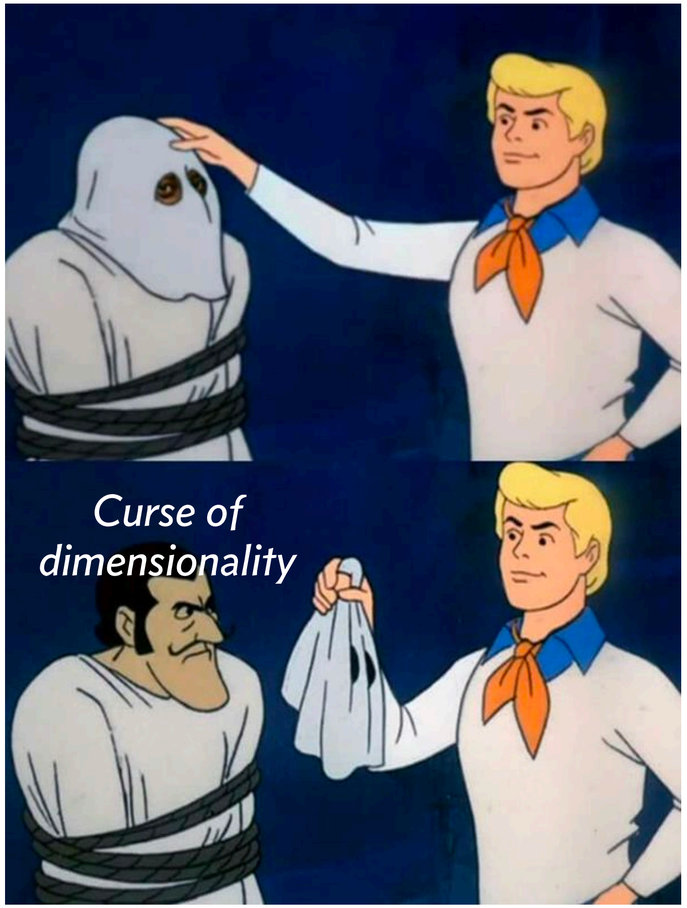
Not so fast…

The curse of dimensionality

Where is most of the probability mass concentrated in high dimensions?

- In high dimensions, most of the probability density of a Gaussian distribution lies in a thin shell at distance $\sqrt{d}$ from the center
- Vanishingly small fraction of distribution support is actually occupied.

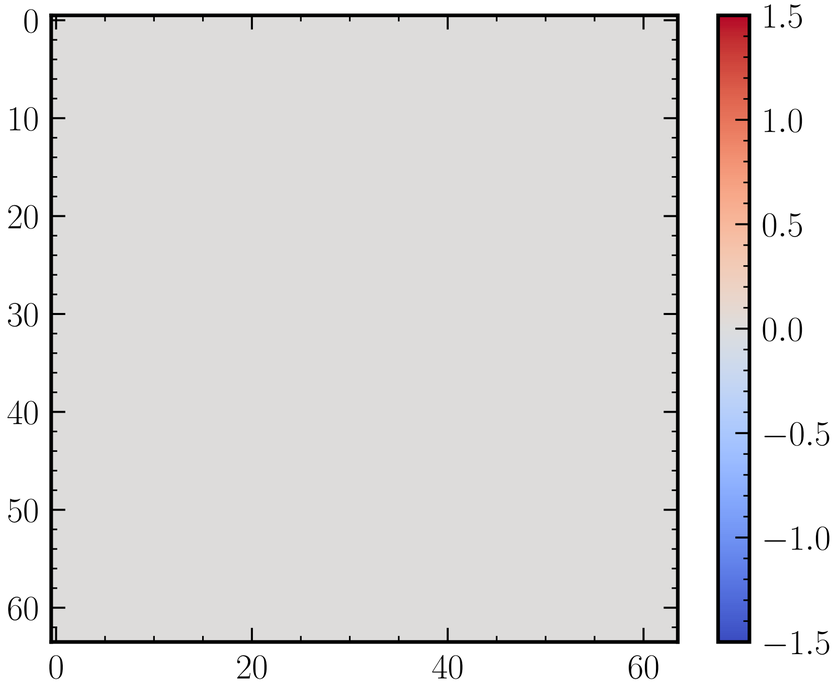




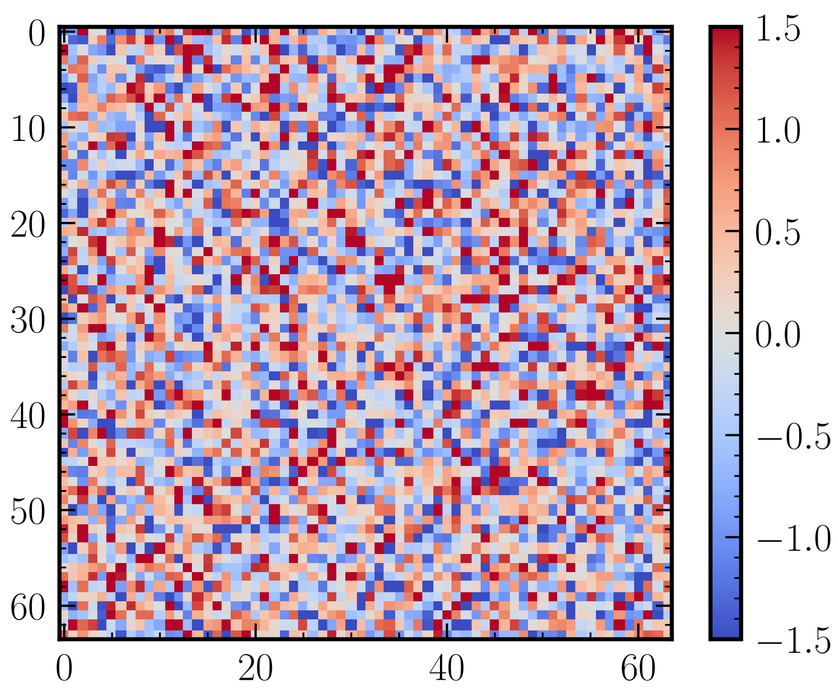
Learning high-dimensional distributions is challenging!

Typicality and *likelihood* of samples

Which of these samples have a higher likelihood under $\mathscr{L} = \mathcal{N}(0, \mathbb{I}_d)$?



$\log \mathscr{L} \approx -0.92$ nats/dim

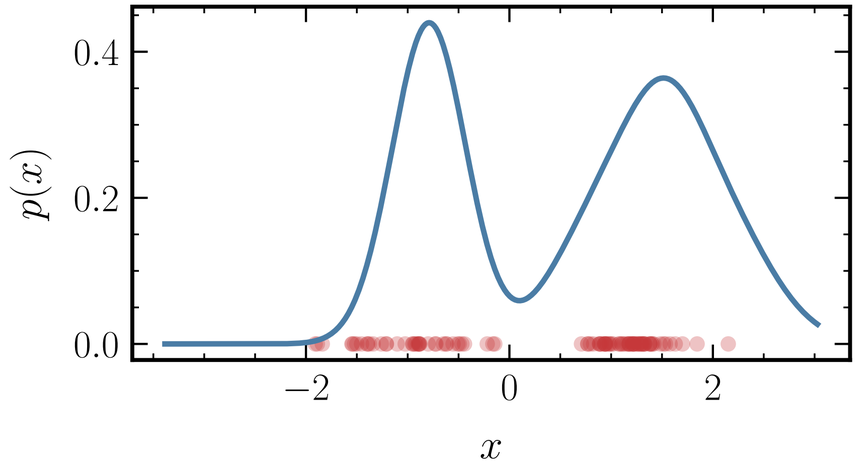


$\log \mathscr{L} \approx -1.43$ nats/dim

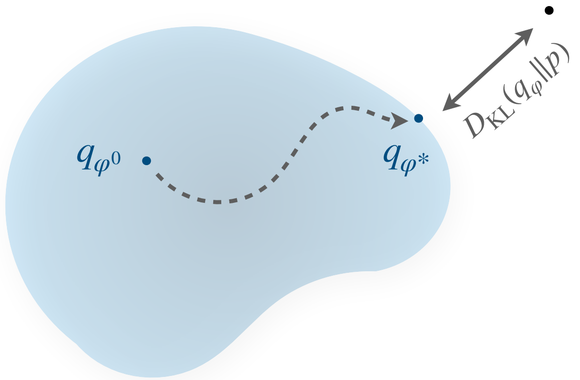
Evaluation of high-dimensional distributions is challenging!

(Some) Ways of training deep generative models

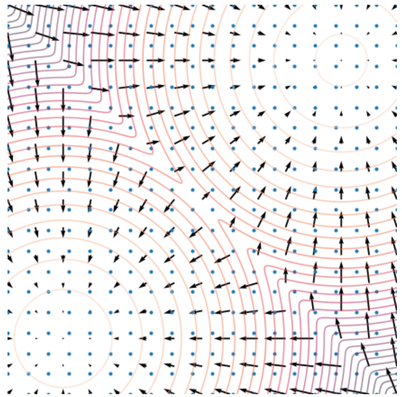
Maximum-likelihood



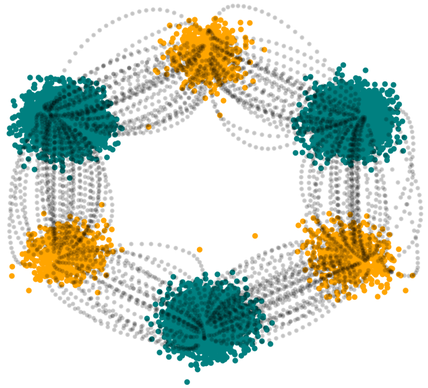
Optimizing a bound on the likelihood



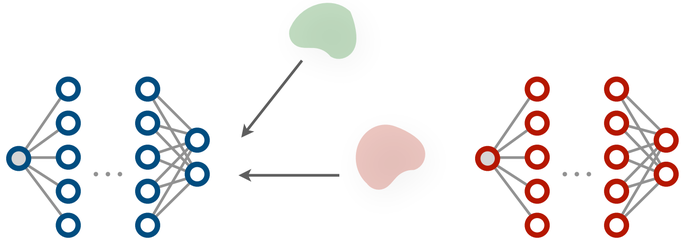
Score-matching



Optimal transport



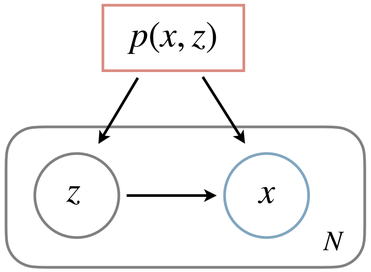
Adversarial training



Outline

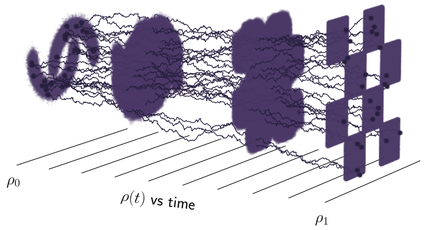
Why (deep) generative modeling?

What is it, and what can it do for you?



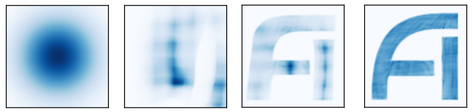
Variational auto encoders

Latent-variable modeling, and compression is all you need



Diffusion models

Models based on iterative refinement



Normalizing flows (and some other models)

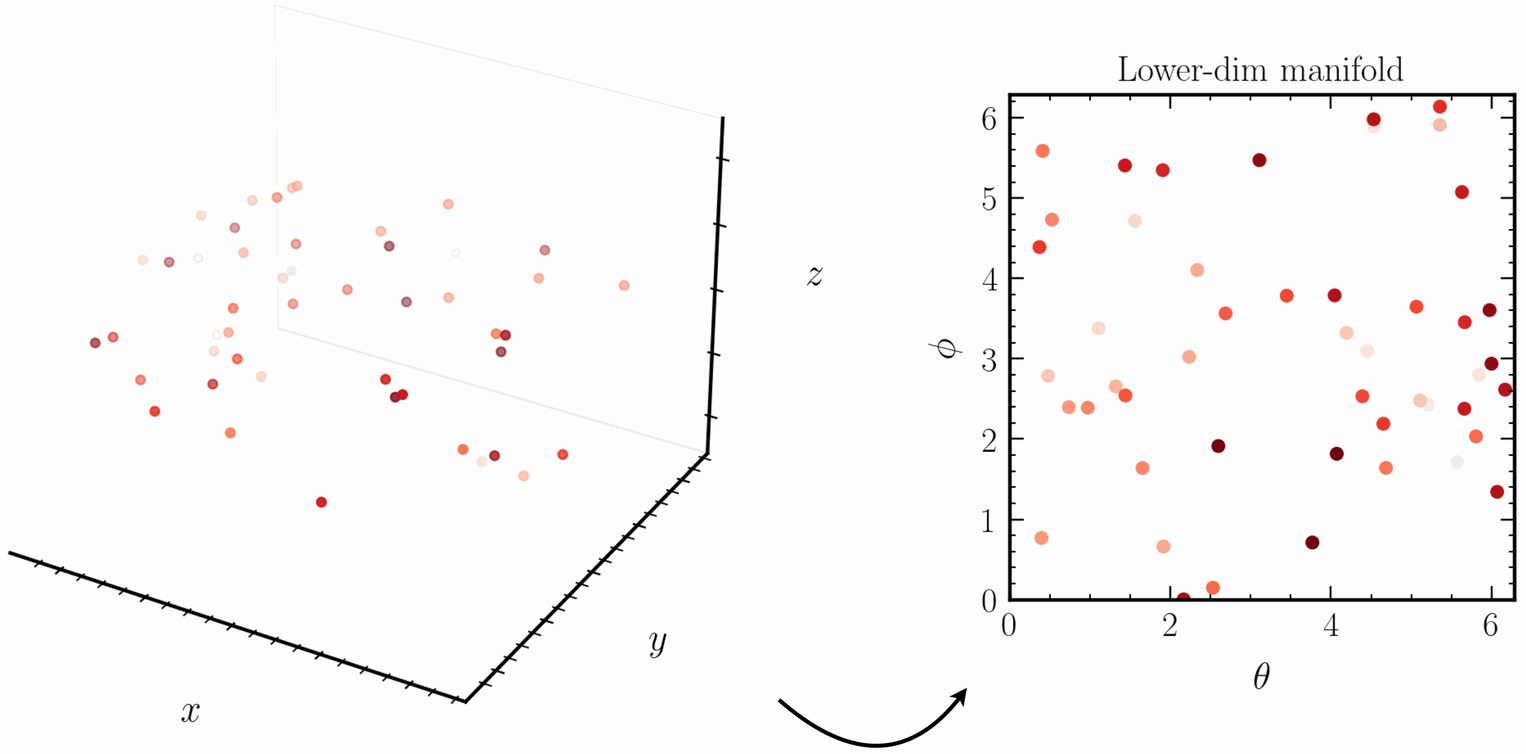
Invertible transformations

The pursuit of low-dimensional structure

Real-world datasets often live in structured low-dimensional manifolds

"Difficult to model" x

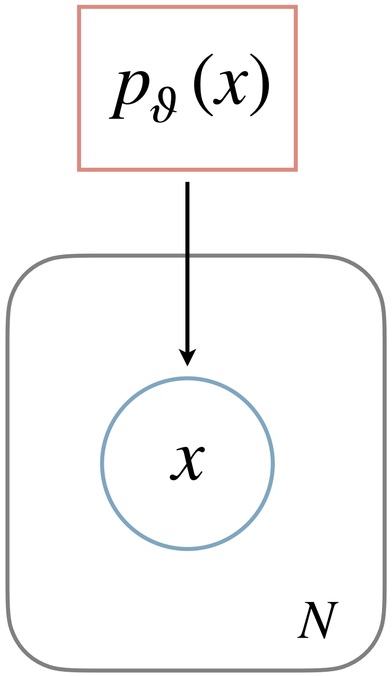
"Easy to model" z



$$p(z \mid x)$$

Latent-variable modeling

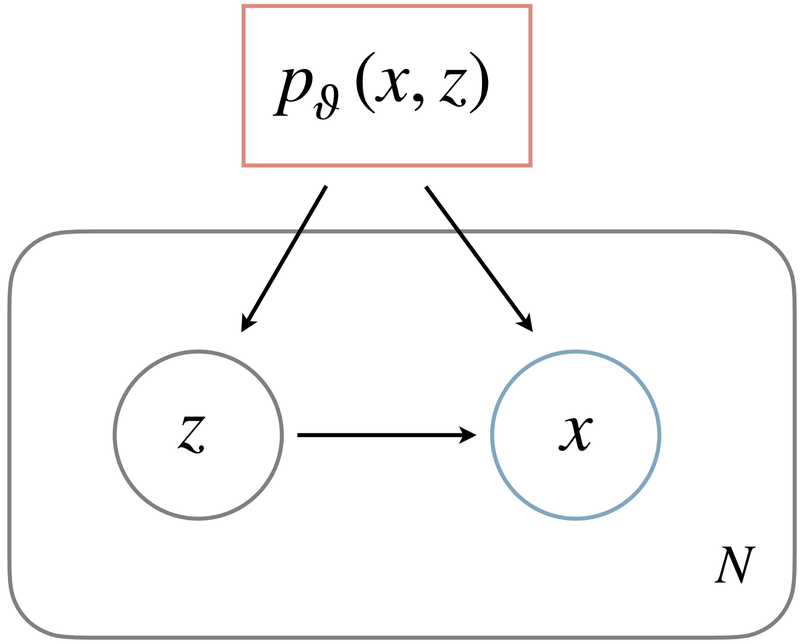
Learn lower-dimensional structure in the data distribution



Observed variables

Make the problem easier by making it "harder":

introduce *joint distribution* $p_\theta(x, z)$



Latent variables *Observed variables*

Common factorization:

$$p_\vartheta(x, z) = p(z) \cdot p_\vartheta\left(x \mid z\right)$$

Latent-variable modeling

Maximum-likelihood training?

$$\begin{aligned}\vartheta^* &= \arg\max_{\vartheta} p_\vartheta(x) \\ &= \arg\max_{\vartheta} \int p_\vartheta(x \mid z) p(z)\, \mathrm{d}z \\ &= \arg\max_{\vartheta} \left\langle p_\vartheta(x \mid z) \right\rangle_{p(z)}\end{aligned}$$

Difficult to build a good estimator!

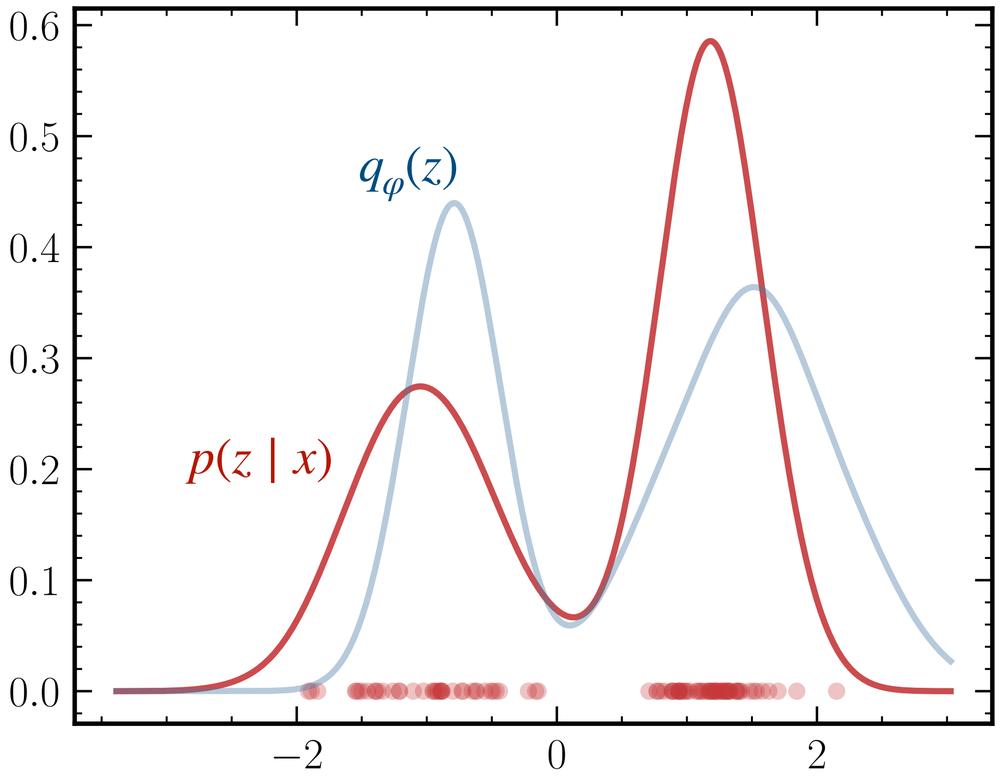
KL-divergence

A measure of similarity between two probability distributions

$$D_{\mathrm{KL}}(Q\|P) = \int_{-\infty}^{\infty} \mathrm{d}x\, q(x)\, \log\left(\frac{q(x)}{p(x)}\right)$$

$$= \left\langle \log \frac{q(x)}{p(x)} \right\rangle_{x\sim q(x)}$$

Formally: *expected excess "surprise" from using P as a model when the actual distribution is Q*



KL-divergence

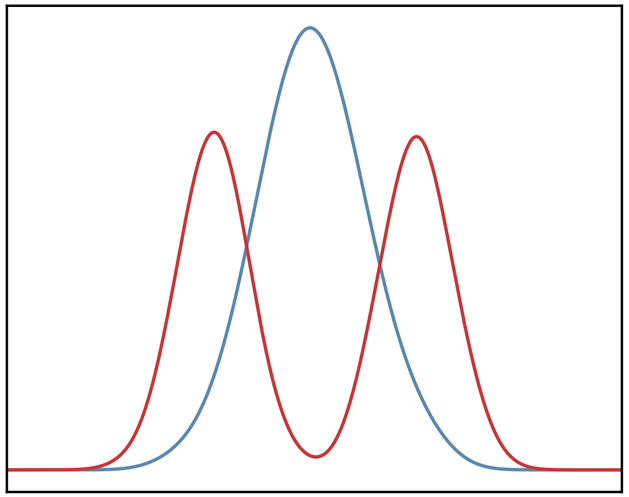
$$D_{\mathrm{KL}}(Q\|P) = \int_{-\infty}^{\infty} \mathrm{d}x\, q(x)\, \log\left(\frac{q(x)}{p(x)}\right)$$

A measure of similarity between two probability distributions

Not symmetric! $D_{\mathrm{KL}}(Q\|P) \neq D_{\mathrm{KL}}(P\|Q)$

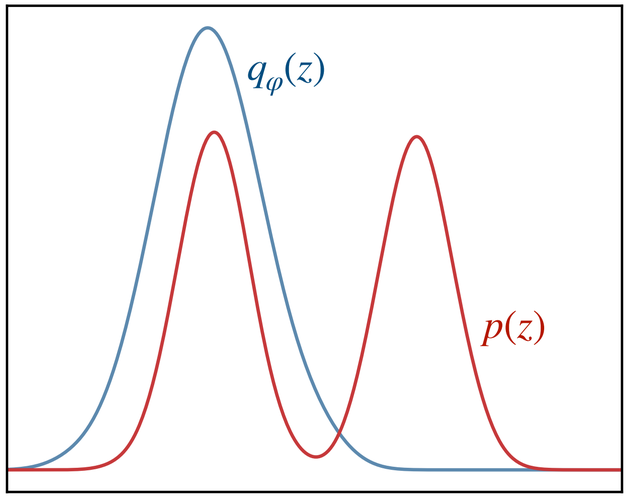
"True" distribution

"Forward" KL $D_{\mathrm{KL}}(P\|Q)$



Mean seeking

"Reverse" KL $D_{\mathrm{KL}}(Q\|P)$



Mode seeking

Forward KL

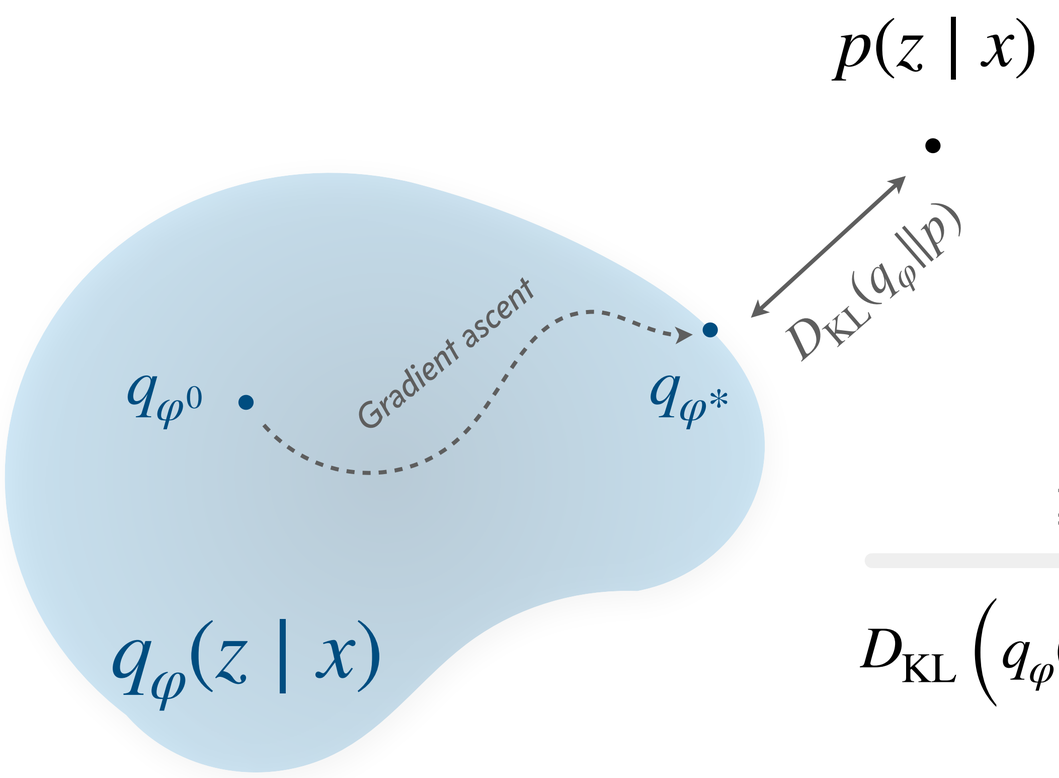
$$D_{\mathrm{KL}}(P_{\mathscr{D}}\|Q_\varphi) = -\left\langle \log q_\varphi(z) \right\rangle_{z\sim p_{\mathscr{D}}(z)} + \text{const}.$$

Maximum-likelihood inference is equivalent to minimizing the *forward* KL

Variational inference

The intractability of $p(x)$ is closely related to the intractability of the *posterior* $p(z \mid x)$

$$p(z \mid x) = \frac{p(x, z)}{p(x)} \quad \text{(Bayes' theorem)}$$



A two-for-one!

- Estimate approximate posterior $q_\phi(z) \approx p(z \mid x)$
- Estimate likelihood/evidence $\mathrm{ELBO} \approx p(x)$

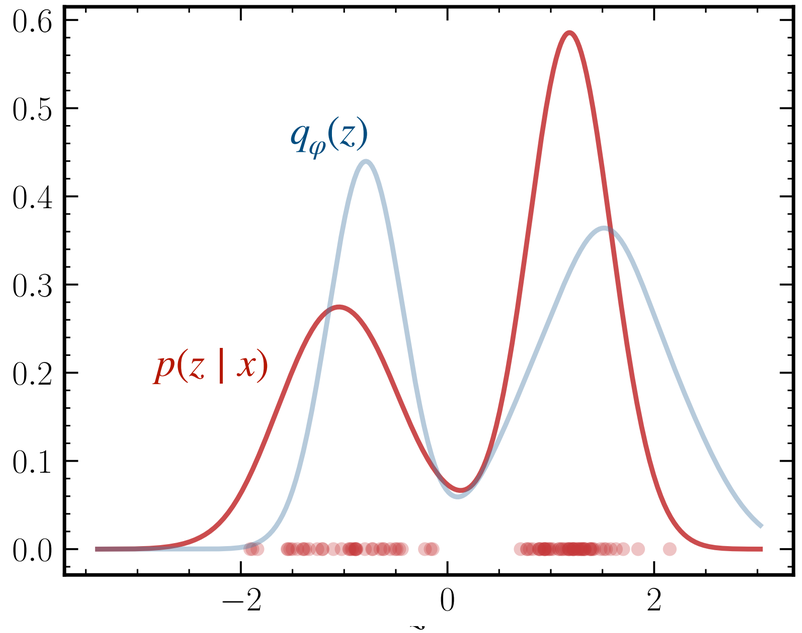
$$\underbrace{D_{\mathrm{KL}}\left(q_\varphi(z) \| p(z \mid x)\right)}_{\geq 0} = \underbrace{\log p(x)}_{\text{Evidence}} - \underbrace{\left\langle \log p_\vartheta(x, z) - \log q_\varphi(z) \right\rangle_{q_\varphi(z)}}_{\text{Evidence Lower BOund (ELBO)}}$$

Variational inference

A general-purpose technique for posterior estimation

$$D_{\mathrm{KL}}\left(q_\varphi(z) \| p(z \mid x)\right) = \log p(x) - \left\langle \log p_\vartheta(x, z) - \log q_\varphi(z) \right\rangle_{q_\varphi(z)}$$

≥ 0 : Evidence − Evidence Lower BOund (ELBO)

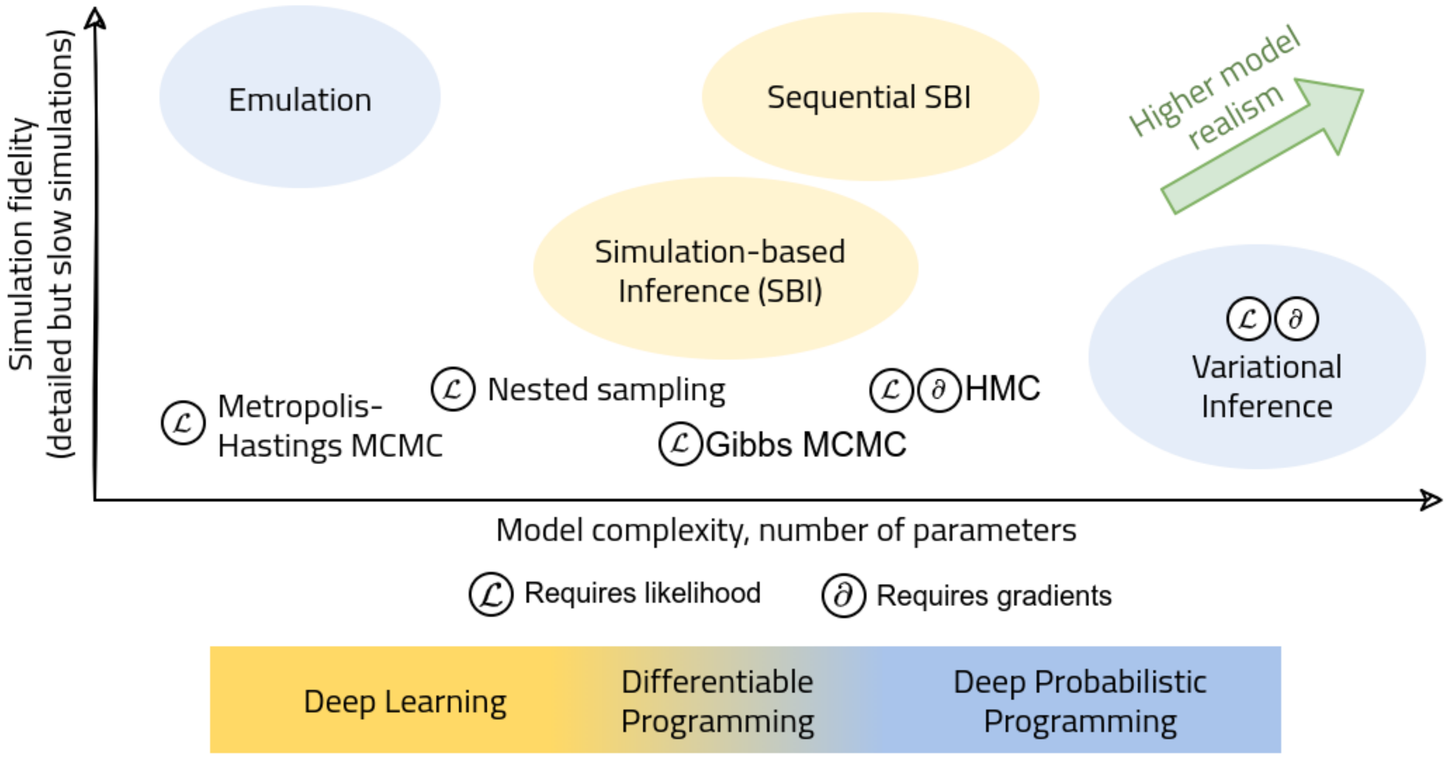


$$\begin{aligned}
\mathrm{ELBO} &= \left\langle \log p_\vartheta(x, z) - \log q_\varphi(z \mid x) \right\rangle_{q_\varphi} \\
&= \left\langle \log p_\vartheta(x \mid z) + \log p(z) - \log q_\varphi(z \mid x) \right\rangle_{q_\varphi} \\
&= \left\langle \log p_\vartheta(x \mid z) \right\rangle_{q_\varphi} - D_{\mathrm{KL}}\left(q_\varphi(z \mid x) \,\|\, p(z)\right)
\end{aligned}$$

"Reconstruction" "Regularization"

Variational inference

A general-purpose technique for posterior estimation: *optimization* instead of *sampling*



[EuCAPT White Paper 2021]

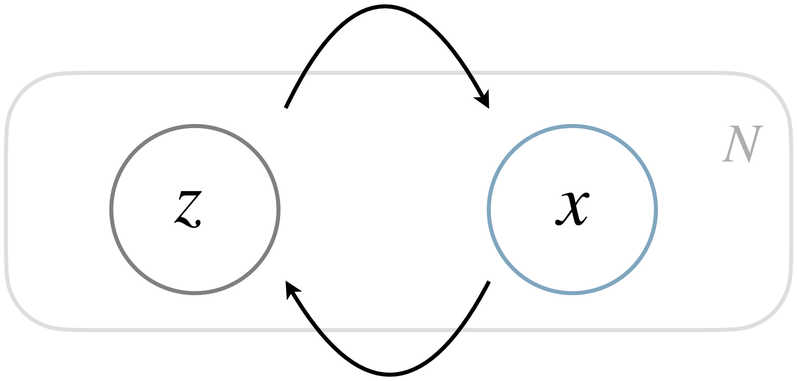
Can handle a huge number of parameters

A Bayesian latent-variable model optimized with variational inference

We're so back

Reverse process

$$p_\vartheta\left(x \mid z\right) \cdot p(z)$$



$$q_\varphi(z \mid x) \cdot p(x)$$

Forward process

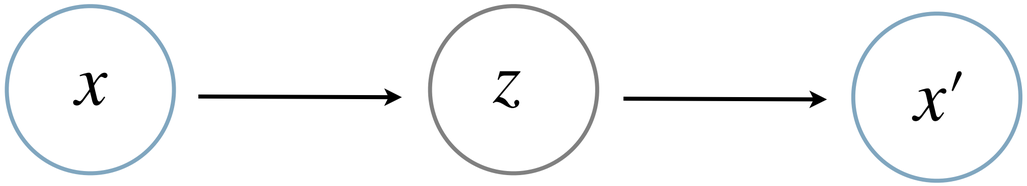
It's so over

Maximizing ELBO

$\equiv$ Minimizing *reverse* KL

$\equiv$ Aligning the forward and reverse processes

Minimize $\left\langle \log \frac{q(x,z)}{p(x,z)} \right\rangle$



Christopher Yau
@cwcyau

People do realise that a variational autoencoder comes from the application of variational inference to a Bayesian latent variable model right? It isn't an arbitrary loss function with a KL term stuck on to it with a tweakable parameter to balance the two?

Julian Togelius @togelius · Sep 22, 2021
No. I think of it as an arbitrary loss function and it works well for me. I'm in favor of arbitrary loss functions.

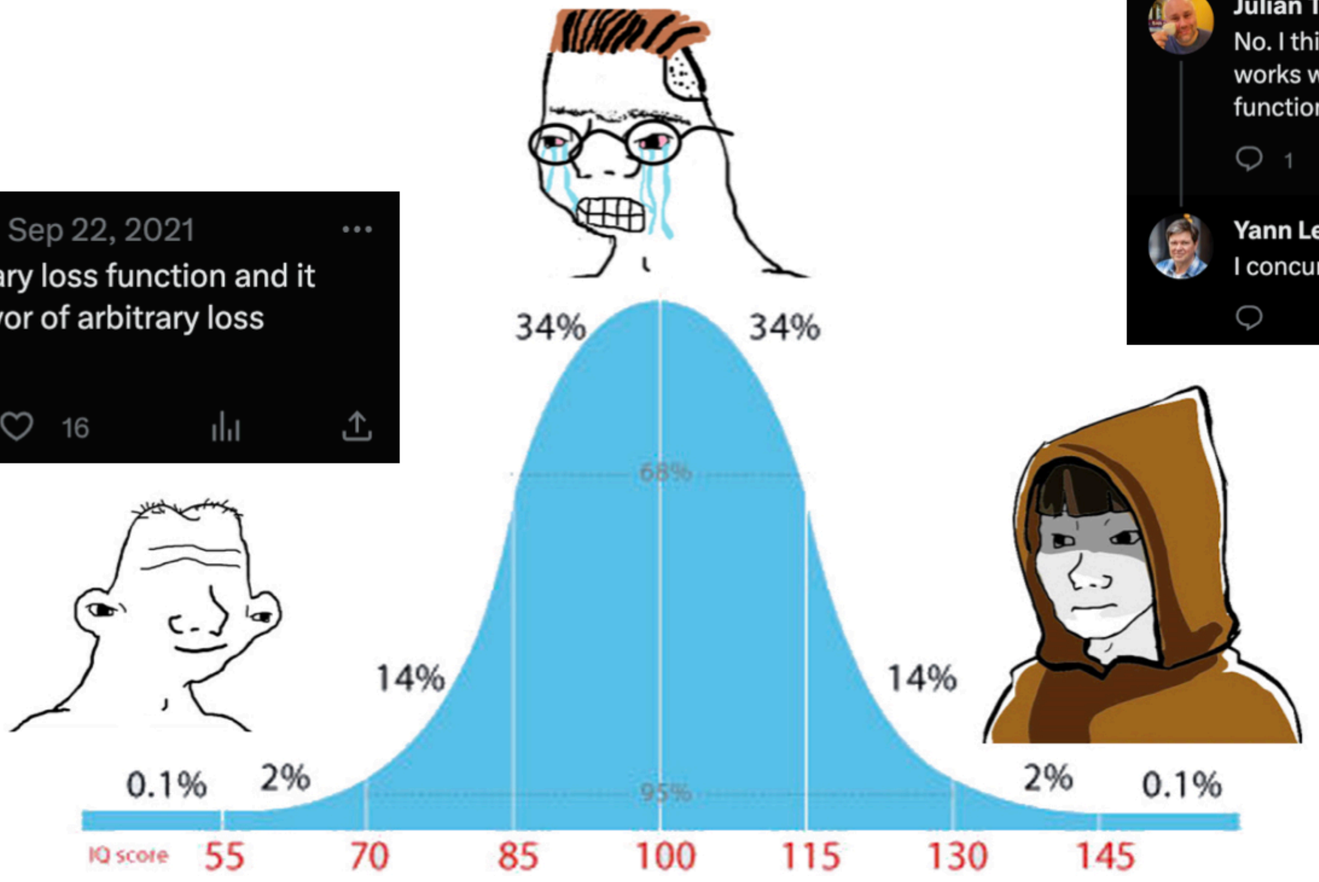
1 1 16

Julian Togelius @togelius · Sep 22, 2021
No. I think of it as an arbitrary loss function and it works well for me. I'm in favor of arbitrary loss functions.

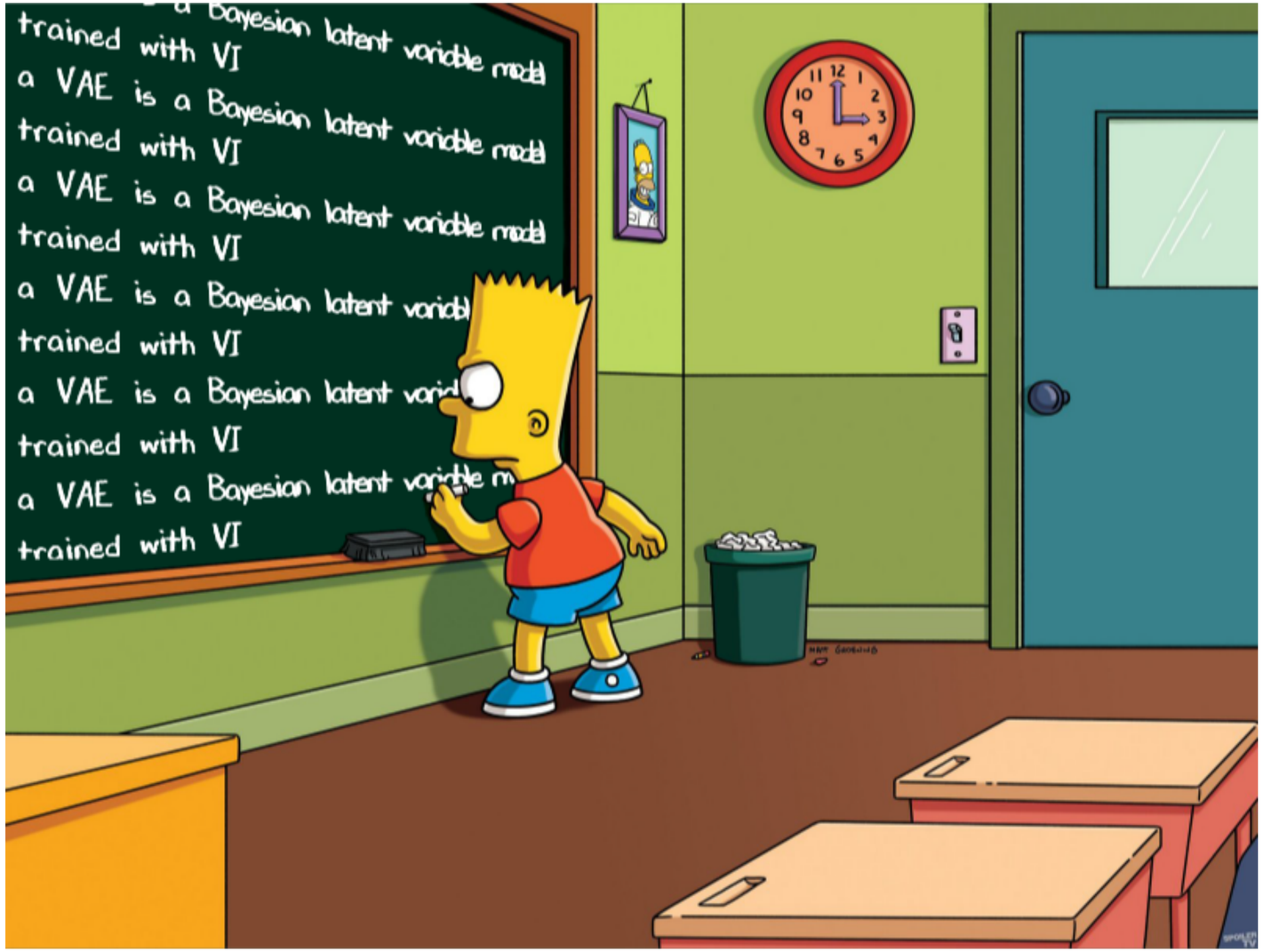
1 1 16

Yann LeCun @ylecun · Sep 22, 2021
I concur.

6



In This House We Believe

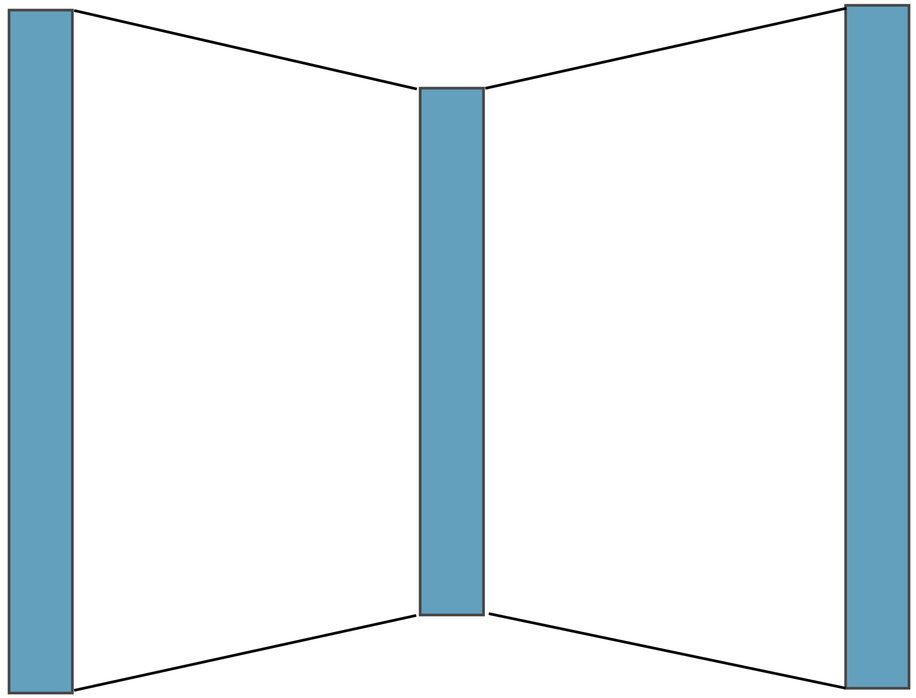


VAEs in practice

$$p(z) = \mathcal{N}(z; 0, I)$$

Prior

Original image



Reconstruction

$x \sim p(x)$ $\quad z \sim q_\varphi(z \mid x)$ $\quad x' \sim p_\vartheta(x \mid z)$

Encoder

$$q_\varphi(z \mid x) = \mathcal{N}(z; \mu, \sigma^2 \mathbb{I})$$

$$\mu, \sigma^2 = \mathrm{NN}_\varphi(x)$$

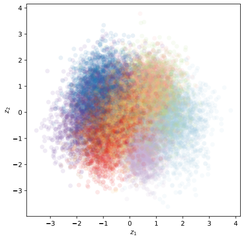
Decoder

Noise model / data likelihood

VAEs in practice

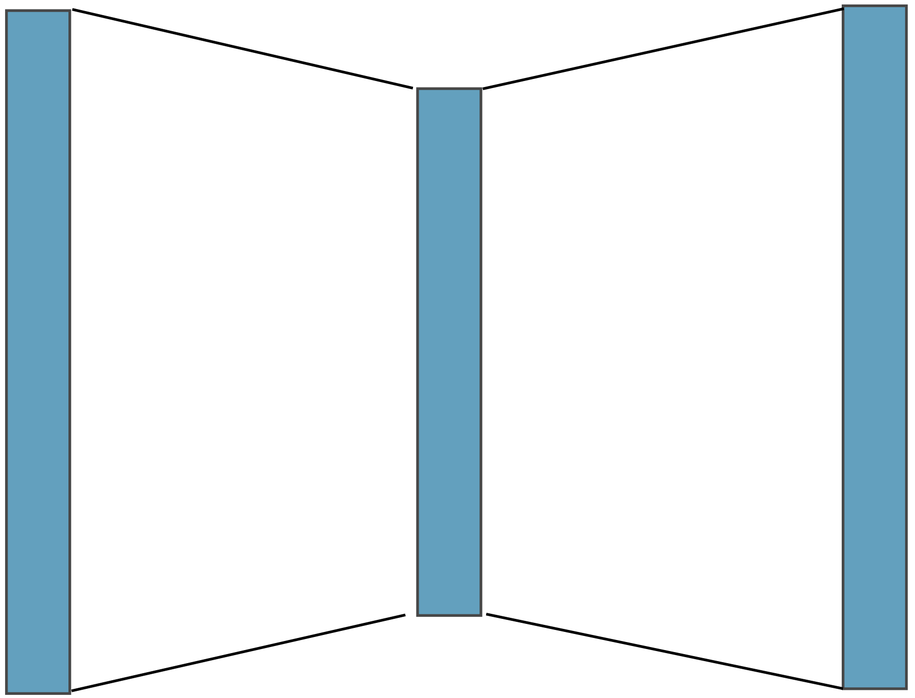
$$\text{ELBO} = \left\langle \log p_\vartheta(x \mid z) \right\rangle_{q_\varphi} - D_{\text{KL}}\left(q_\varphi(z \mid x) \,\|\, p(z)\right)$$

$q_\varphi(z \mid x)$



$$D_{\text{KL}}\left(q_\varphi(z \mid x) \,\|\, p(z)\right)$$

Regularization



$$\left\langle \log p_\vartheta(x \mid z) \right\rangle_{q_\varphi}$$

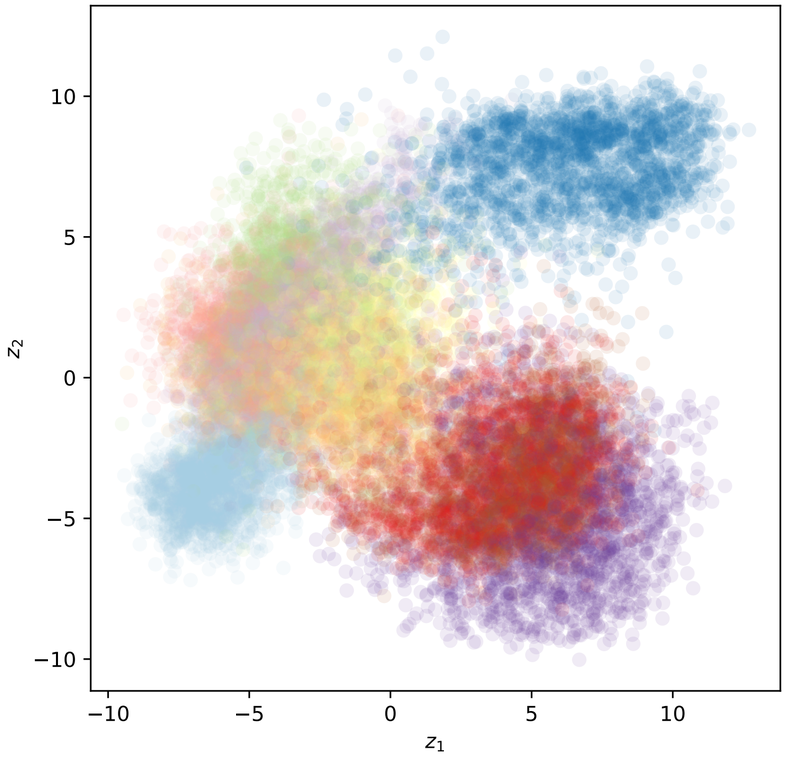
Reconstruction (e.g., MSE, …)

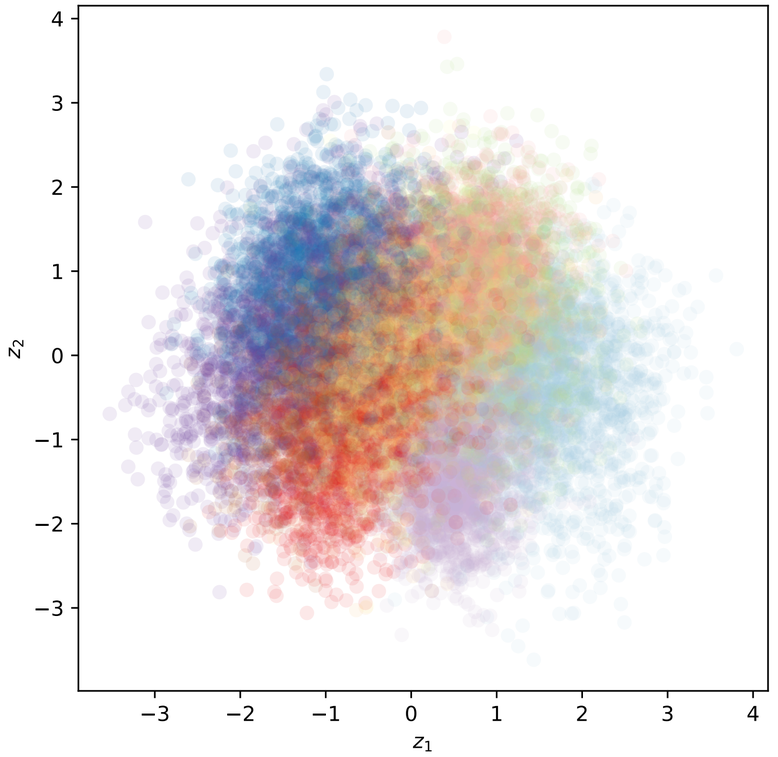
$$\|x - x'\|_2^2$$

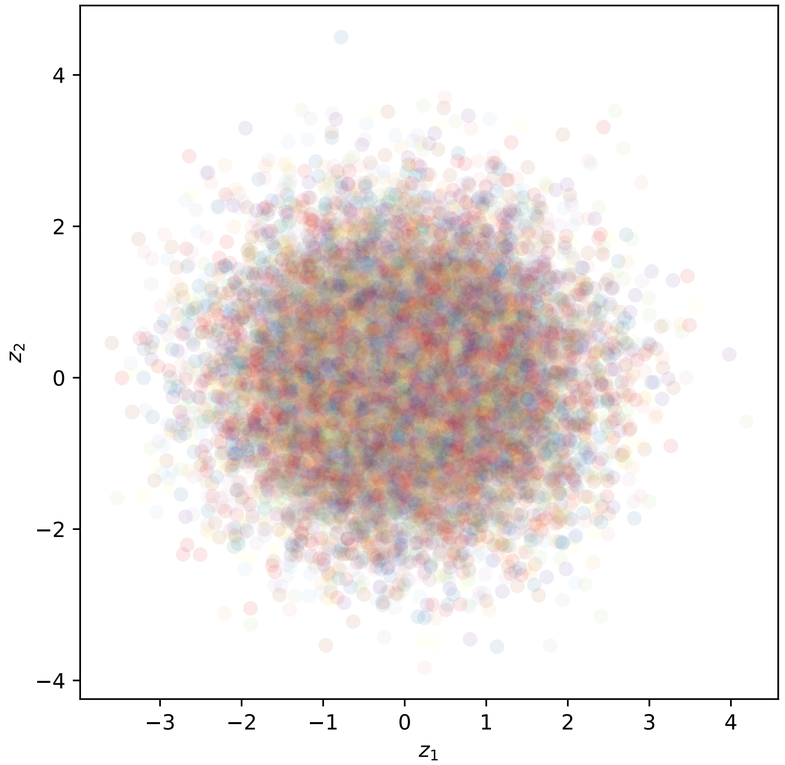
A semantically meaningful latent space

The KL-term enforces simplicity in the latent space, encouraging learned semantic structure and *disentanglement*

Pure reconstruction ⟷ *More latent regularization*







Rate-distortion theory: *neural compression*

Autoencoding is a form of (neural) compression!

$$\text{ELBO} = \left\langle \log p_\vartheta(x \mid z) \right\rangle_{q_\varphi} - D_{\text{KL}}\left(q_\varphi(z \mid x) \,\|\, p(z) \right)$$

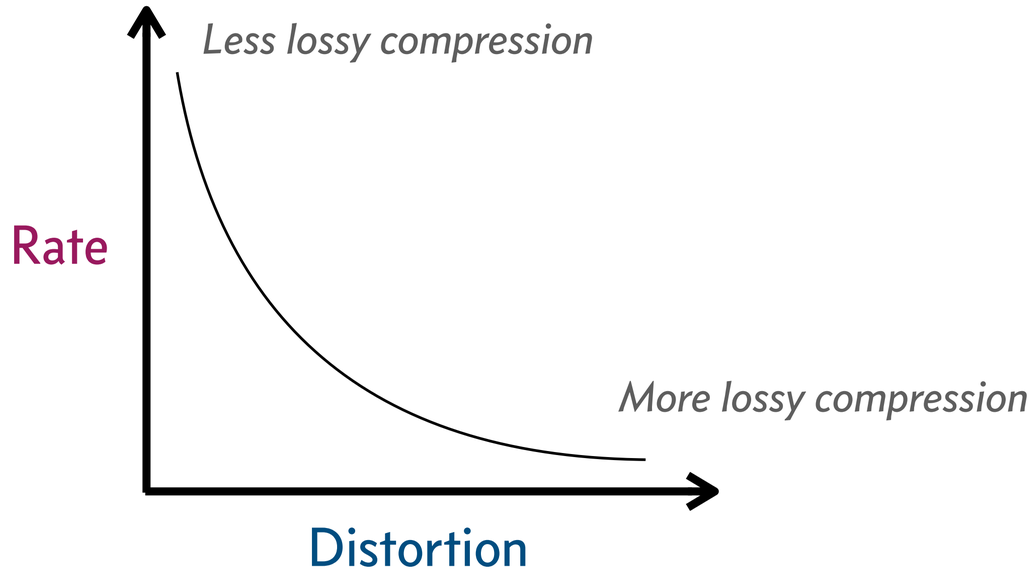
Distortion

"Information loss"

Rate

"Amount of compression"

Rate-distortion curve quantified this tradeoff



Controlling compression and disentanglement: β-VAEs

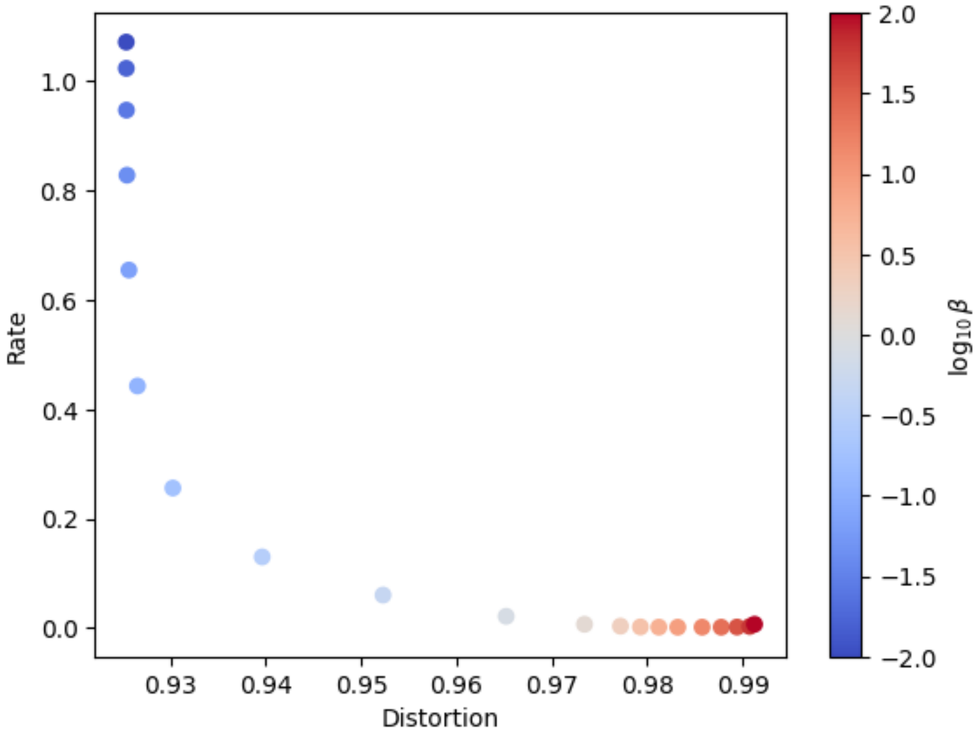
$$\text{ELBO} = \left\langle \log p_\vartheta(x \mid z) \right\rangle_{q_\varphi} - \beta \cdot D_{\text{KL}} \left(q_\varphi(z \mid x) \,\|\, p(z) \right)$$

Distortion | Rate

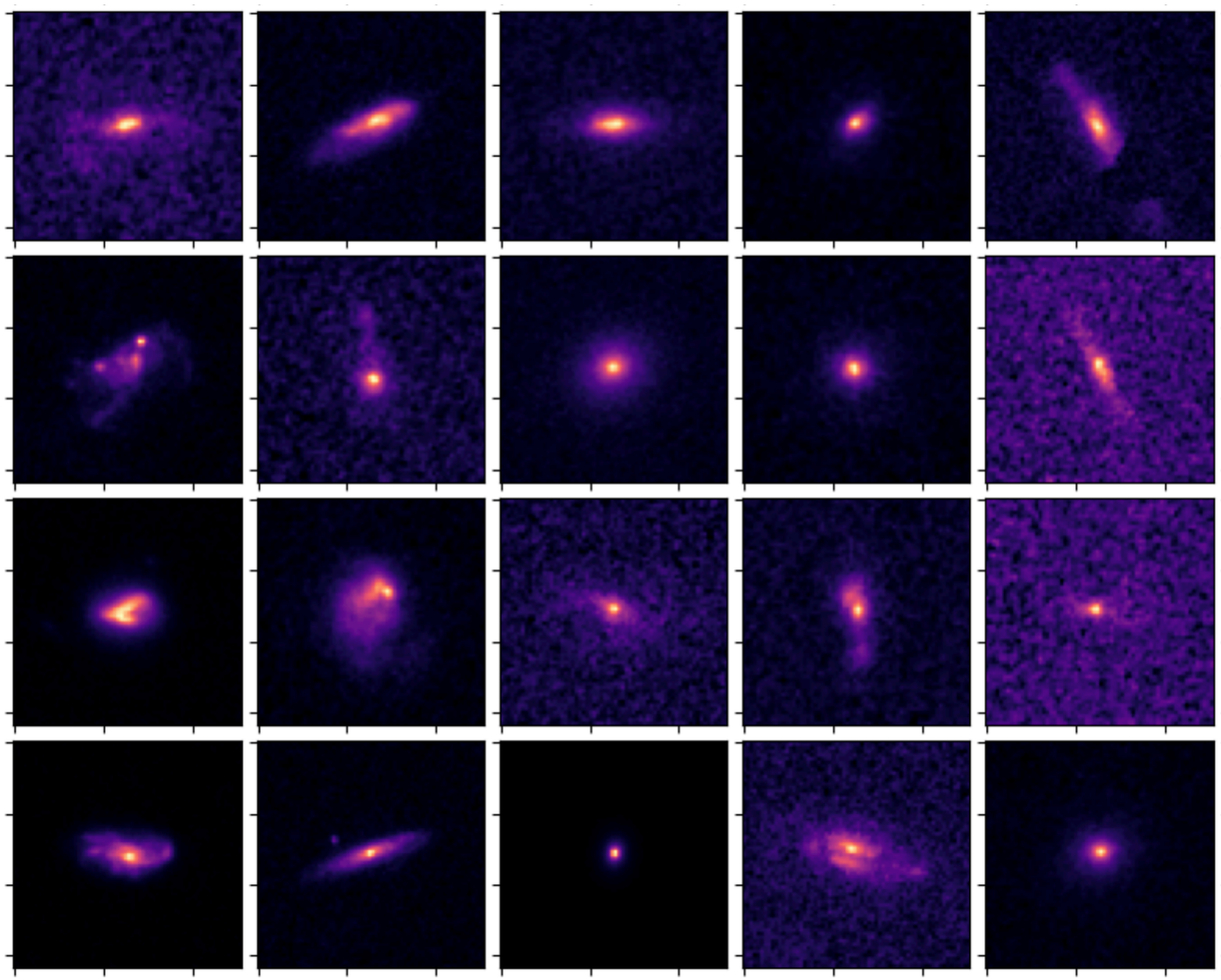
If the data-generating process is associated with a principled noise model, by using it (the *likelihood*) as the reconstruction loss we are aiming to reconstruct the mean data.

$$\log p(x \mid z; x') = -\frac{1}{2}\left(\frac{x - x'}{\sigma}\right)^2 + \log\left(\frac{1}{\sigma\sqrt{2\pi}}\right)$$

- Larger σ: More of the data variation is attributed to the likelihood → larger "β", more compression
- Smaller σ: Latents z try to capture more of the variation in the data (e.g. small perceptual features)



Tutorial Task 1



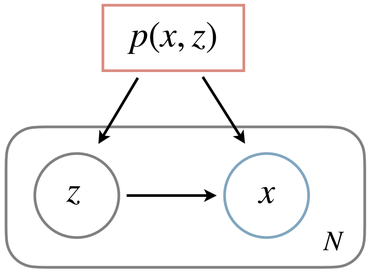
[Mandelbaum et al; https://zenodo.org/record/3242143]

- Construct a VAE and use it to build a generative model of galaxy images using samples form the HST COSMOS dataset
- Boilerplate code for training/reconstruction/ sampling for quick iteration
- Experiment with trade-offs between reconstruction quality and a disentangled latent space

Outline

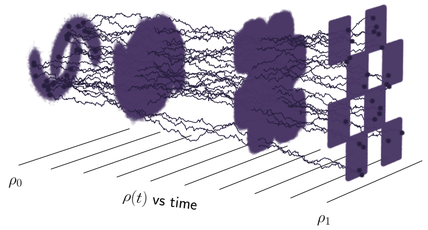
Why (deep) generative modeling?

What is it, and what can it do for you?



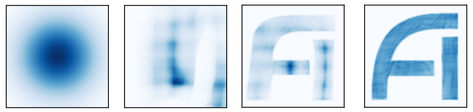
Variational auto encoders

Latent-variable modeling, and compression is all you need



Diffusion models

Models based on iterative refinement



Normalizing flows (and some other models)

Invertible transformations

Diffusion models: overview

Forward process *(adding noise)*

$x(t=0) \sim p(x)$

$x(t=1) \sim \mathcal{N}(0,1)$

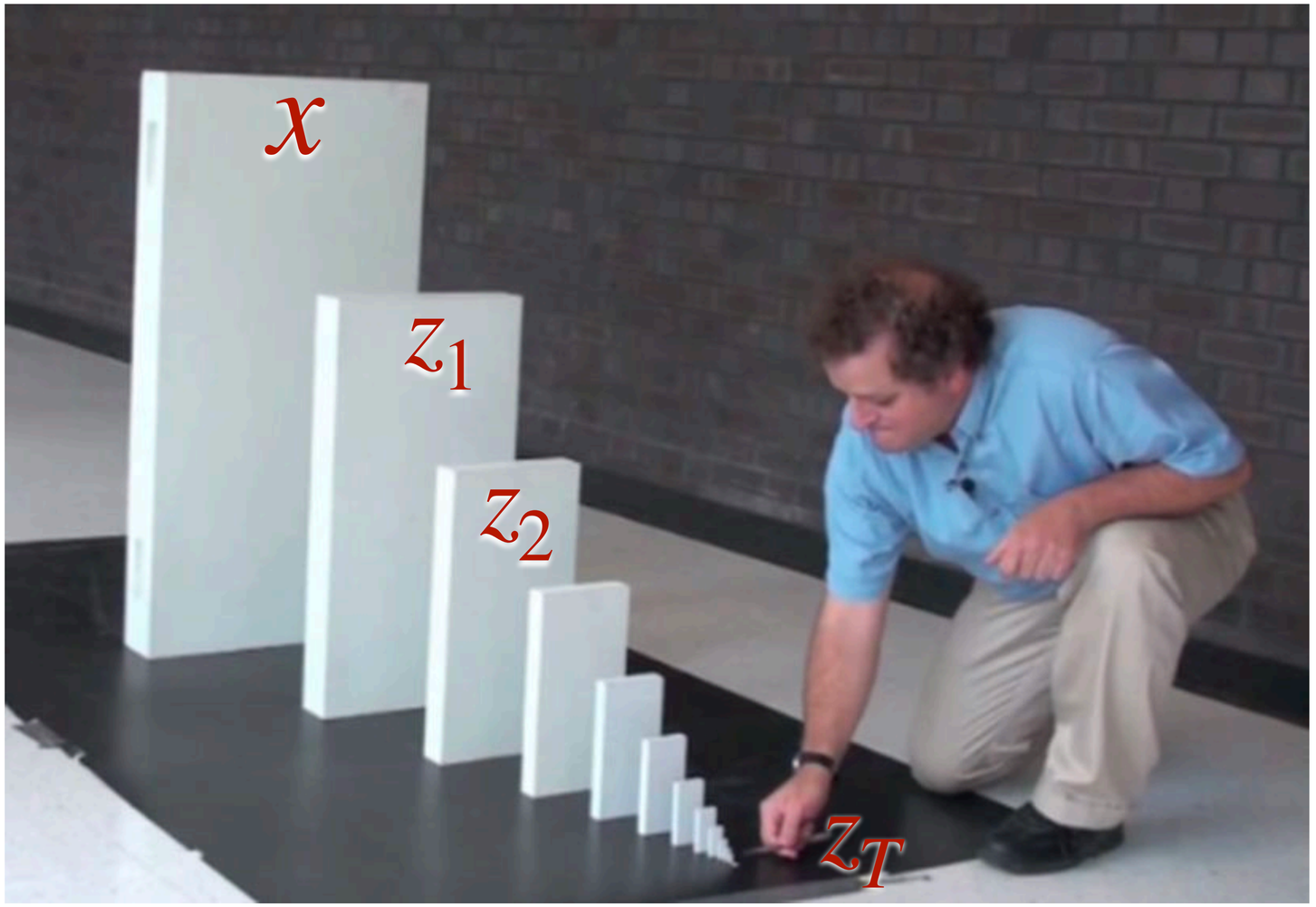
$x_{t-1} \sim p(x_{t-1} \mid x_t)$

Reverse process *(denoising)*

Prompt: *"A cat perched on an Ikea Poang chair"*

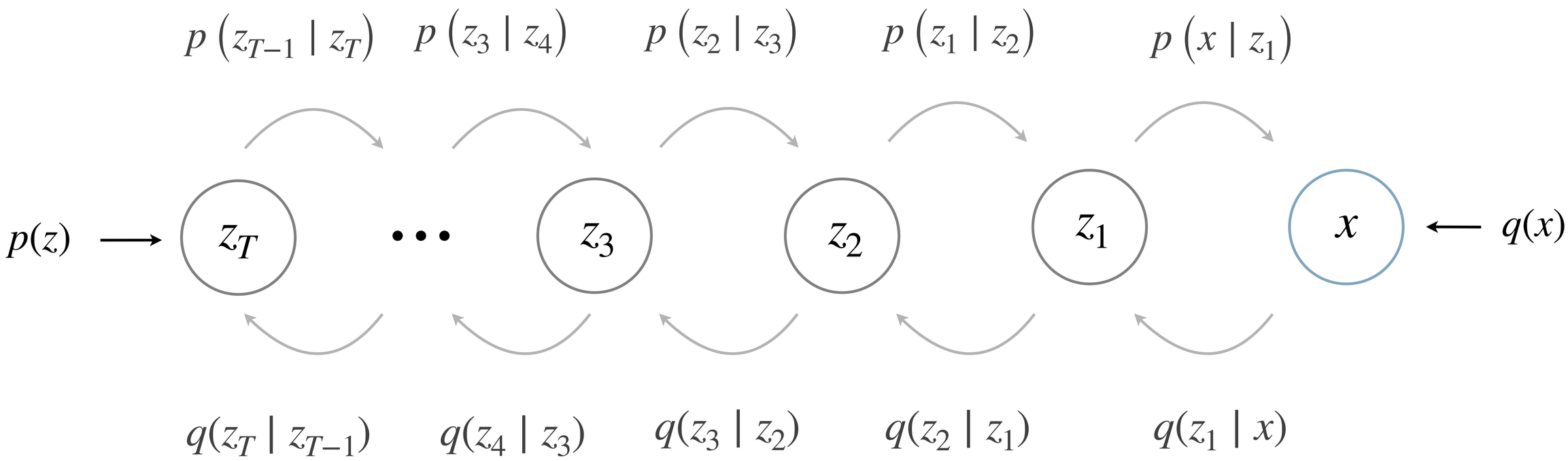
image ~ *p*(image | text prompt)

Towards diffusion: hierarchical VAEs



Towards diffusion: (Markovian) hierarchical VAEs

Reverse process $$p\left(x, z_1, z_2, \cdots, z_T\right) = p\left(z_T\right) p\left(z_{T-1} \mid z_T\right) \cdots p\left(z_1 \mid z_2\right) p\left(x \mid z_1\right)$$



Forward process $$q\left(x, z_1, z_2, \cdots, z_T\right) = q(x)\, q\left(z_1 \mid x\right)\, q\left(z_2 \mid z_1\right) \cdots q\left(z_T \mid z_{T-1}\right)$$

Towards diffusion: (Markovian) hierarchical VAEs

Diffusion models can be seen as hierarchical VAEs with a few restrictions:

- The forward *(encoding)* distribution prescribed as a Markov chain of Gaussians; it is not learned

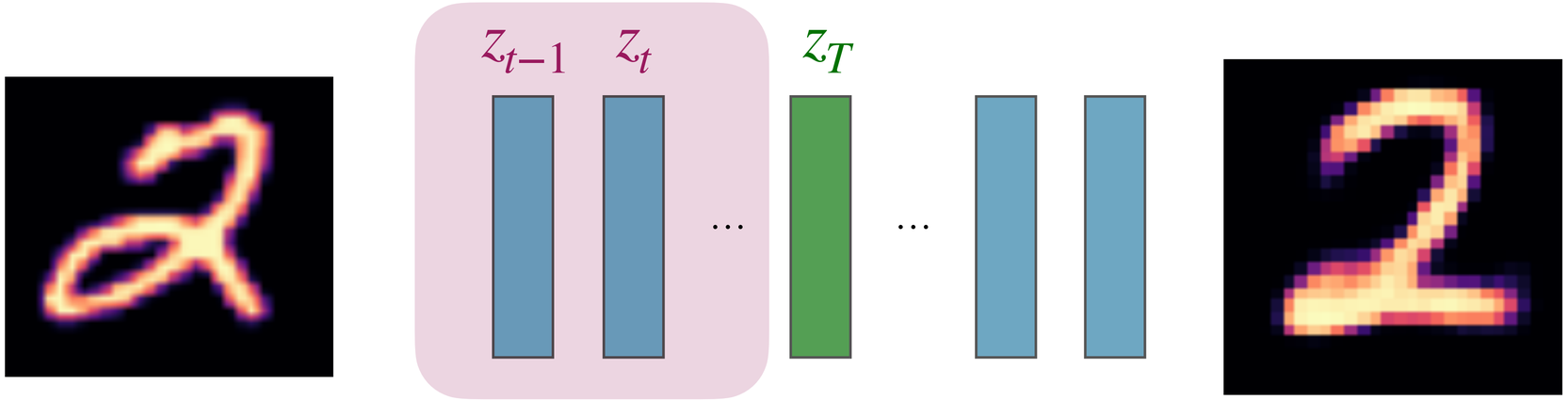
$$q\left(z_t \mid z_{t-1}\right) = \mathcal{N}\left(z_t; \alpha_t z_{t-1}, \beta_t\right)$$

- Distributions of latents at the final timestep T is a standard (unit) Gaussian

$$q(z_T \mid z_{T-1}, \ldots x) = \mathcal{N}(z_T; 0, \mathbb{I})$$

- The dimensionality of latents is the same as the data dimensionality

$$\dim(z_t) = \dim(x)$$

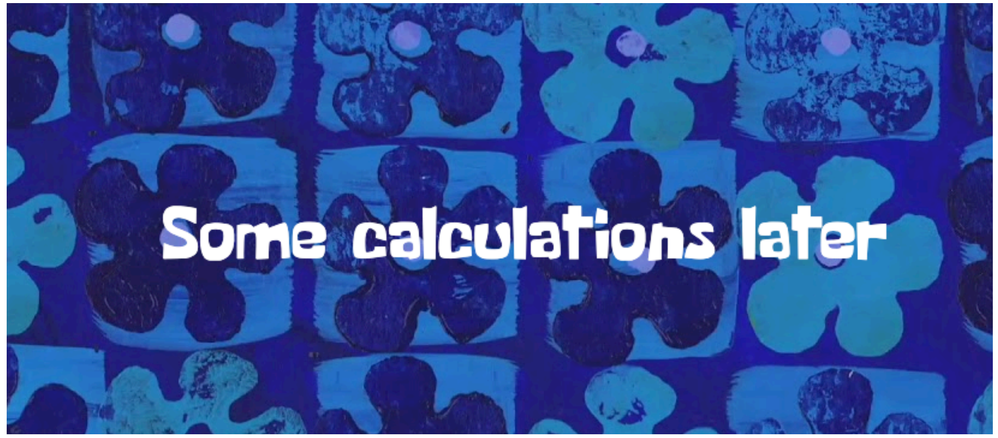


[Check expectations wrt]

Variational diffusion models

Align the forward and reverse distributions; variational lower bound (ELBO) as before

$$L = \left\langle \log \frac{q\left(x, z_1, z_2, \cdots, z_T\right)}{p\left(x, z_1, z_2, \cdots, z_T\right)} \right\rangle_{q(x)}$$



$$L = \left\langle \log p_\theta\left(x \mid z_1\right) \right\rangle_{q(z_1|x)} - D_{\mathrm{KL}}\left(q\left(z_T \mid x\right) \| p\left(z_T\right)\right) - \sum_{t=2}^{T} \left\langle D_{\mathrm{KL}}\left(q\left(z_{t-1} \mid z_t, x\right) \| p_\theta\left(z_{t-1} \mid z_t\right)\right) \right\rangle_{q(z_t|x)}$$

Reconstruction

(Noise model; no trainable parameters)

Prior regularization

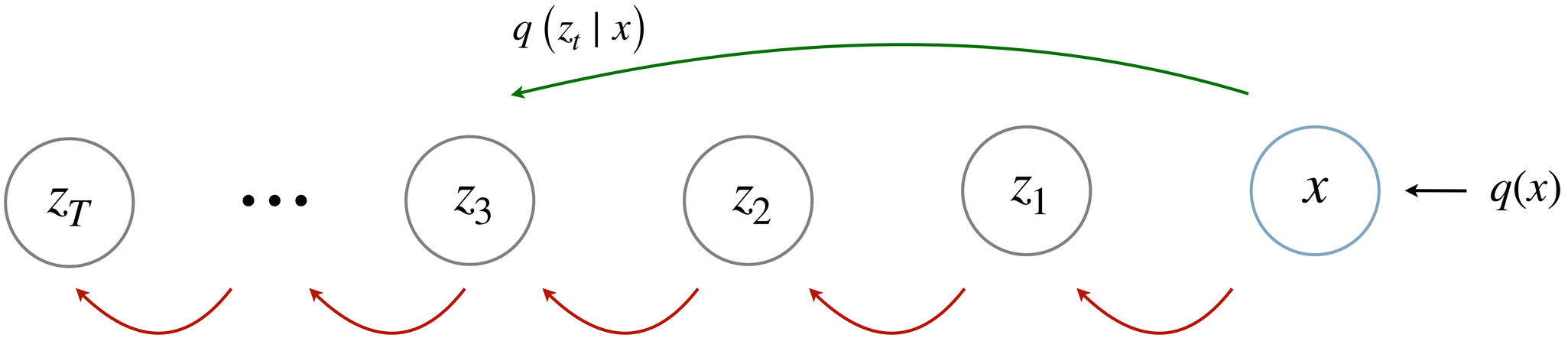
(No trainable parameters)

Denoising matching

The forward process and diffusion kernel

Predict arbitrary timestep without Markovian sampling

$$\sum_{t=2}^{T} \left\langle D_{\mathrm{KL}}\left(q\left(z_{t-1} \mid z_t, x\right) \| p_\theta\left(z_{t-1} \mid z_t\right)\right)\right\rangle_{q(z_t|x)}$$



Variance-preserving noise schedule

$$q\left(z_t \mid z_{t-1}\right) = \mathcal{N}\left(\sqrt{1-\beta_t} \cdot z_{t-1}, \beta_t\right)$$

$$z_t = \sqrt{1-\beta_t} \cdot z_{t-1} + \sqrt{\beta_t} \cdot \varepsilon$$

Diffusion kernel

$$q\left(z_t \mid x\right) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t} \cdot x, \sqrt{1-\bar{\alpha}_t}\right)$$

$$\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$$

The denoising objective

Given the nature of the forward (noising) process, $q\left(z_{t-1} \mid z_t, x\right)$ can be computed analytically

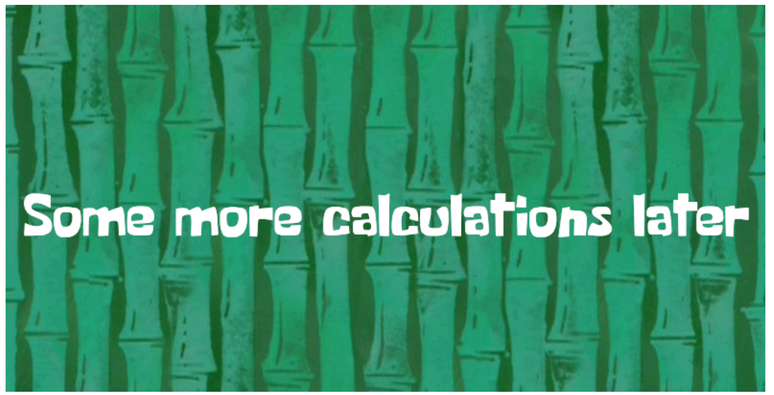
$$q\left(z_{t-1} \mid z_t, x\right) = \mathcal{N}\left(z_{t-1}; \mu_q(x_t, x_0), \sigma_q(t)\mathbb{I}\right)$$

Match ↕ ↕ *Set equal*

$$p_\theta\left(z_{t-1} \mid z_t, x\right) = \mathcal{N}\left(z_{t-1}; \mu_\theta(x_t, x_0), \sigma_\theta(t)\mathbb{I}\right)$$

Learnable denoising distribution; assume Gaussian

$$\sum_{t=2}^{T} \left\langle D_{\mathrm{KL}}\left(q\left(z_{t-1} \mid z_t, x\right) \| p_\theta\left(z_{t-1} \mid z_t\right)\right)\right\rangle_{q(z_t|x)}$$



Denoising loss

$$\frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}\left(1-\alpha_t\right)^2}{\left(1-\bar{\alpha}_t\right)^2} \left[\left\| \hat{x}_\theta\left(z_t, t\right) - x \right\|^2\right]$$

The denoising objectives

x-prediction; MLE

$$\frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}\left(1-\alpha_t\right)^2}{\left(1-\bar{\alpha}_t\right)^2} \left[\left\| \hat{x}_\theta\left(z_t, t\right) - x \right\|^2 \right]$$

ϵ-prediction; MLE

$$\frac{1}{2\sigma_q^2(t)} \frac{\left(1-\alpha_t\right)^2}{\left(1-\bar{\alpha}_t\right)\alpha_t} \left[\left\| \epsilon - \hat{\epsilon}_\theta\left(x_t, t\right) \right\|^2 \right]$$

ϵ-prediction; "simple"

$$\left\| \epsilon - \hat{\epsilon}_\theta\left(x_t, t\right) \right\|^2$$

Typical objective for training image diffusion models: *SOTA on many tasks!*

Simple objectives as a weighted sum of ELBOs

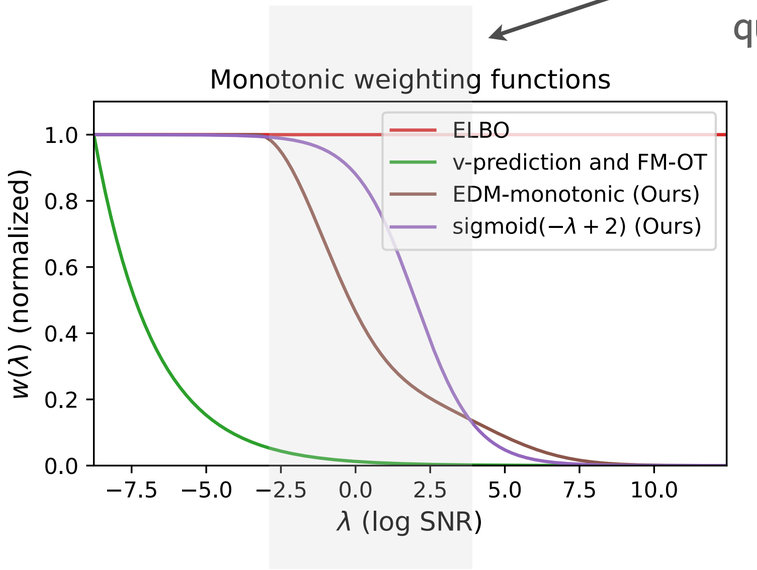
Kingma et al (2023) showed that common objectives can be written as a weighted sum (across different noise levels) of ELBOs

$$L_w(x) = \left\langle w(t) \cdot w_{\mathrm{ML}}(t) \left\| \epsilon - \hat{\epsilon}_\theta\left(x_t, t\right) \right\|^2 \right\rangle$$

Additional weighting (w_{ML}^{-1} for ϵ-prediction)

Weighting for ELBO/ ML objective

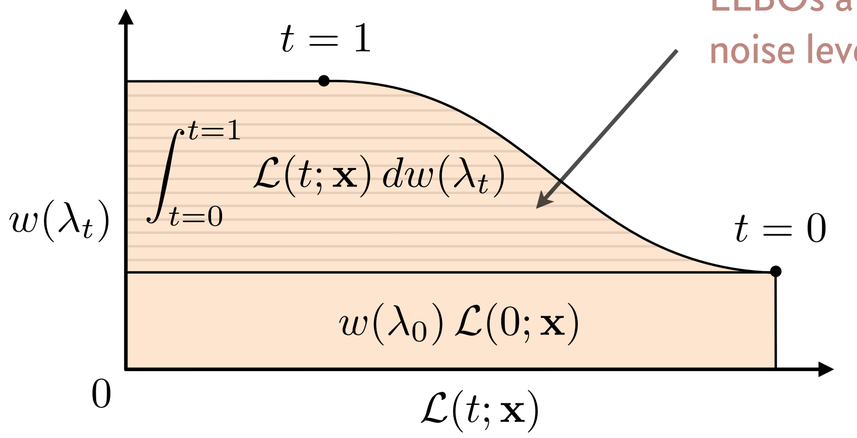
More important for perceptual quality?



$$L_w(x) = \left\langle w_{\mathrm{ML}}(t) \left\| \epsilon - \hat{\epsilon}_\theta\left(x_t, t\right) \right\|^2 \right\rangle_{p(w)}$$

Importance weighting of different noise levels

ELBOs at different noise levels



Interpretation: *data augmentation with additive Gaussian noise / data-distribution smoothing*

Continuous-time/SDE formulation

In the limit of infinite time steps, $\Delta_t \to 0$ and the forward diffusion process can be written as

$$x_t = \sqrt{1-\beta(t)\Delta_t}x_{t-1} + \sqrt{\beta(t)\Delta_t}\mathcal{N}(0,\mathbb{I})$$
$$\approx x_{t-1} - \frac{\beta(t)\Delta_t}{2}x_{t-1} + \sqrt{\beta(t)\Delta_t}\mathcal{N}(0,\mathbb{I})$$

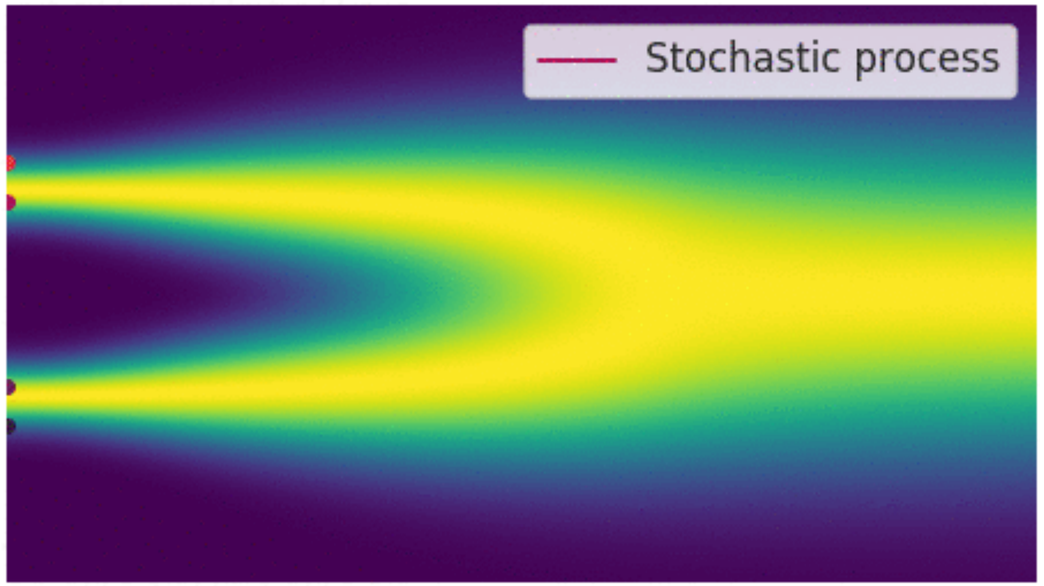
Which is an update rule corresponding to the Euler-Murayama discretization of the stochastic differential equation (SDE)

$$\mathrm{d}x_t = -\frac{1}{2}\beta(t)x_t\ \mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}w_t$$

Continuous-time/SDE formulation

The forward diffusion process defined by an SDE

[Song et al 2021]

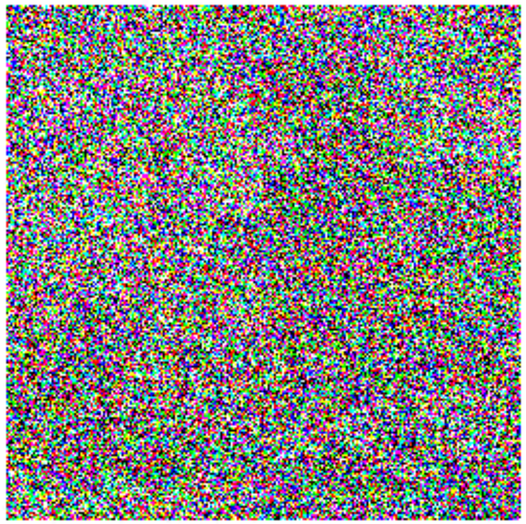


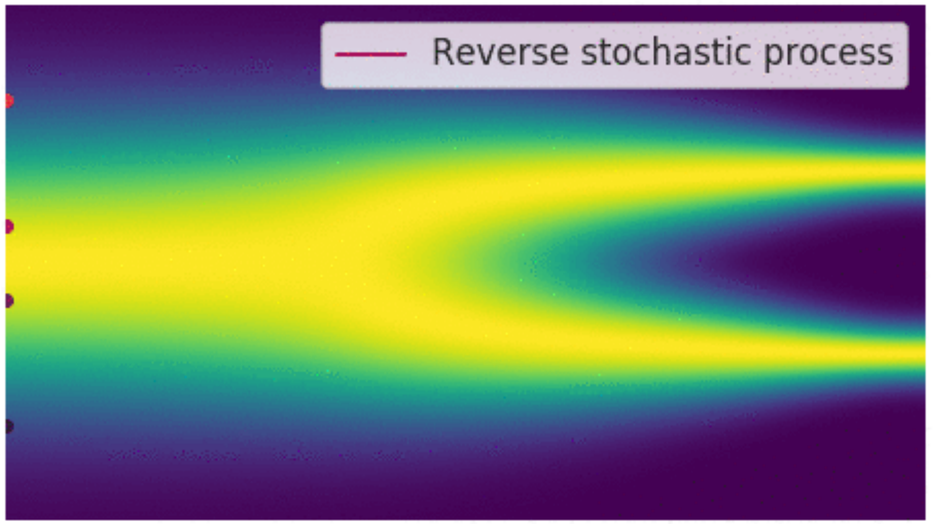
$$\mathrm{d}x_t = -\frac{1}{2}\beta(t)x_t\ \mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}w_t$$

The reverse SDE

The reverse process satisfies a reverse-time SDE that can be derived from the forward SDE and the score of the marginal distribution, $\nabla_{x_t} \log q(x_t)$

[Song et al 2021]





$$\mathrm{d}x_t = \left[-\frac{1}{2}\beta(t)x_t - \beta(t)\,\nabla_{x_t}\log q\left(x_t\right)\right]\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}w_t$$

Explain why x conditioning disappears

Denoising score matching

Need to compute the score $\nabla_{x_t} \log q(x_t)$

The *conditional* score $\nabla_{x_t} \log q(x_t \mid x)$ can be computed using the diffusion kernel

$$\nabla_{x_t} \log q(x_t \mid x) = -\frac{(x_t - x)}{\sigma_t^2} = -\frac{\epsilon}{\sigma_t}$$

Noise-prediction

$$\frac{1}{2\sigma_q^2(t)} \frac{\left(1-\alpha_t\right)^2}{\left(1-\bar{\alpha}_t\right)\alpha_t} \left[\left\| \epsilon - \hat{\epsilon}_\theta\left(x_t, t\right) \right\|^2 \right]$$

$\Longleftrightarrow$

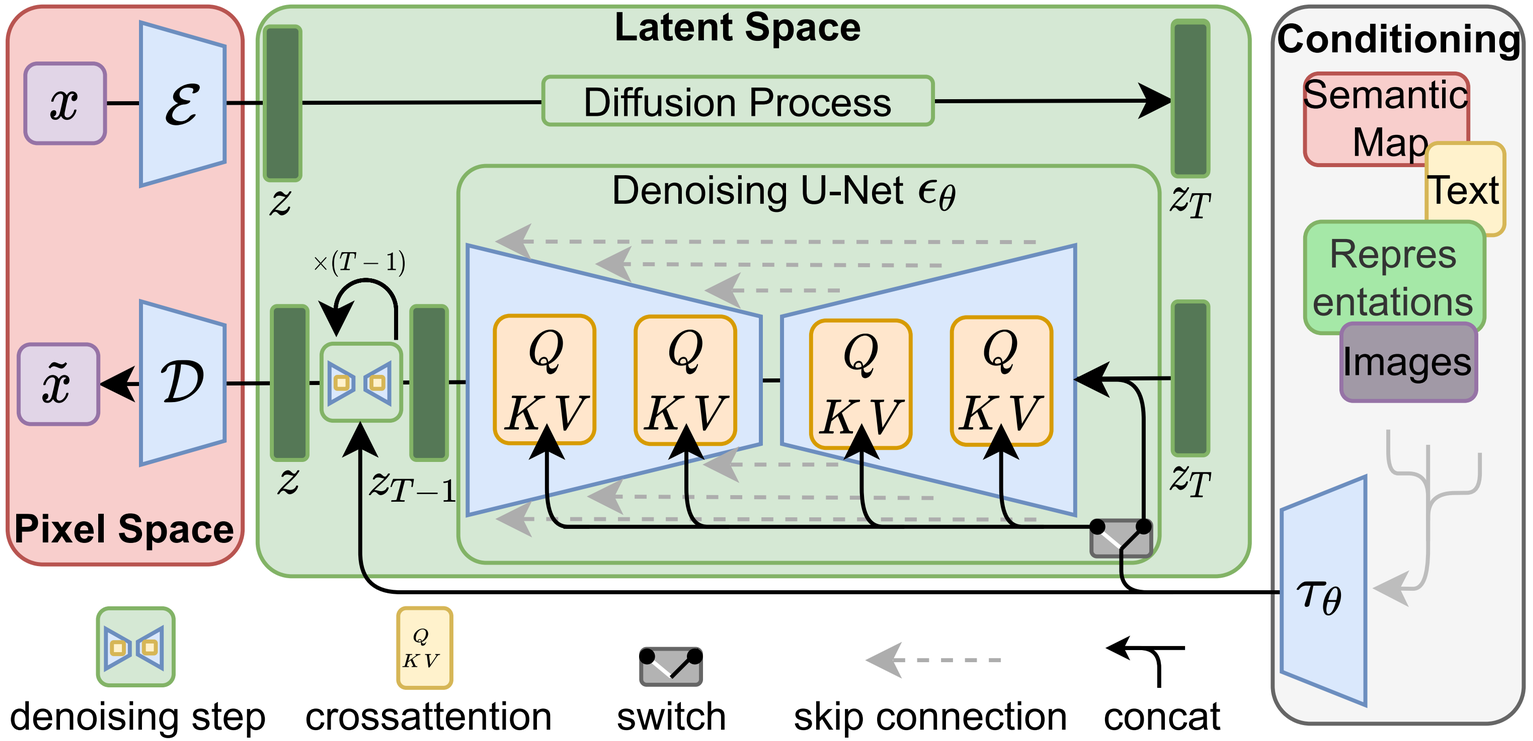
Score-matching

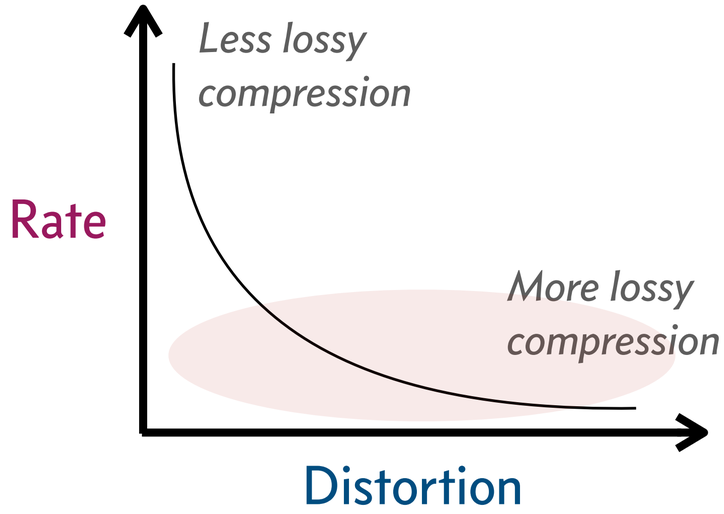
$$\frac{1}{2\sigma_q^2(t)} \frac{\left(1-\alpha_t\right)^2}{\alpha_t} \left[\left\| s_\theta\left(x_t, t\right) - \nabla \log p\left(x_t\right) \right\|^2 \right]$$

The noise- and score-prediction networks are equivalent up to a std-scaling

The noise/score-prediction model and *latent diffusion*

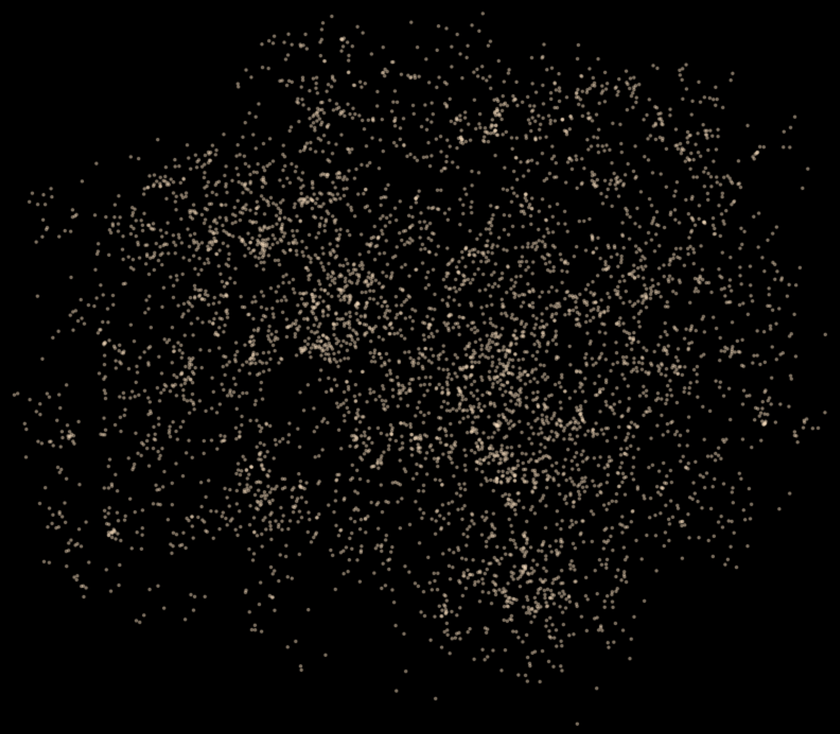
[Rombach et al 2021]



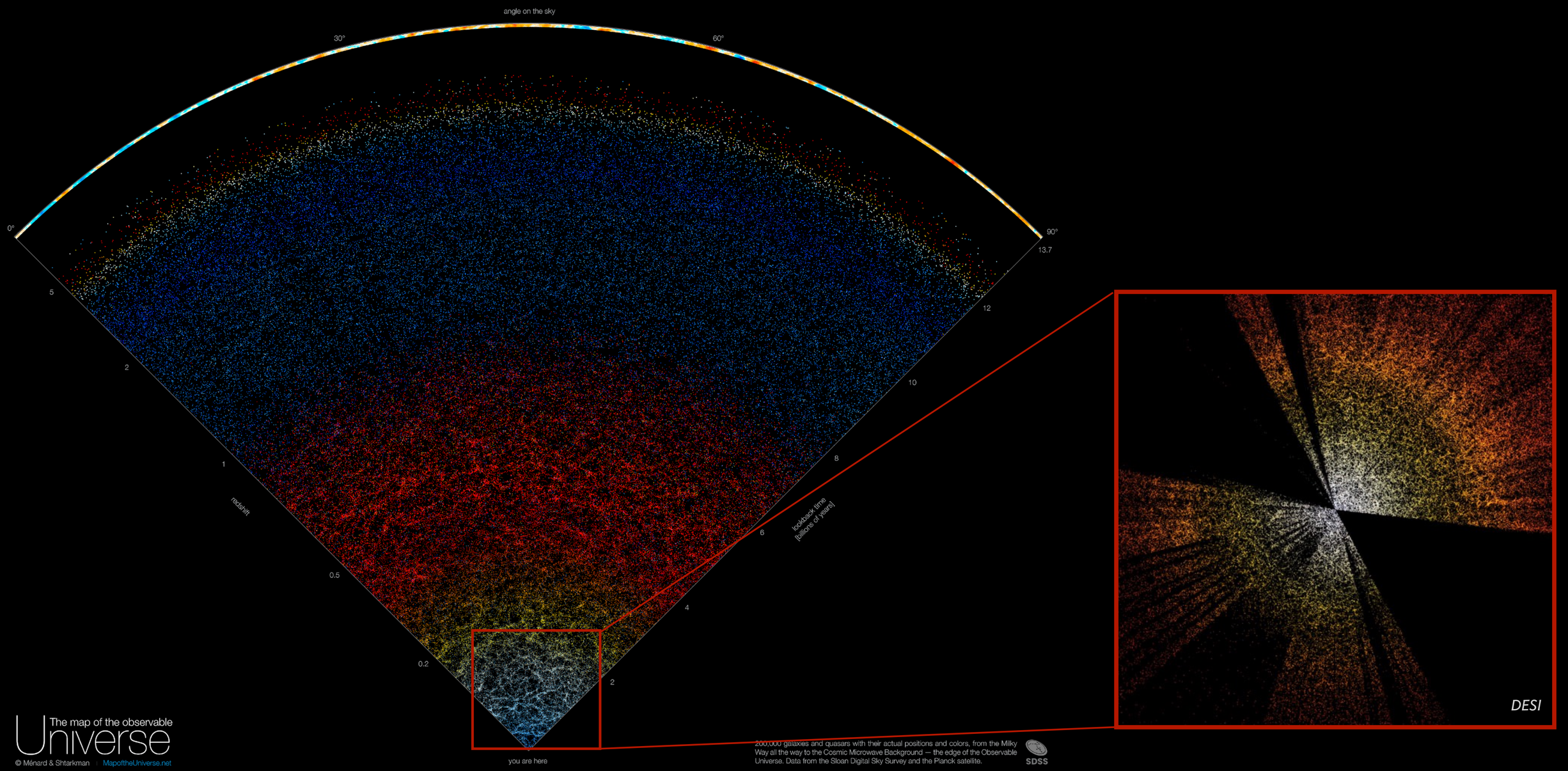


Perceptual compression while retaining semantically meaningful information

An application to *galaxy clustering*



Work with Carol Cuesta-Lazaro [MIT/IAIFI]



The map of the observable Universe

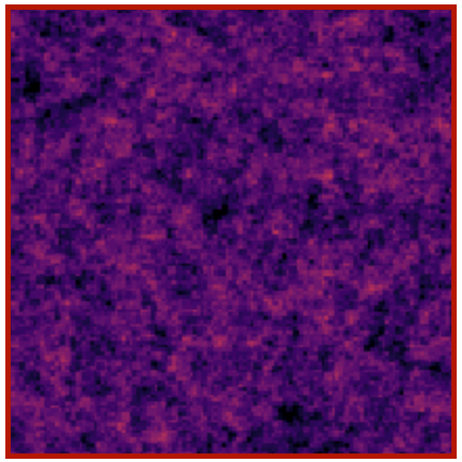
200,000 galaxies and quasars with their actual positions and colors, from the Milky Way all the way to the Cosmic Microwave Background — the edge of the Observable Universe. Data from the Sloan Digital Sky Survey and the Planck satellite.

SDSS

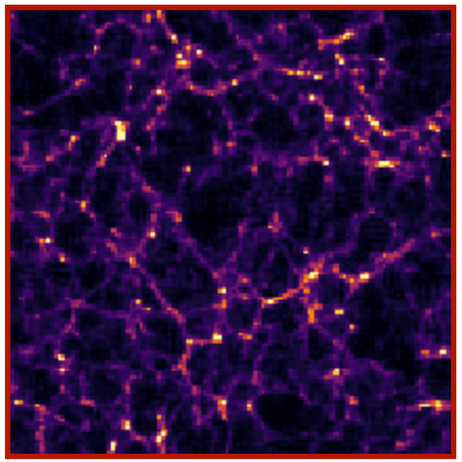
Forward modeling

Slide: Chirag Modi

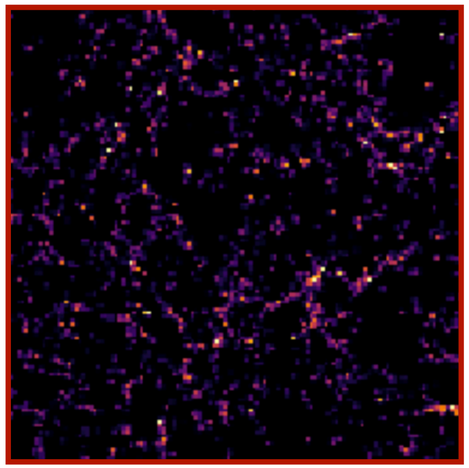
Gaussian field



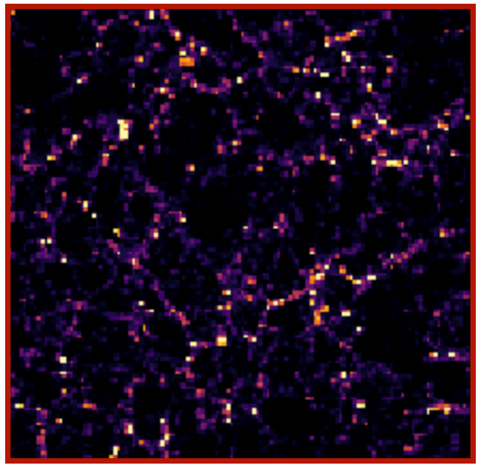
Dark matter field



Dark matter halos



Galaxies



+ Observational effects…

N-body simulation

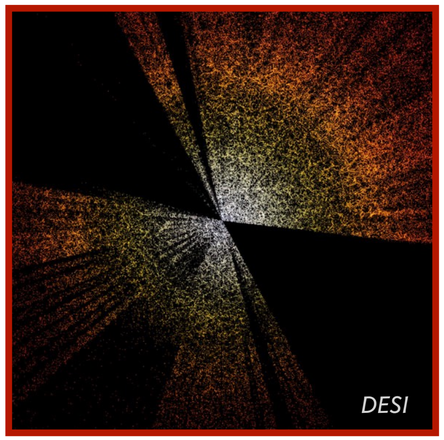
Group-finding algorithms

Galaxy-halo connection

The likelihood of a galaxy field

$p(x \mid \theta)$

$$p\Big(\ \Big| \ \Big)$$



Cosmology parameters
$\Omega_m, \sigma_8, \ldots$

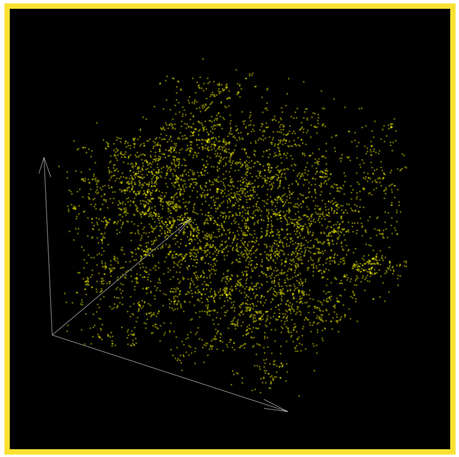
Galaxy-halo connection
(HOD, semi-analytical,…)

Simplify this

and/or this

Intractable!

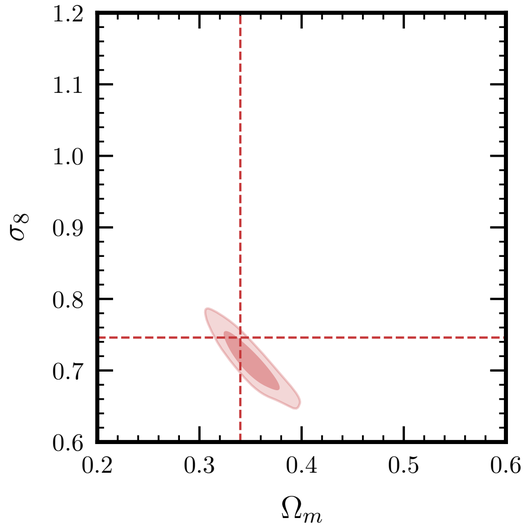
Emulation and inference



$$\sim p\left(\; \right| \; \text{Cosmology} \; \text{Galaxy-halo} \;)$$

Emulation/ sampling

$$p\left(\text{Cosmology } \Omega_m, \sigma_8 \; \right| \;)$$



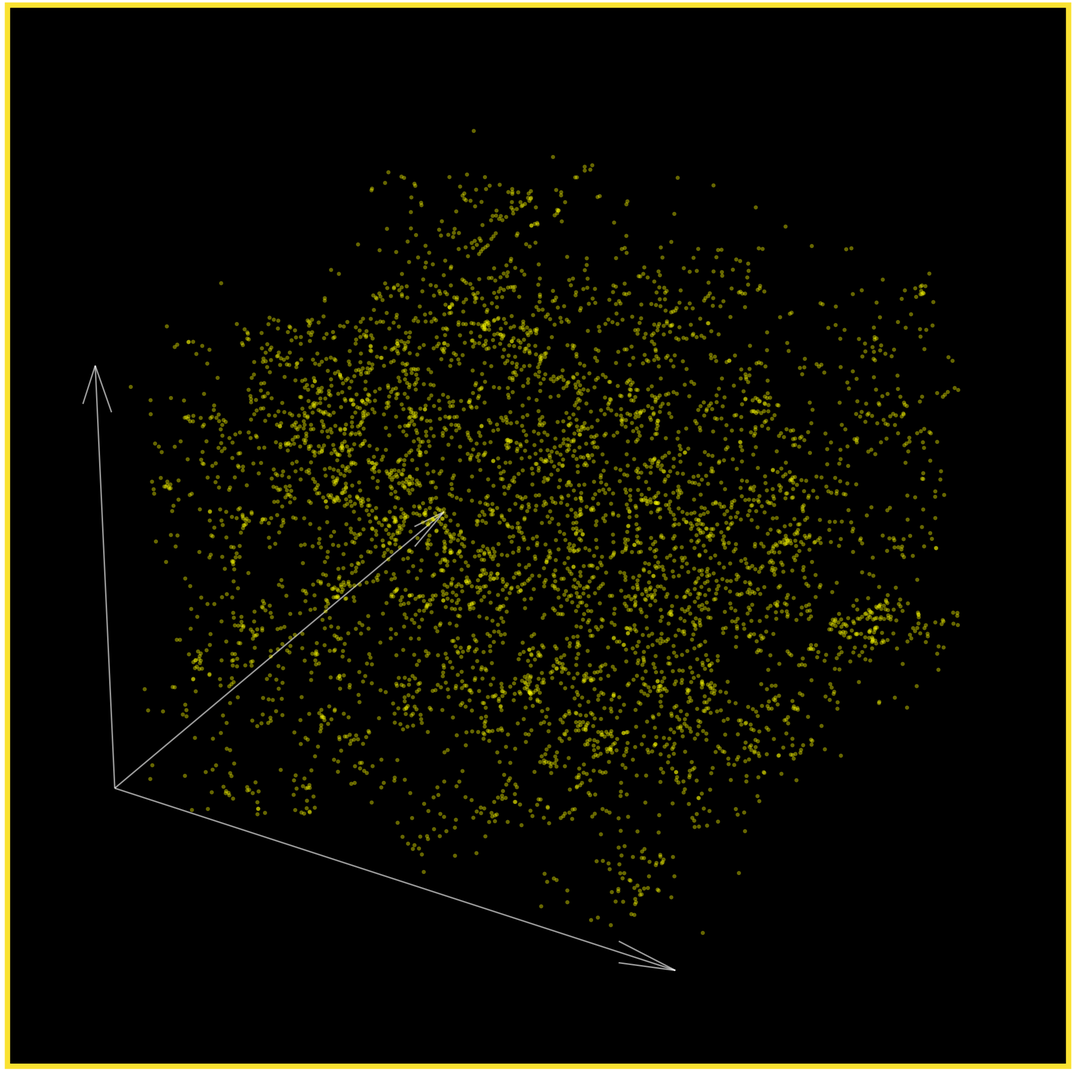
Parameter estimation

$\nabla_{\{\Omega_m, \sigma_8\}} p$ *Differentiable likelihood → even better!*

The diffusion score model

Want a score model that

- Operates on *sets* of varying cardinality
- Is permutation equivariant
- Efficiently captures correlation structure of point cloud

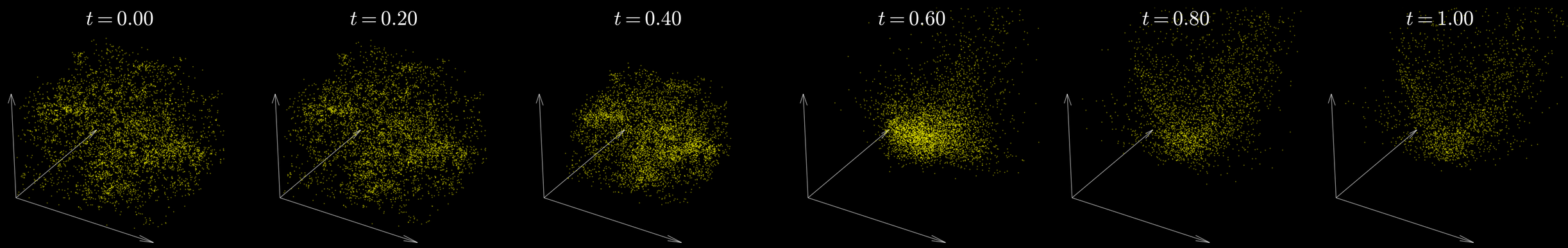


Graph neural network-guided diffusion on galaxies

SM, Cuesta-Lazaro [in prep]

Samples from *Quijote* [Villaescusa-Navarro et al, 2021].

Forward process *(adding noise)*



Reverse process *(denoising)*

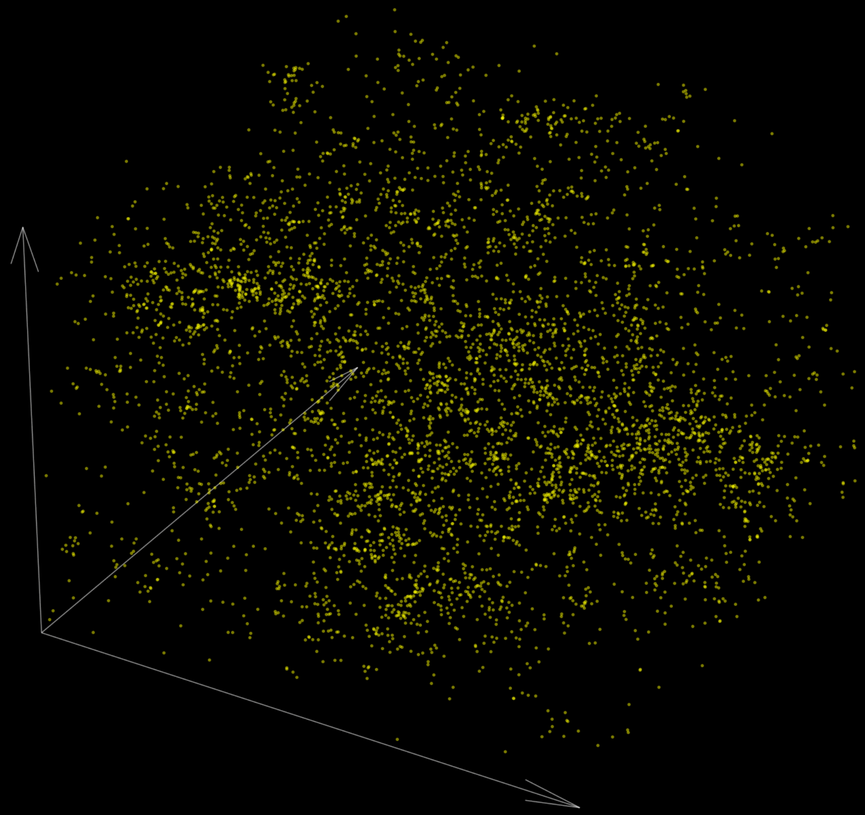
Cosmology parameters $\{\Omega_m, \sigma_8, \dots\}$
+ Galaxy-halo parameters

Diffusion on galaxies

SM, Cuesta-Lazaro [in prep]

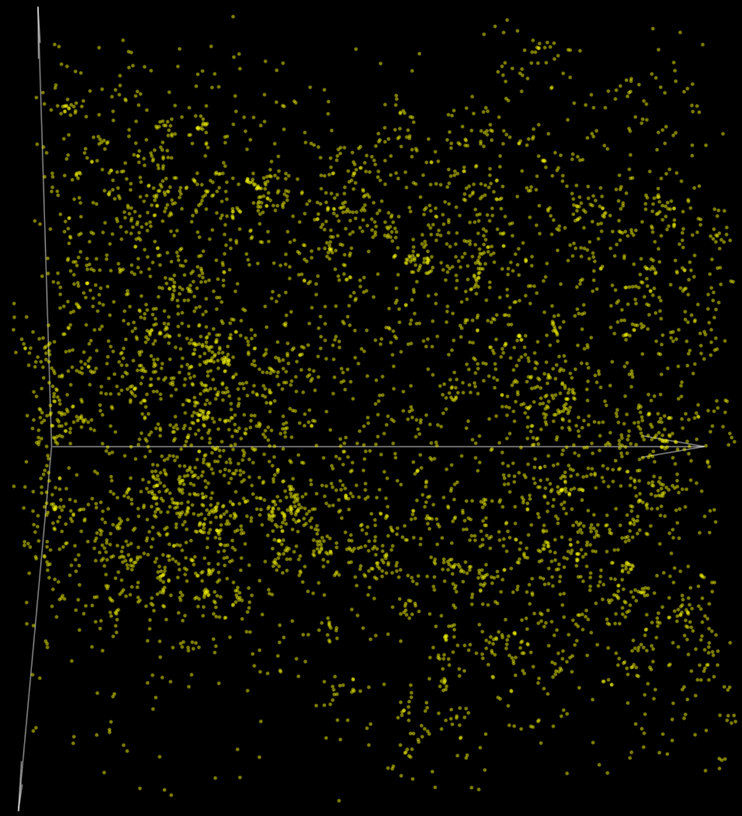
Diffusion process

$t = 0.00$



Conditional generation $x \sim p(x \mid \Omega_m, \sigma_8)$

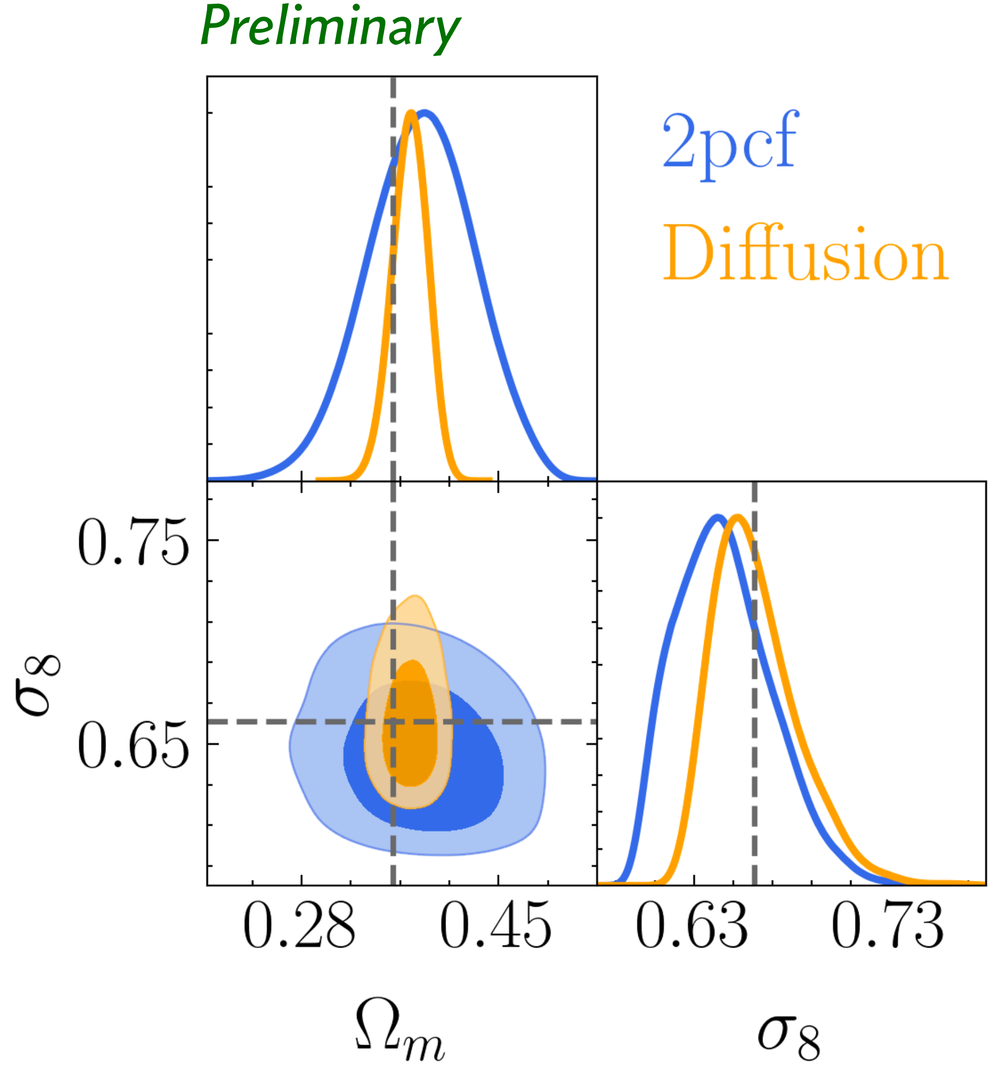
$\Omega_m = 0.10, \sigma_8 = 0.60$



Likelihoods and parameter inference

For a given dataset, can use the likelihood $p(x \mid \theta)$ for posterior parameter inference

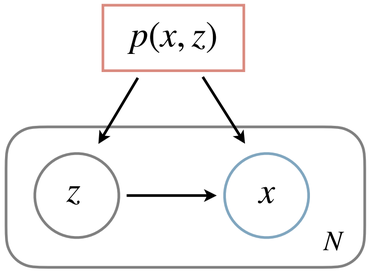
- Monte Carlo sampling (MCMC, nested sampling, HMC…)
- Variational inference



Outline

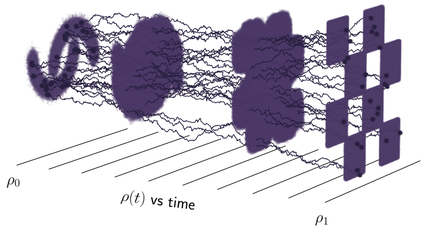
Why (deep) generative modeling?

What is it, and what can it do for you?



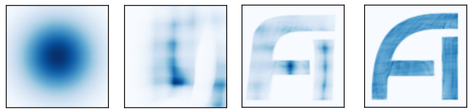
Variational auto encoders

Latent-variable modeling, and compression is all you need



Diffusion models

Models based on iterative refinement



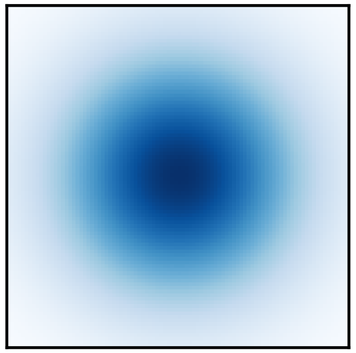
Normalizing flows (and some other models)

Invertible transformations

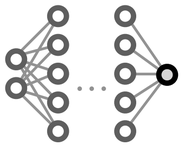
[Rezende & Mohamed 2015]

Normalizing flows

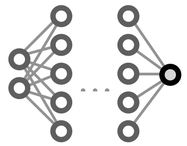
Base density



$\pi(u)$



$\theta = f(u)$



One-to-one transformation

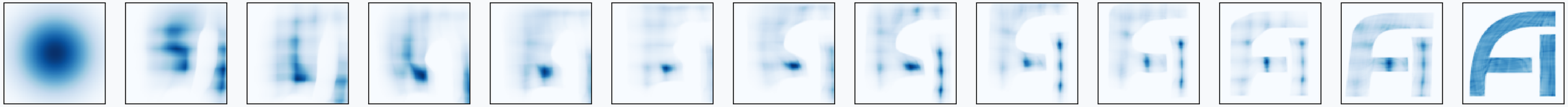
Tractable f^{-1} and $\det \nabla f$

(IAIFI logo)

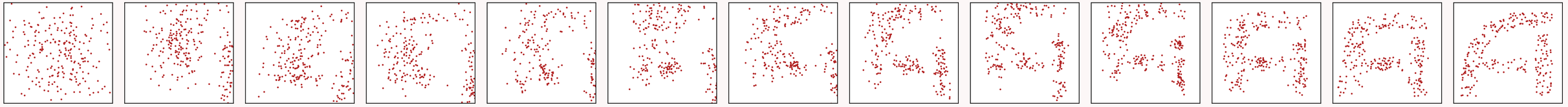
Target density

$$p(\theta) = \pi\left(f^{-1}(\theta)\right) \left|\det \nabla f\right|^{-1}$$

Efficient density estimation: $\log \hat{p}(\theta)$



Efficient sampling: $\theta \sim \hat{p}(\theta)$



Normalizing flows

The distribution $p(z)$ should

- *Have an easy-to-evaluate density*
- *Be easy to sample from* $z \sim p(z)$

Typically

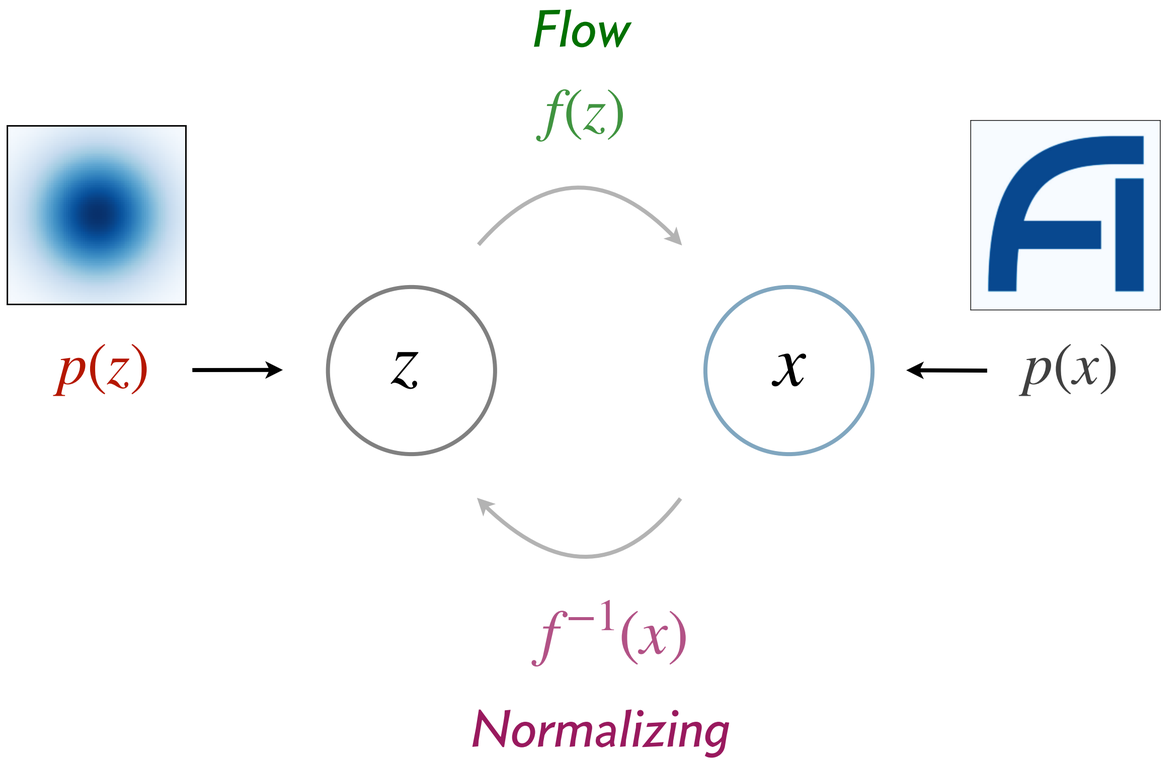
$$p(z) = \mathcal{N}(0, \mathbb{I})$$

The function f should be

- *One-to-one*
- *Differentiable*
- *Invertible*

} Diffeomorphism

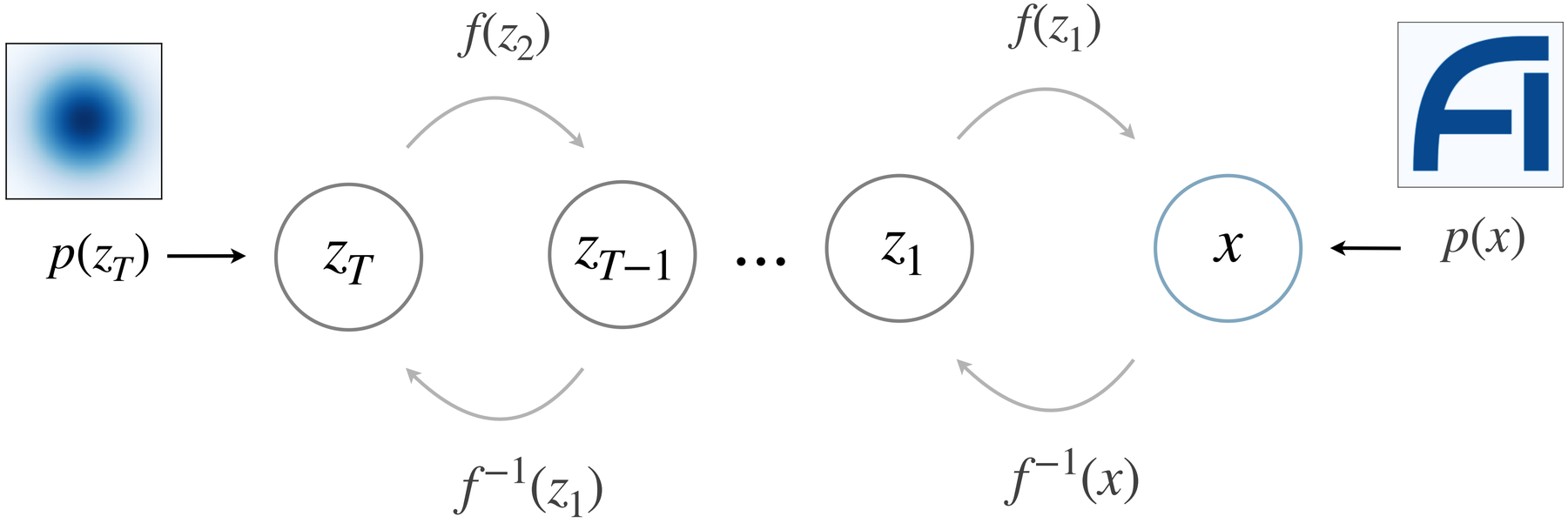
- *Tractable* f^{-1} *and* $\det \nabla f$



- Constrained form of the transformation can limit the expressivity of flows compared to e.g. diffusion models.
- However, for certain physics applications the transformation can be restricted in a specific, desired way; *see Miranda Cheng's lectures on Wednesday!*

Normalizing flows

Multiple flow transformation can be easily composed for e.g. expressivity



Computing $p(x)$: *change-of-variables formula*

$$\int p(x)dx = \int p(z)dz = 1$$

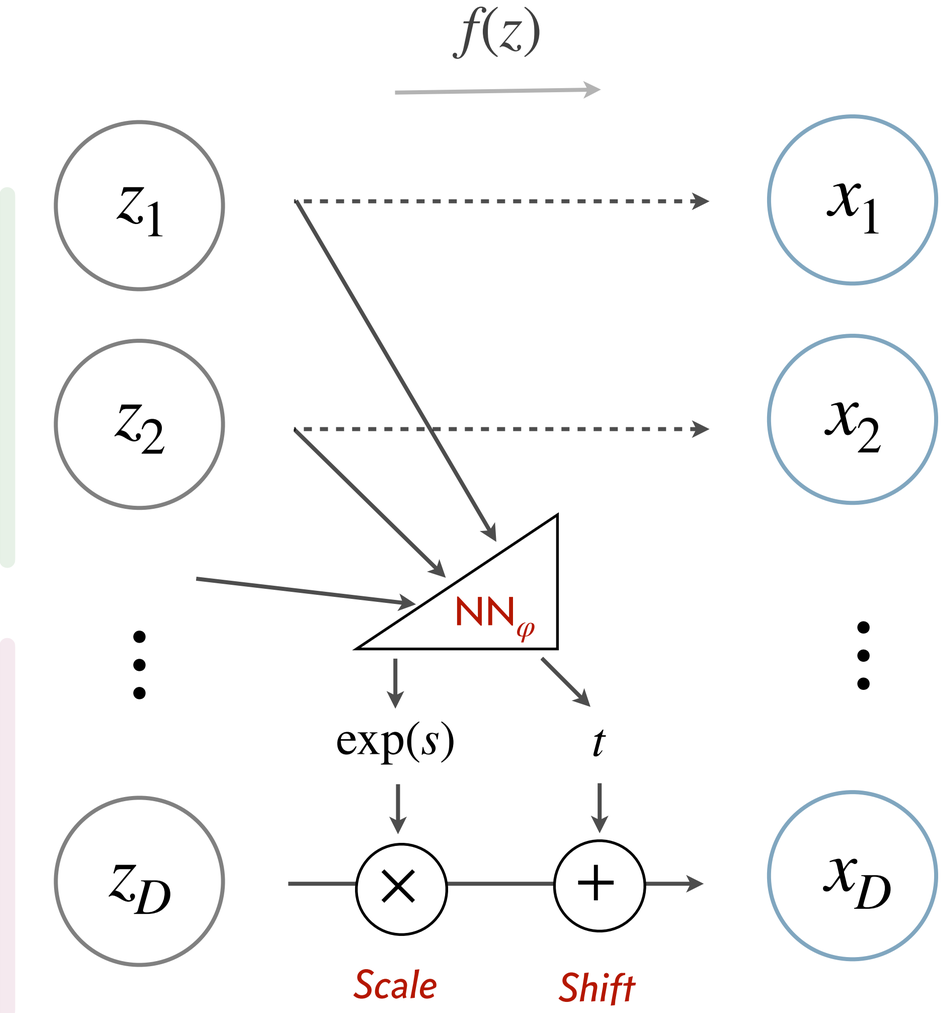
$$p(x) = \pi(z)\left|\frac{dz}{dx}\right| = p\left(f^{-1}(x)\right)\left|\frac{df^{-1}}{dx}\right| = p\left(f^{-1}(x)\right)|\det \nabla f|^{-1}$$

Train using maximum-likelihood objective

$$\varphi^* = \left\langle \arg\max_{\varphi} p\left(f_\varphi^{-1}(x)\right)|\det \nabla f_\varphi|^{-1} \right\rangle_{x\sim p(x)}$$

Simple flow transformations

Example: *Affine coupling flow* [Dinh et al 2016]



$z_{1:d}$ don't change

$z_{d+1:D}$ transform conditioned on $z_{1:d}$

Transformation ✅

$$x_{d+1:D} = z_{d+1:D} \odot \exp\left(s\left(x_{1:d}\right)\right) + t\left(x_{1:d}\right)$$

⇕

Inverse ✅

$$z_{d+1:D} = \left(x_{d+1:D} - t\left(x_{1:d}\right)\right) \odot \exp\left(-s\left(x_{1:d}\right)\right)$$

Jacobian determinant ✅

$$\det(\nabla f) = \prod_{j=1}^{D-d} \exp\left(s\left(z_{1:d}\right)\right)_j = \exp\left(\sum_{j=1}^{D-d} s\left(z_{1:d}\right)_j\right)$$

+ Switch up order of transformed variables at every transformation

Continuous-time normalizing flows

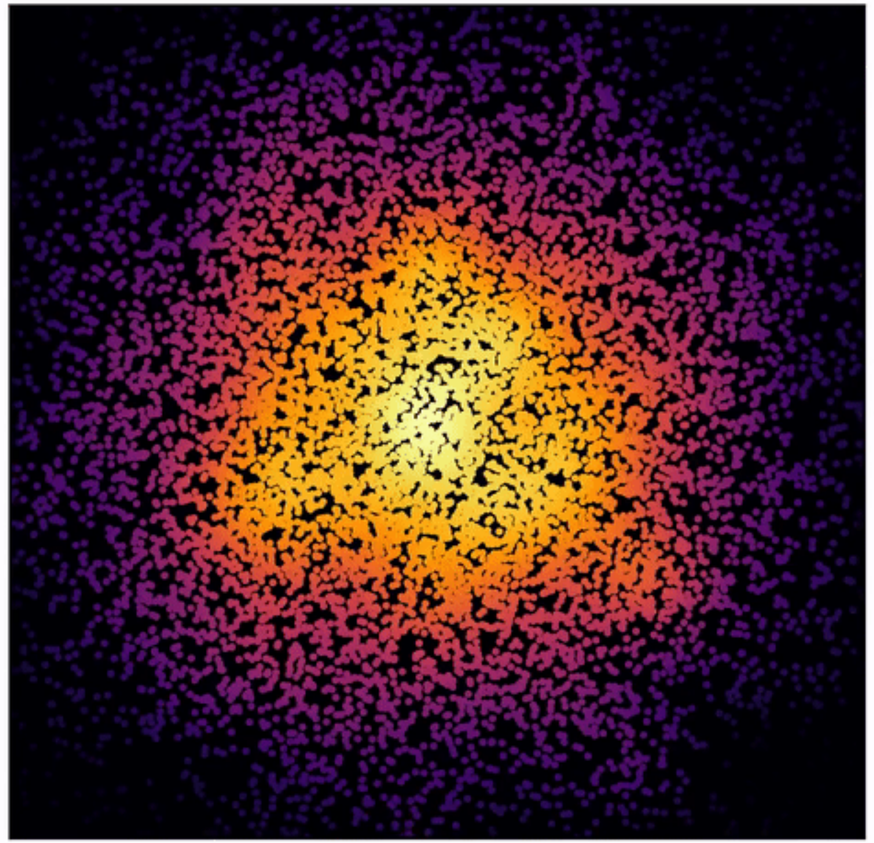
Parameterize the transformation by a neural ODE

ODE with reversible dynamics

$$\frac{\mathrm{d}z}{\mathrm{d}t} = f\left(z(t)\right)$$

Instantaneous change-of-variable formula

$$\frac{\mathrm{d}\log p(z(\mathrm{d}t))}{\mathrm{d}t} = -\operatorname{Tr}\left(\frac{\mathrm{d}f}{\mathrm{d}z(t)}\right)$$



Pro ✅

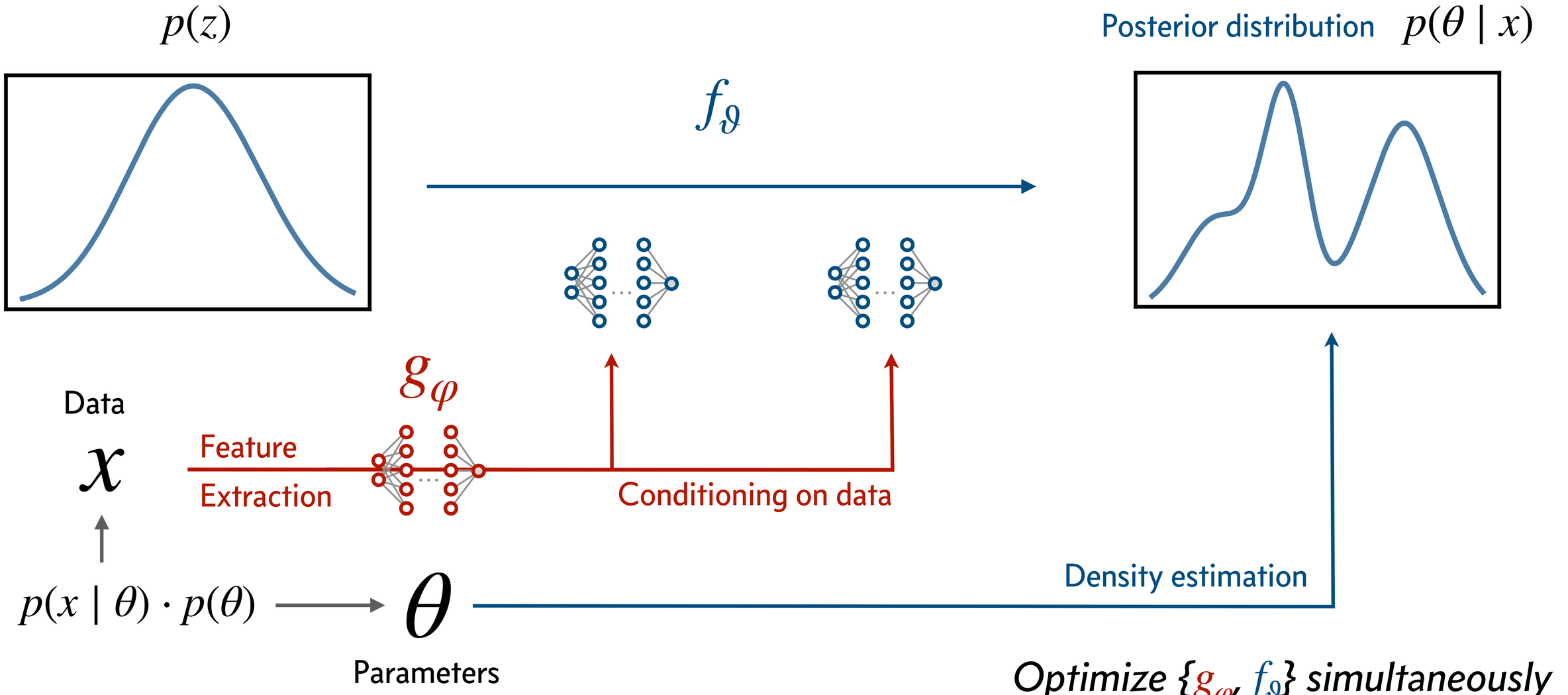
Unrestricted form of transformation $f(z)$!

Cons 😵‍💫

- Need for efficient trace calculation
- Solving an ODE and backpropping through the solution can make for cumbersome training

Flows in *simulation-based inference*

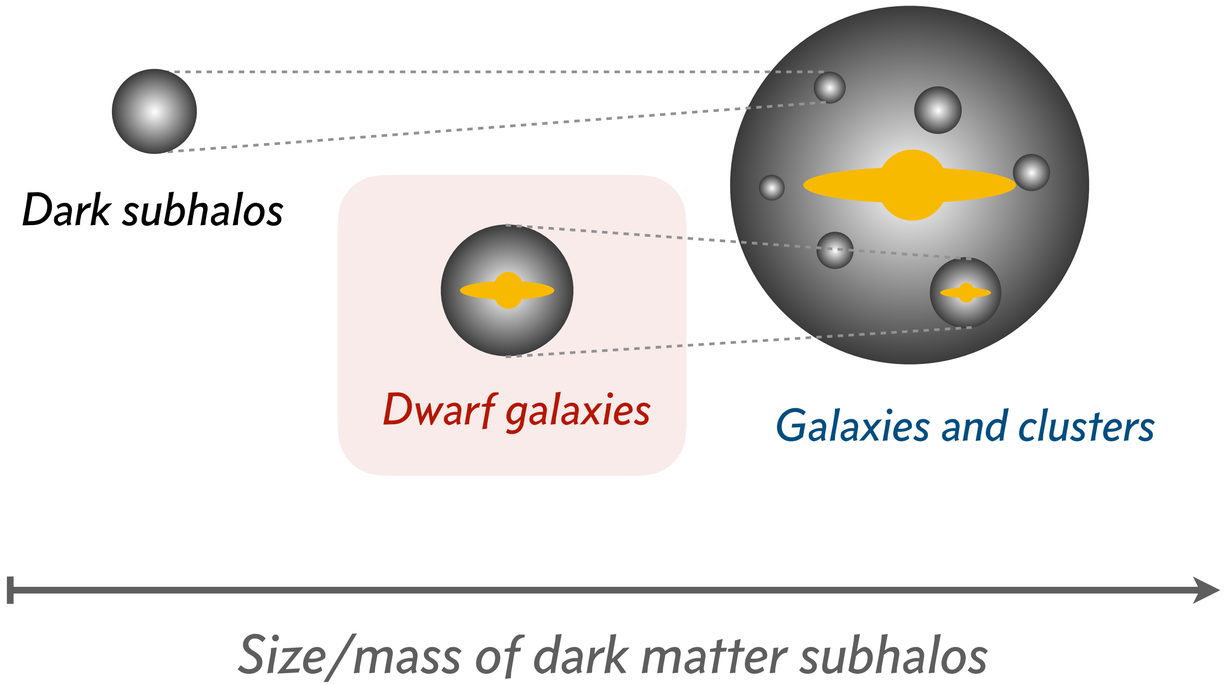
Flows are commonly employed as *conditional posterior density estimators* in simulation-based inference



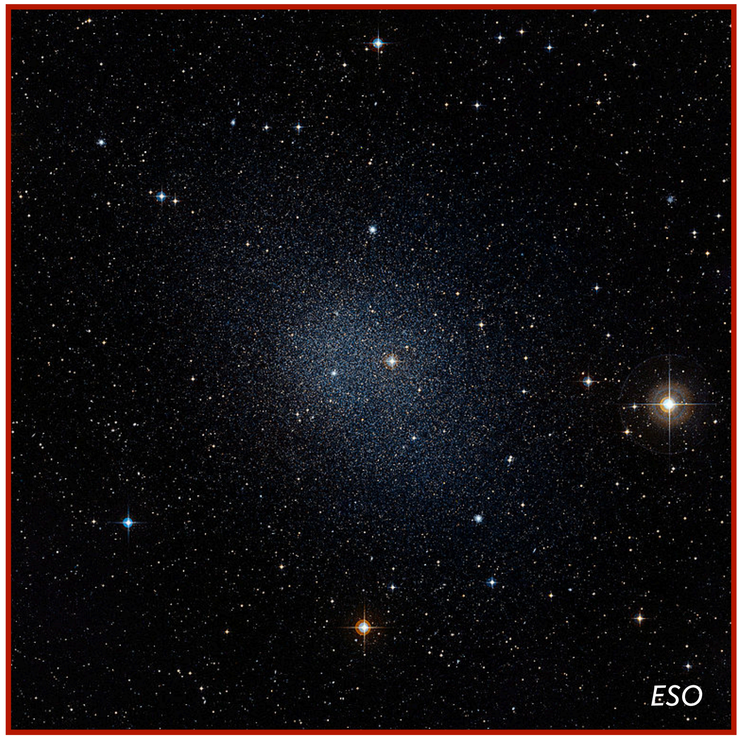
Another application: extracting the dark matter distribution from dwarf galaxies

Dwarf galaxies are intermediate-sized galaxies well-suited for studying dark matter

[Nguyen, SM et al 2022]

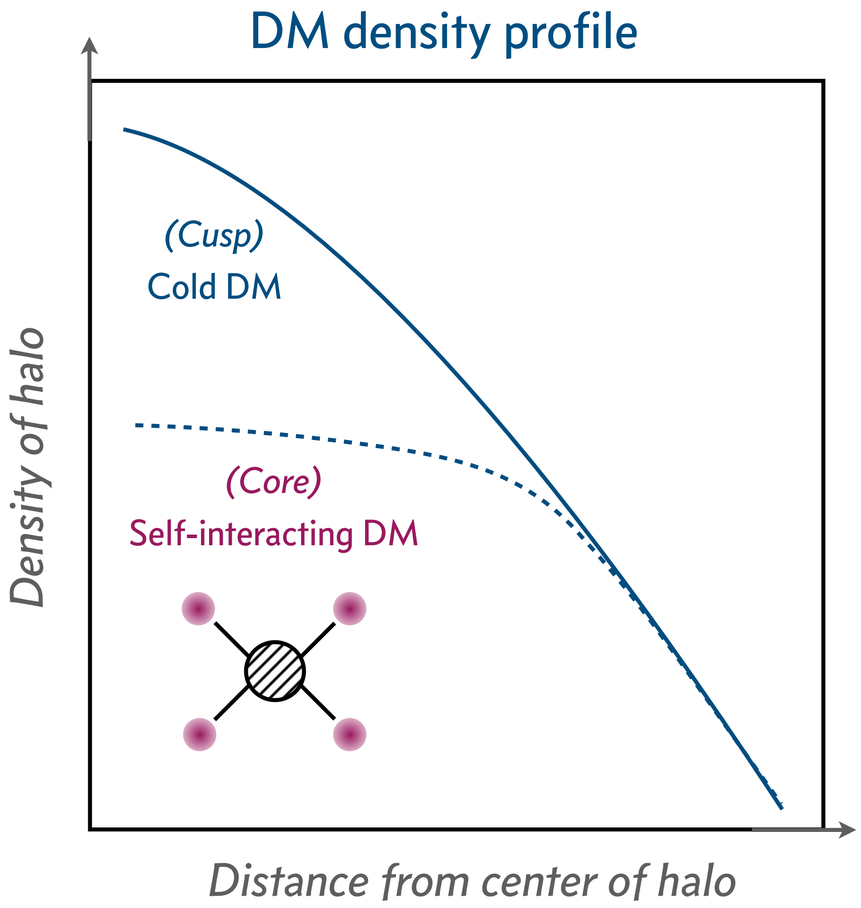


Fornax dwarf galaxy



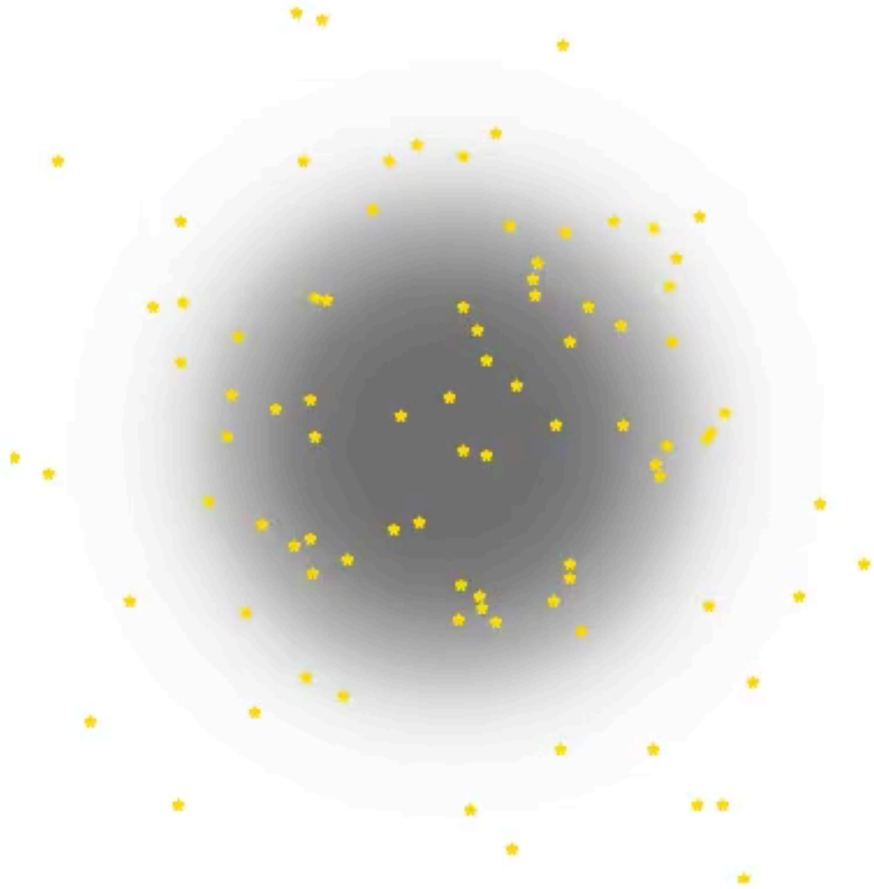
Extracting the dark matter distribution from dwarf galaxies

[Nguyen, SM et al 2022]

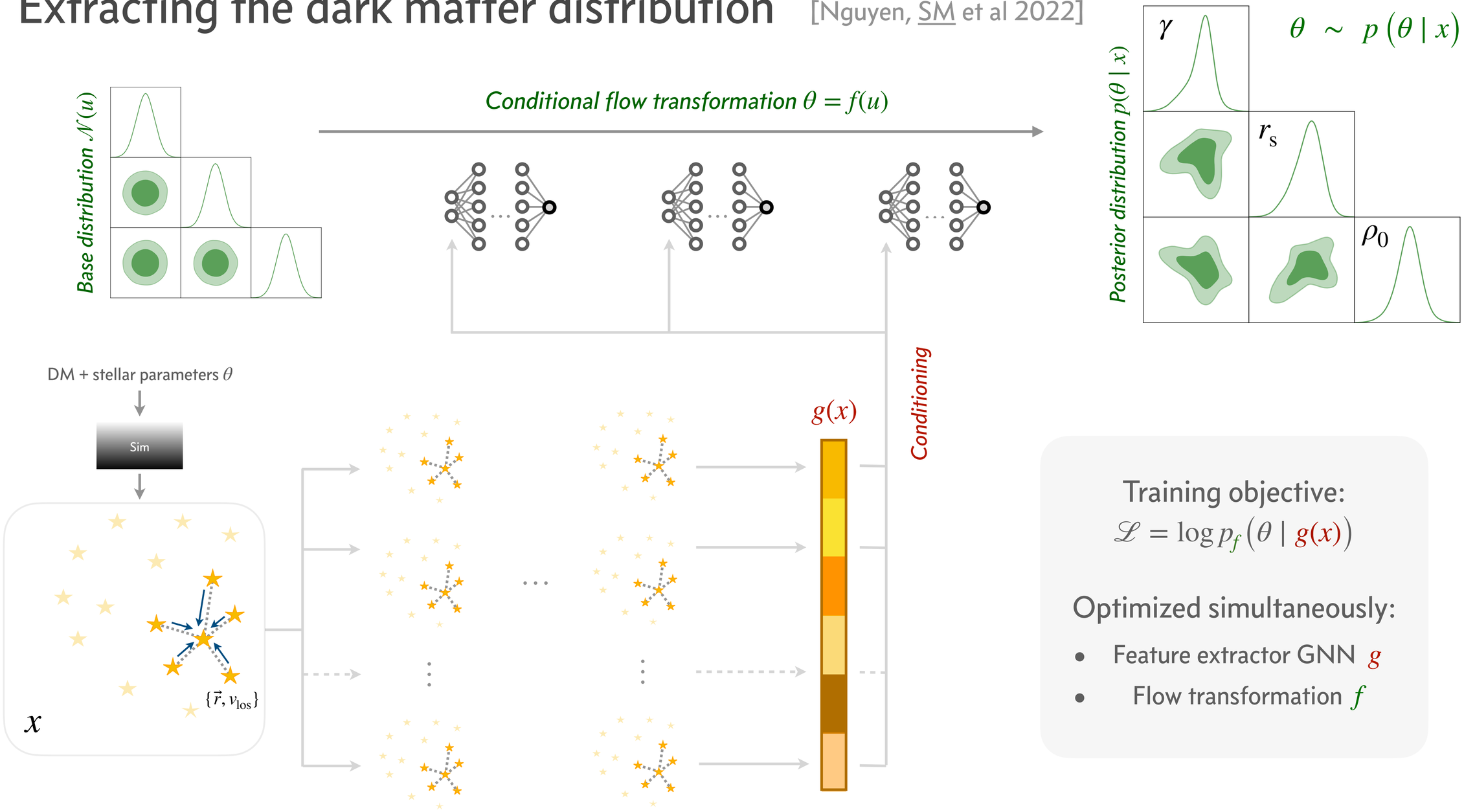


Prediction
(🤗 with a simulator!)

Inference
(💀)



Extracting the dark matter distribution [Nguyen, SM et al 2022]



Training objective:

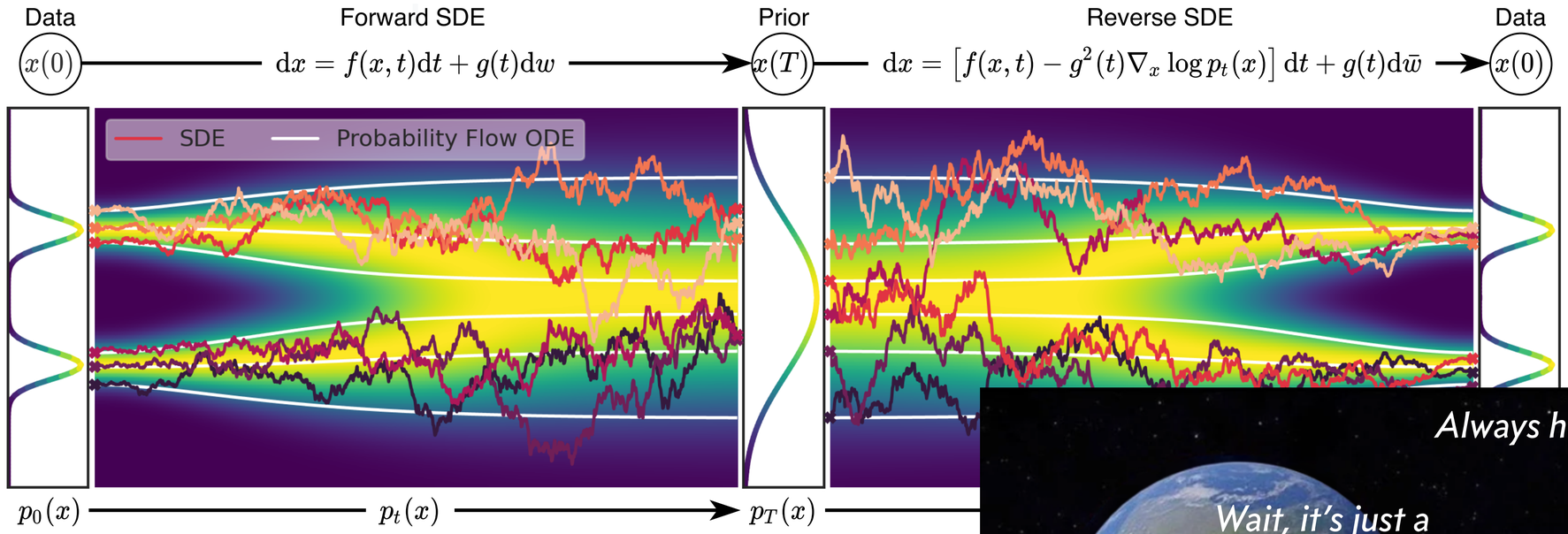
$$\mathscr{L} = \log p_f\left(\theta \mid g(x)\right)$$

Optimized simultaneously:

- Feature extractor GNN g
- Flow transformation f

Back to diffusion: the *probability flow ODE*

For any diffusion process, there exists a corresponding deterministic process whose trajectories share the same marginal probability densities $p(x_t)$ as the SDE [Song et al 2021]



Probability flow ODE

$$\mathrm{d}x = \left[f(x,t) - \frac{1}{2}g^2(t)\nabla_x \log p_t(x)\right]\mathrm{d}t$$

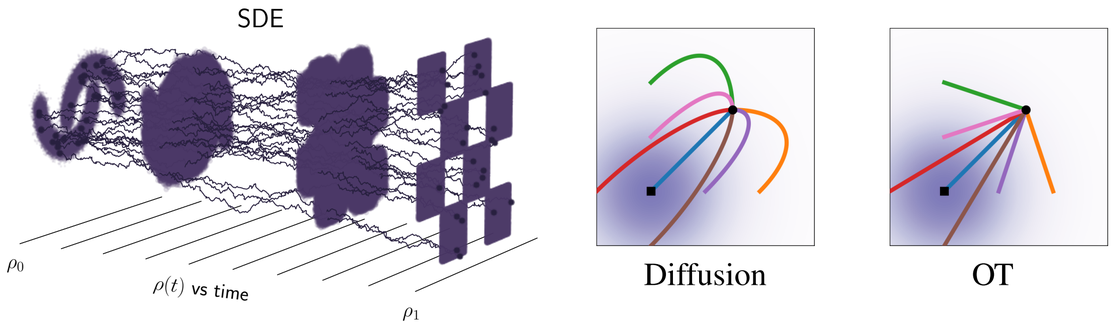
Diffusion mo

"simulation-f

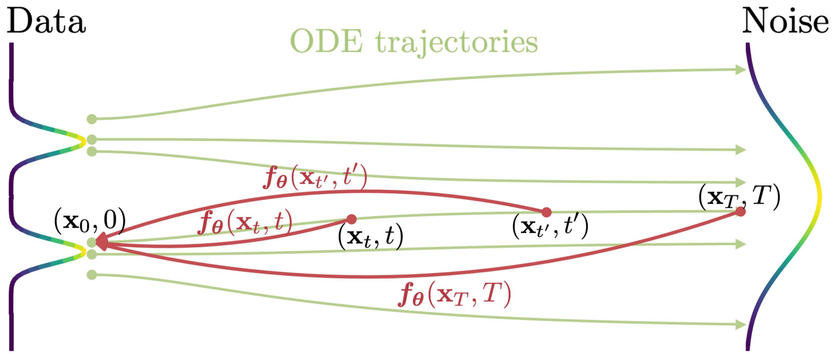
Iterative refinement, interpolants, consistency, …

Empirical success of diffusion models has led to many new formulations and methods of training

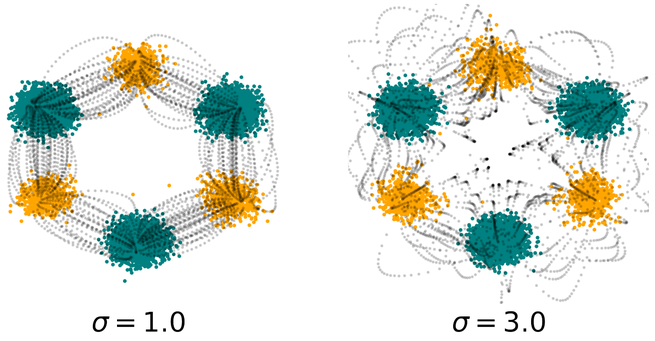
Stochastic interpolants and flow matching



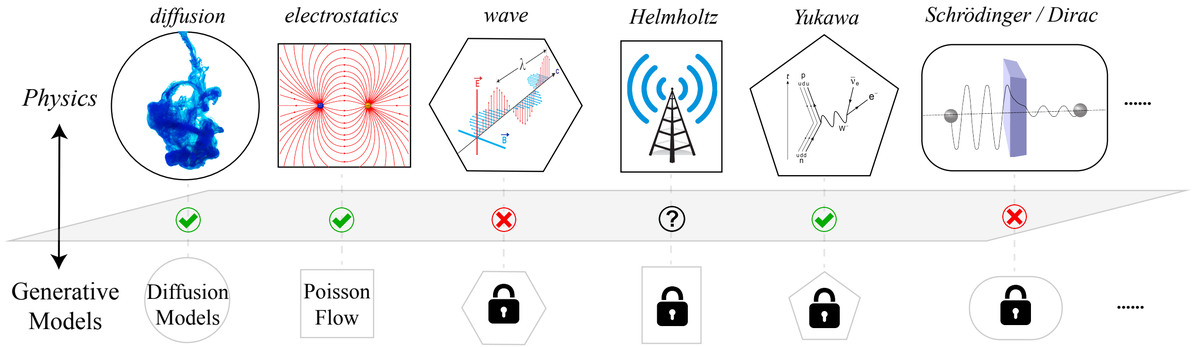
ODE trajectory consistency, Fokker-Planck regularization, …



Diffusion Schrödinger's bridges



Generative models inspired by other physical processes



End.