CaMMLS
Chemistry and Materials
Machine Learning School

UCL

Diffusion Models

Bradley Martin – bradley.martin@ucl.ac.uk

Outline

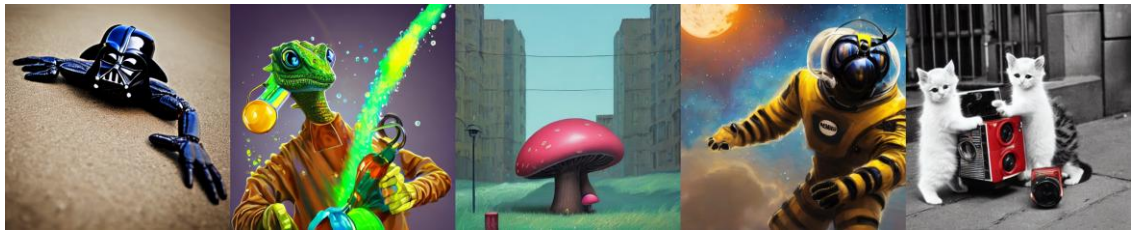
- Diffusion as a physical process: random walks, Langevin dynamics, and links to statistical mechanics/thermodynamics
- From physics to ML: score functions, denoising, and the forward/reverse processes
- Modern diffusion in ML: score matching, denoising diffusion probabilistic models and latent diffusion
- Crystal generation using diffusion models: introducing MatterGen and Chemeleon and how they work

Expected Outcomes

- Build intuition for diffusion models from both physics and ML perspectives
- Know the components that underly diffusion models
- Overview of modern SOTA diffusion model architectures and training procedures
- Learn how to implement and run a simple diffusion model
- How to use pretrained diffusion models (MatterGen/Chemeleon) to generate candidate material structures

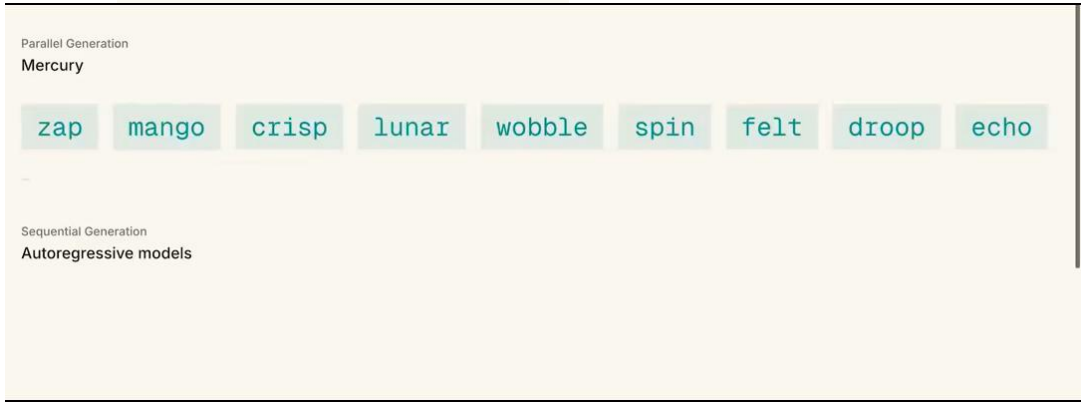
Examples of mainstream diffusion models

Stable Diffusion



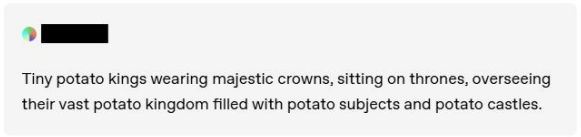
https://github.com/compvis/stable-diffusion

inception Mercury 2



https://chat.inceptionlabs.ai/

DALL·E 3

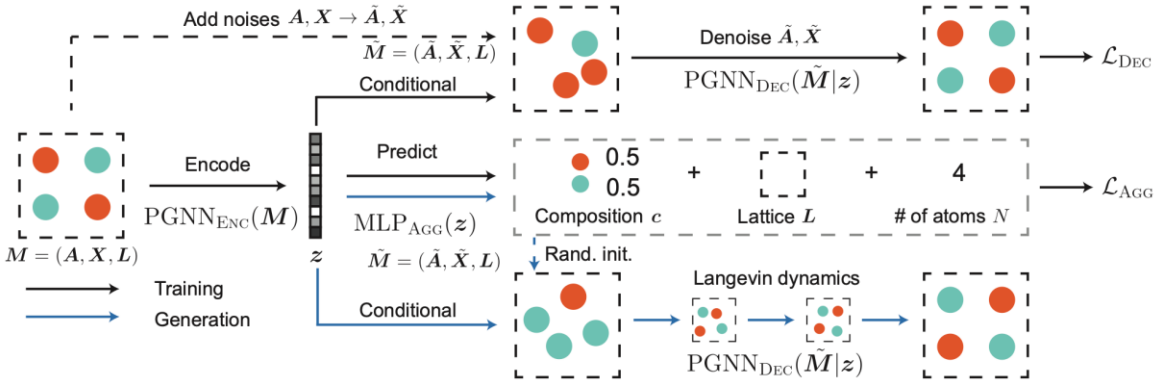


https://openai.com/index/dall-e-3/

Examples for crystal diffusion

Shameless plug: crystallm-pi.com

CDVAE (2021) and DiffCSP (2023)

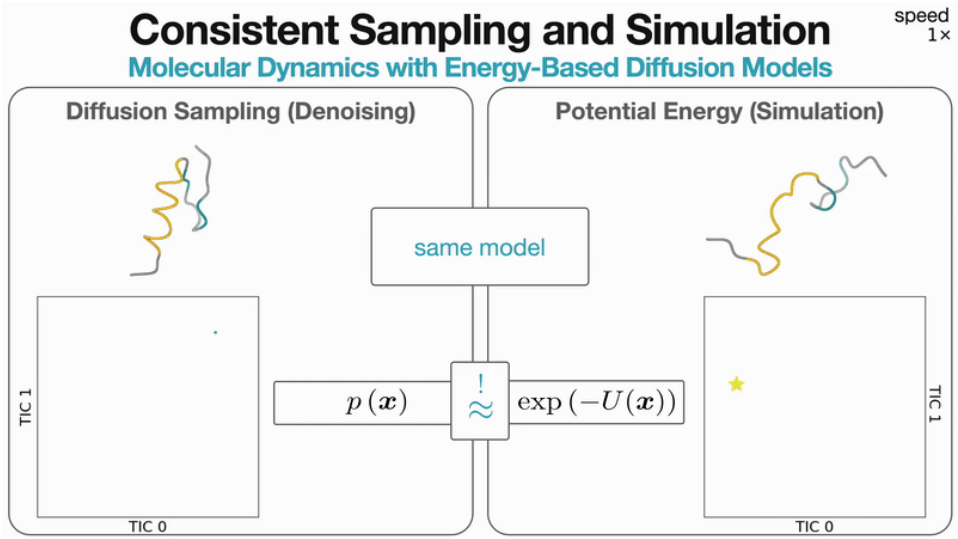


arXiv.2110.06197
arXiv:2309.04475

MatterGen™

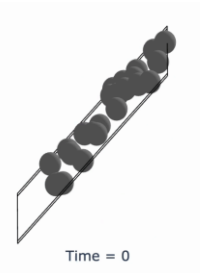
Zeni, C., Pinsler, R., Zügner, D. *et al.* *Nature* **639**, 624–632 (2025).

ScoreMD (2026)

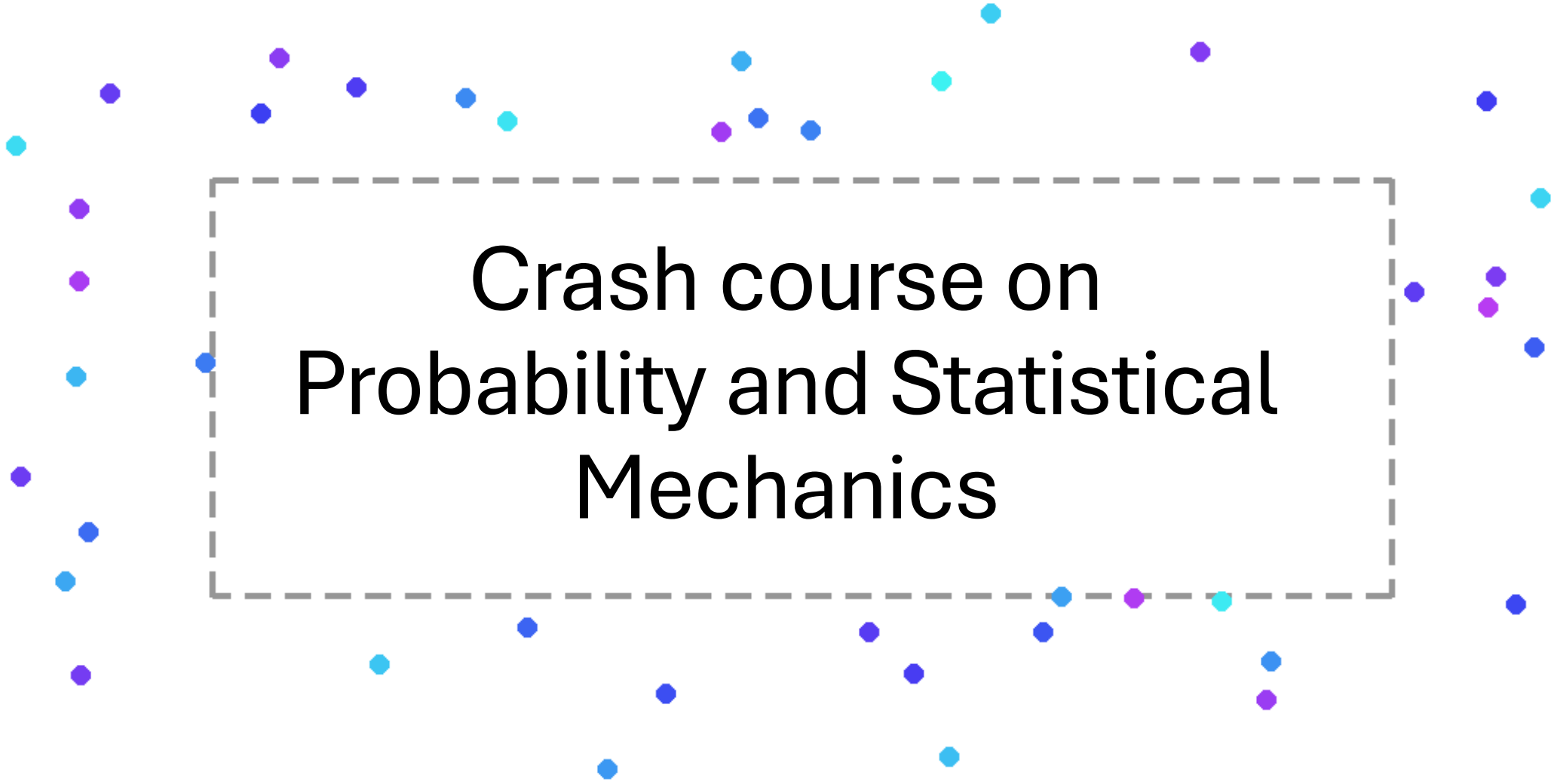


arXiv:2506.17139

Chemeleon



Park, H., Onwuli, A. & Walsh, A. *Nat Commun* **16**, 4379 (2025).



Crash course on Probability and Statistical Mechanics

Why probability and statistical mechanics?

Often, the same mathematics underly methods used in computational chemistry and machine learning.

Examples:

- Molecular dynamics and potential energy surfaces
- Energy-based models and Langevin dynamics
- Equilibrium and non-equilibrium thermodynamics

Likewise, many Machine Learning techniques derive their origins from statistical mechanics and probability theory.

Example: Flips of a coin

Heads:  Tails: 

What's the most likely number of heads or tails?

Flip 6 coins: 

Assuming a fair coin, since there are just two outcomes, we know that there's 50% heads or tails.

N(H=3|T=3) = 10

N(H=4|T=2) = 9

N(H=5|T=1) = 6

N(H=6|T=0) = 1

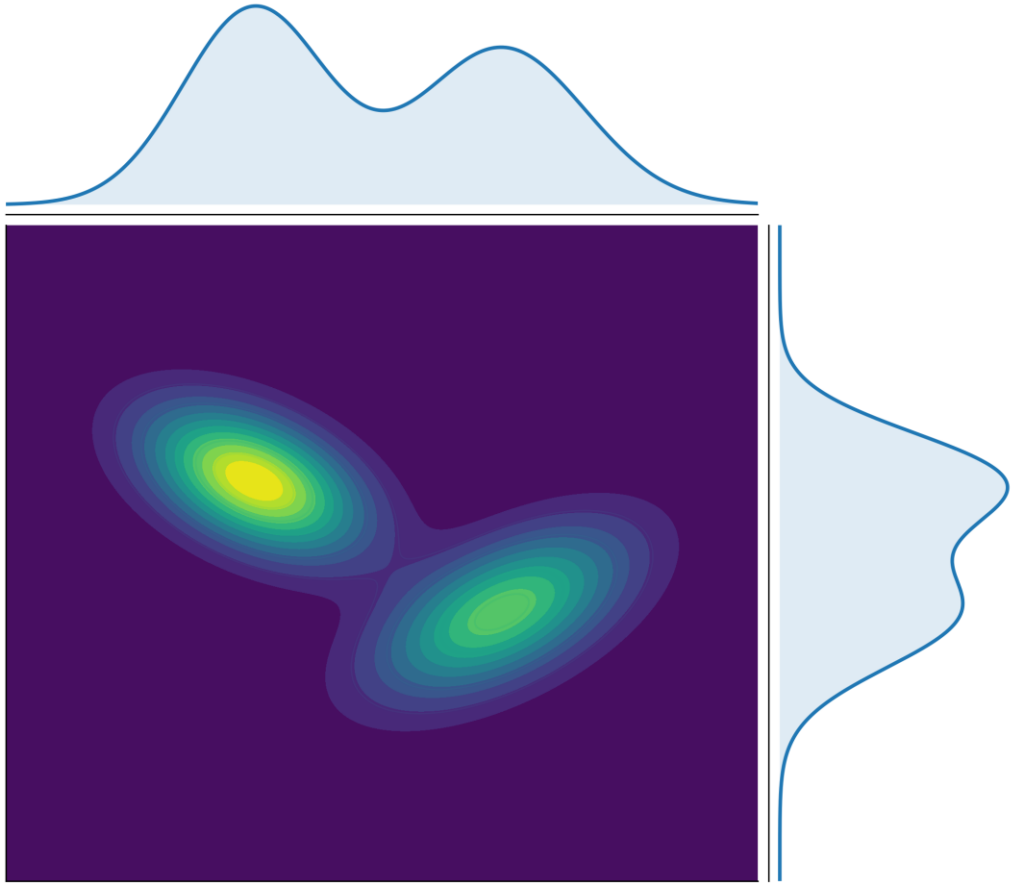
We can determine all possible configurations for N(H|T) and see that the most likely number of heads or tails in a set of 6 flips is 3.

This is just the binomial distribution.

Probability distributions over configurations

Let x state of the system (e.g. heads or tails, atomic coordinates).

A probability density $p(x)$ describes how likely different configurations are.



- Normalisation:

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

- Ensemble average of an observable $A(x)$:

$$\langle A \rangle = \int A(x)p(x)dx$$

The Boltzmann distribution

At thermal equilibrium in the canonical ensemble:

$$p(x) = \frac{1}{Z} e^{-\beta E(x)}$$

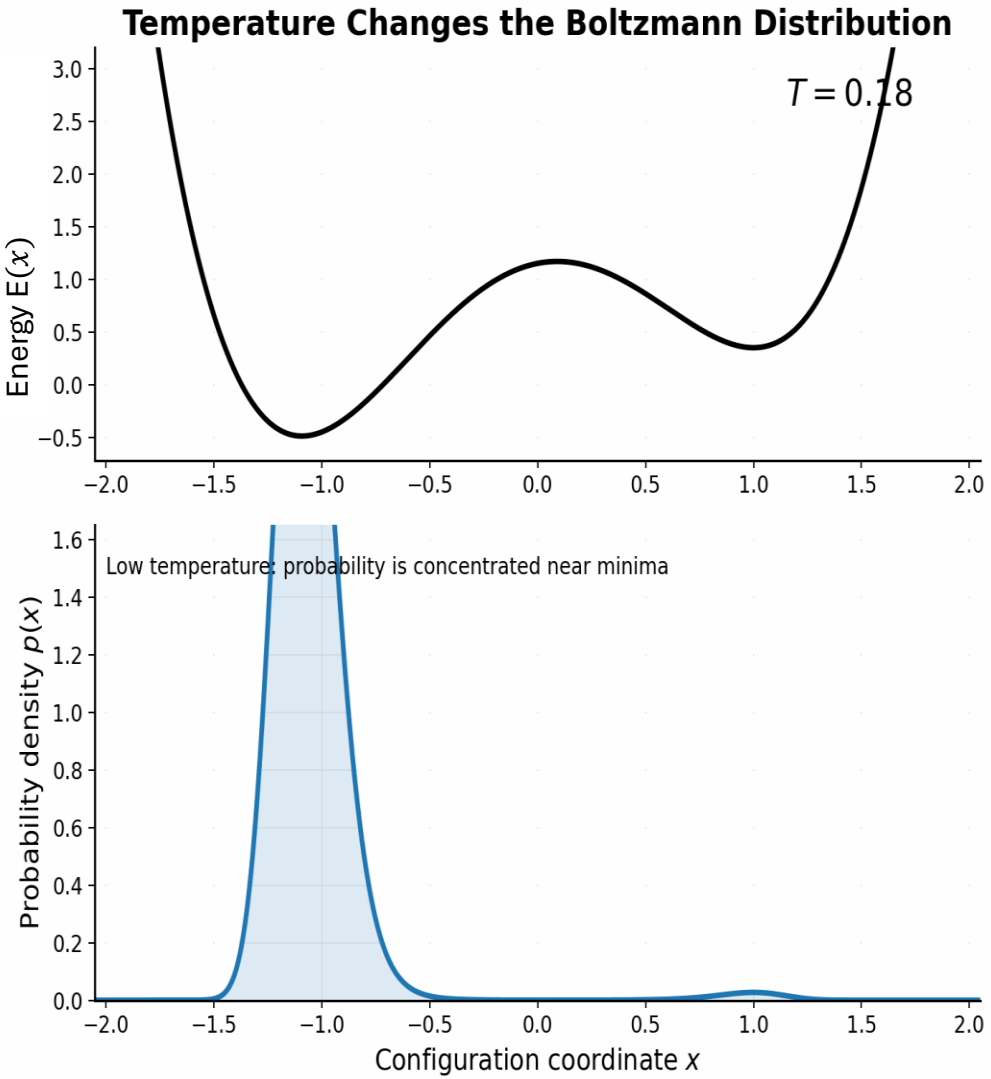
where $\mathrm{E}(x)$ is the energy of state x and $\beta = \frac{1}{k_B T}$.

The normalization constant is called the partition function:

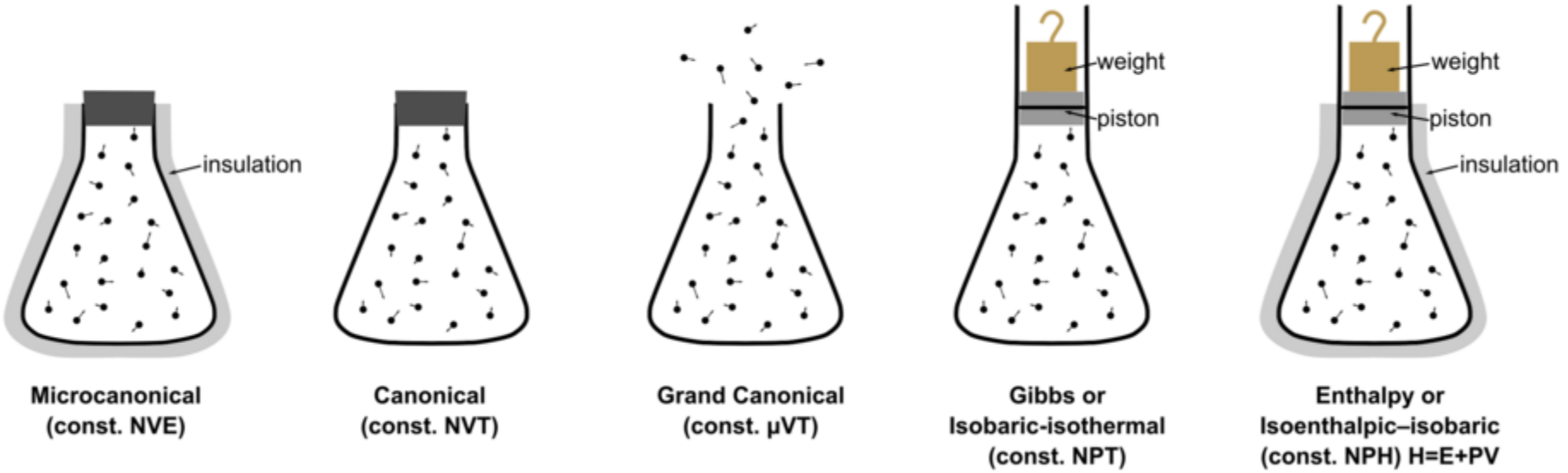
$$Z = \int e^{-\beta E(x)}\, dx$$

Key idea

- Low-energy states are more probable
- Higher temperature broadens the distribution
- Equilibrium is governed by **probability**, not only by minimum energy



Statistical ensembles



	Microcanonical	**Canonical**	**Grand canonical**
Fixed variables	E, N, V	T, N, V	T, μ, V
Microscopic features	Number of microstates	Canonical partition function	Grand partition function
	W	$Z = \sum_k e^{-E_k/k_B T}$	$\mathcal{Z} = \sum_k e^{-(E_k - \mu N_k)/k_B T}$
Macroscopic features	Boltzmann entropy	Helmholtz free energy	Grand potential
	$S = k_B \log W$	$F = -k_B T \log Z$	$\Omega = -k_B T \log \mathcal{Z}$

Free energy and entropy

The partition function contains thermodynamic information

Helmholtz free energy:

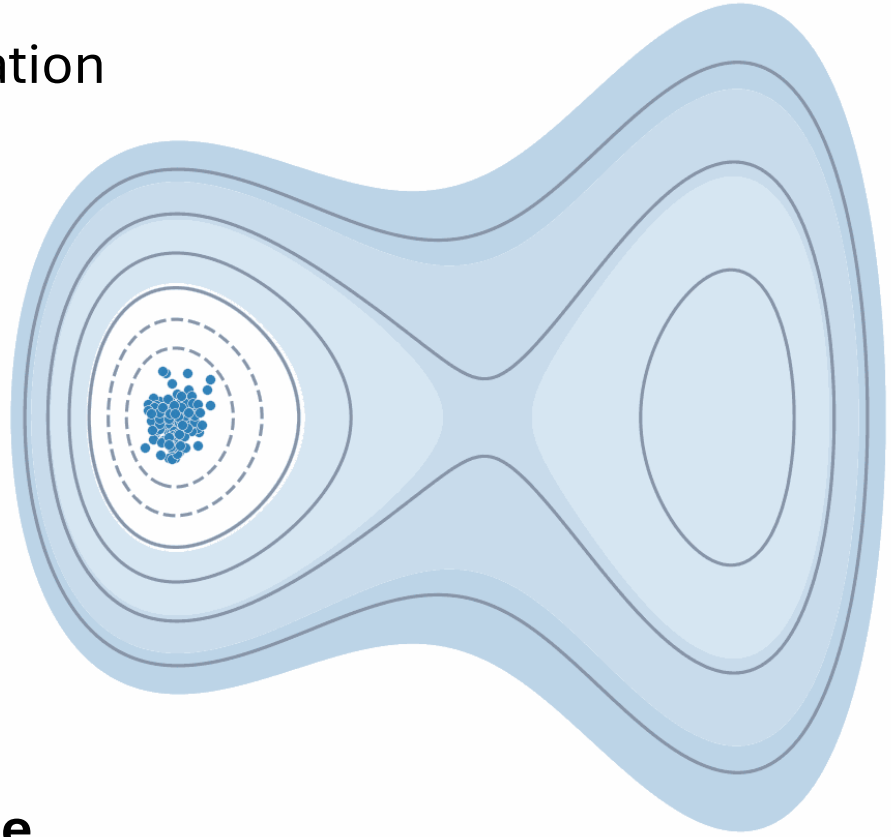
$$F = -k_B T \log Z$$

Systems are governed by a balance between:

- **energy**: favours low-U states
- **entropy**: favours many accessible microstates

This balance is summarized by

$$F = U - TS$$



Key message

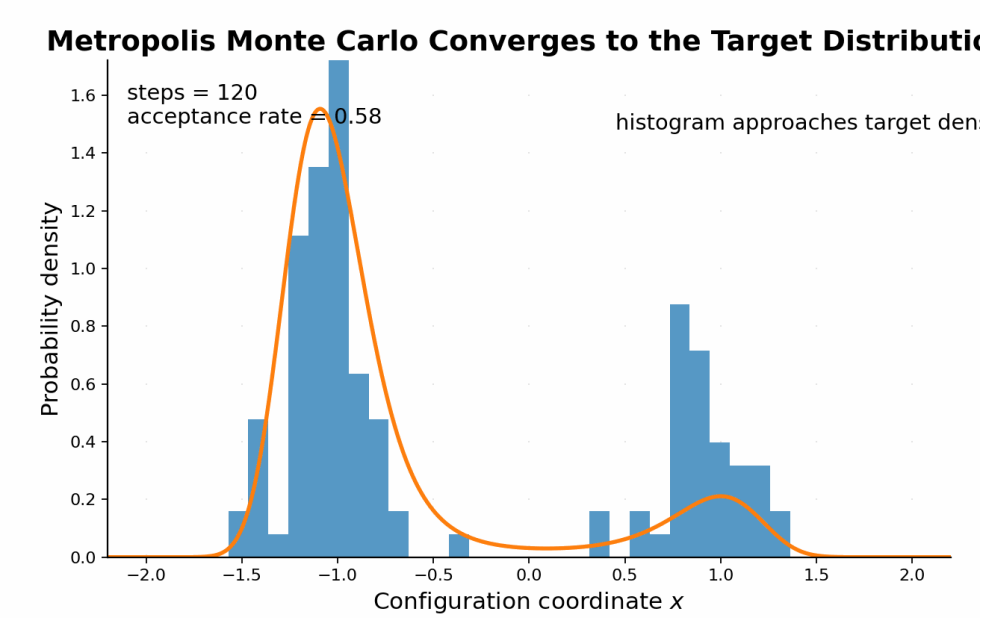
- Physically relevant states are often those with low **free energy**, not simply low potential energy

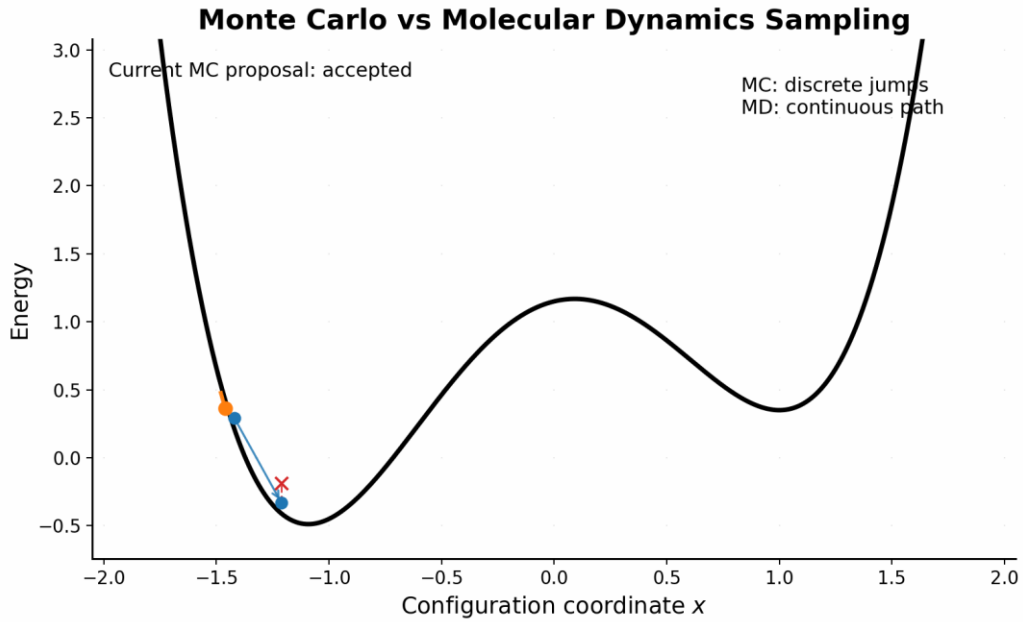
Why sampling is necessary

- Realistic systems live in very high-dimensional configuration spaces
- Exact integration is usually impossible
- So the task becomes: generate samples from the target distribution
- Then approximate averages via sample means

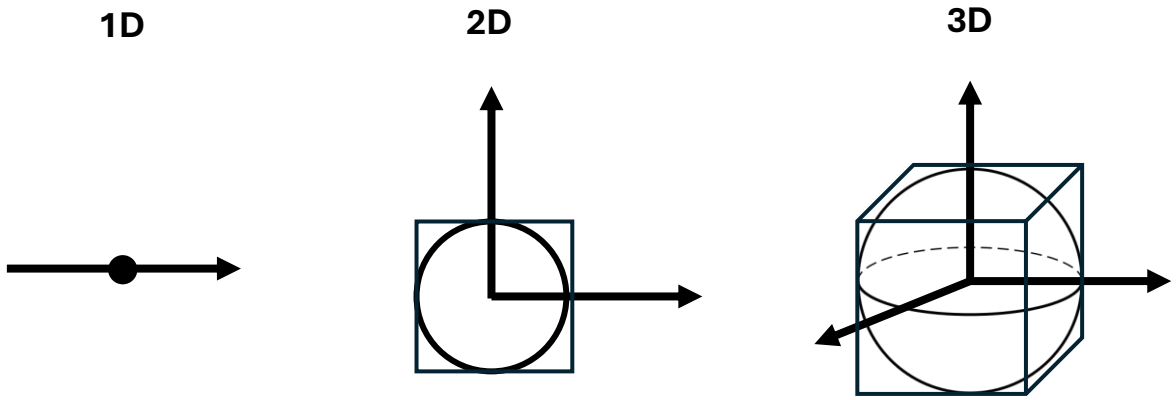
Examples of target distributions

- canonical ensemble
- isothermal-isobaric ensemble
- constrained ensembles

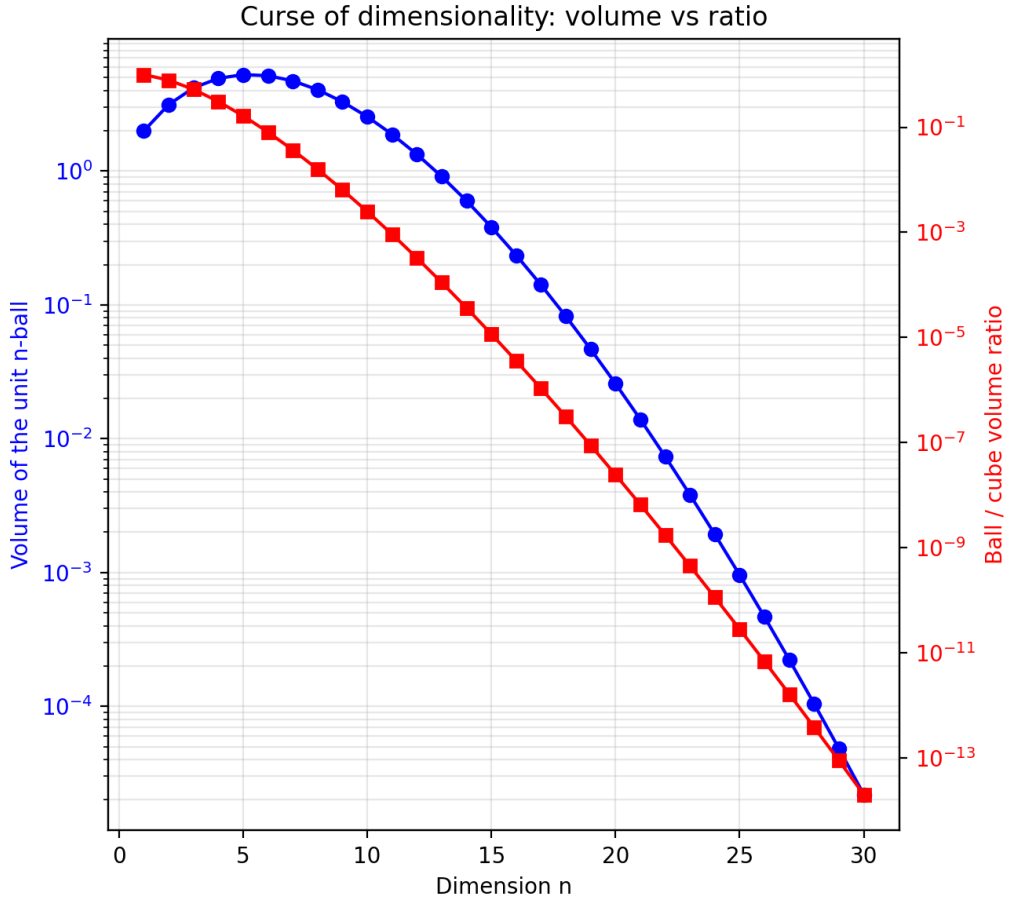




Aside: Curse of dimensionality



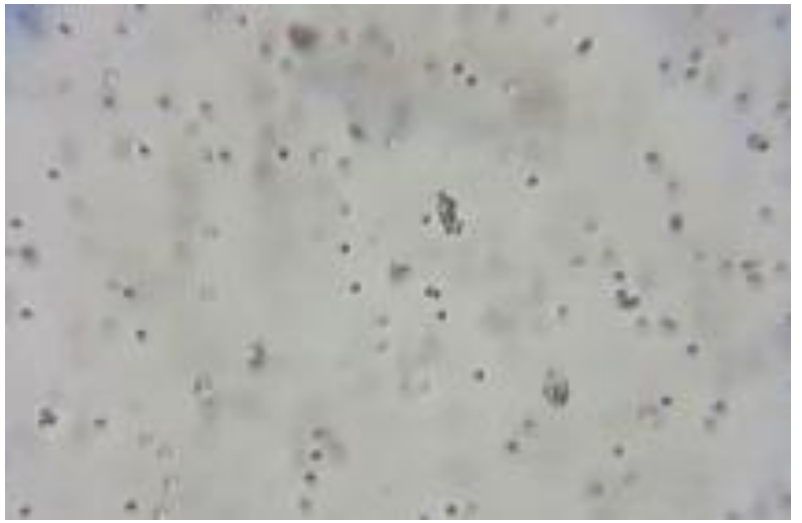
$$\frac{V_{hypercube}}{V_{hypersphere}} = \frac{\pi^{d/2}}{d2^{d-1}\Gamma(d/2)} \to 0 \text{ as } d \to \infty$$

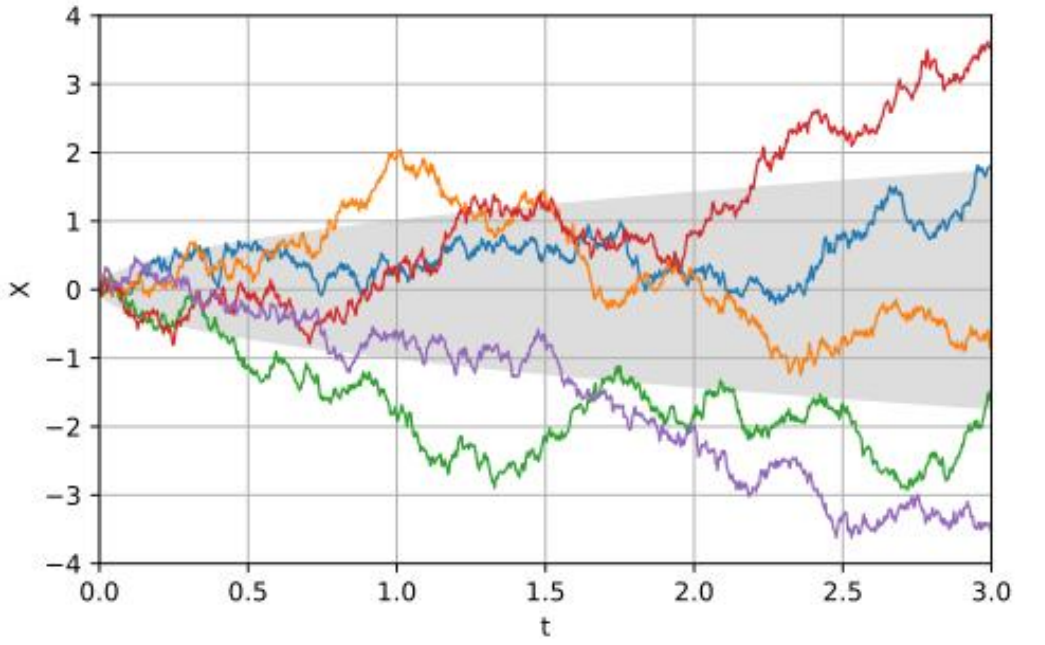


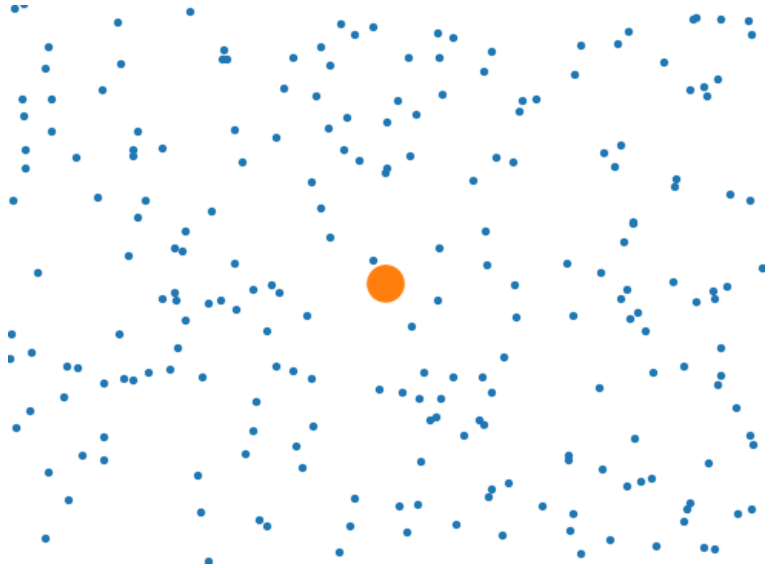
Brownian Motion

- The probability density function of a state (e.g. the position of a pollen grain in water) follows a normal distribution:

$$p(x, t; \sigma) = \mathcal{N}(0, t) = \exp(x^2/2t\sigma^2)/\sqrt{2\pi t\sigma^2}$$







Langevin Dynamics & Stochastic Differential Equations

- Models the dynamics of a molecular systems in e.g. air and solvents
- Jostling of particles causes friction and occasional high velocities.

Ordinary differential equation:

$$M\ddot{\boldsymbol{X}} = \underbrace{-\nabla U(\boldsymbol{X})}_{\textbf{Force}} \underbrace{- \gamma M \dot{\boldsymbol{X}}}_{\textbf{Friction}} + \underbrace{\sqrt{2\mathrm{M}\gamma k_B T}\, \boldsymbol{R}(t)}_{\textbf{Thermal}}$$

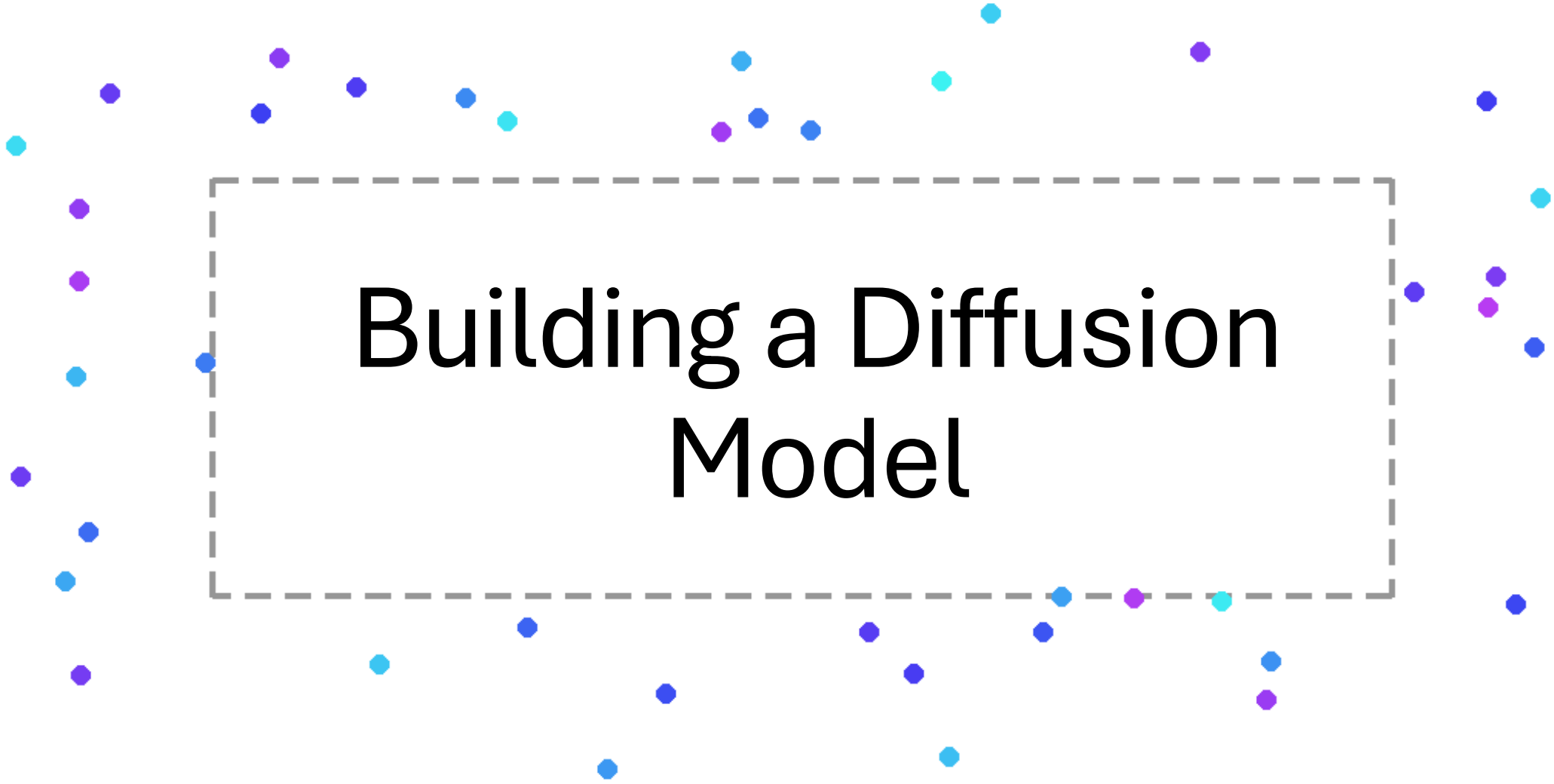
Stochastic differential equation:

$$\boldsymbol{d\dot{X}} = -\nabla U(\boldsymbol{X}) dt/M - \gamma d\boldsymbol{X} + \sqrt{2\gamma k_B T/M}\, d\boldsymbol{W}(t)$$

- Brownian dynamics ≡ *Overdamped* Langevin: $d\dot{\boldsymbol{X}} = 0$ i.e. **no acceleration**.

Takeaway

- Probability theory describes distributions over configurations
- Statistical mechanics connects those distributions to energy and temperature
- Free energy balances energetic and entropic effects
- Exact averages are usually impossible to compute directly
- Monte Carlo and molecular dynamics are two central strategies for sampling high-dimensional ensembles
- ML interatomic potentials (MACE, NequIP) now extend MD to millions of atoms at near-DFT accuracy
- Generative models (diffusion, flow matching, normalizing flows) are emerging as complementary sampling tools that learn the target distribution directly



Building a Diffusion Model

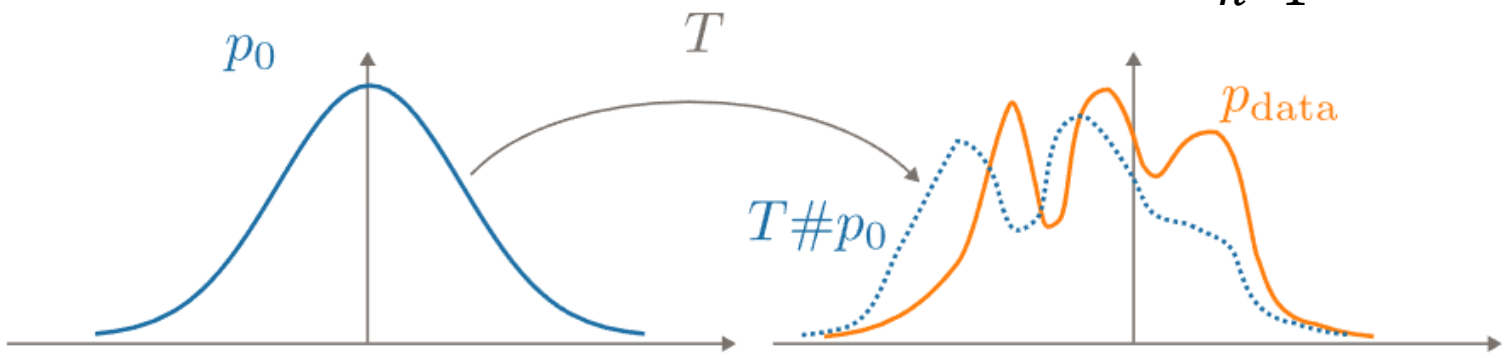
Generative Modelling – “parametric density estimation”

Data samples: $\{x^{(n)}\}_{n=1}^{N} \sim q_{data}(x)$ drawn from some underlying “true but unknown” data distribution.

Define a suitable surrogate model $p_\theta(x)$ with unknown parameters θ.

We want to minimize the difference between the surrogate and true distribution:

$$\theta^* = arg\max_\theta \mathbb{E}_{x\sim q_{data}(x)}[\log p_\theta(x)] \approx arg\max_\theta \frac{1}{N}\sum_{n=1}^{N} \log p_\theta\left(x^{(n)}\right)$$



The Normalization Problem

Our surrogate "probability density" $p_\theta(x)$ must be normalized (integrates to one).

The normalization constant stems from the axiom of probability:

$$p_\theta(x) = \frac{\tilde{p}_\theta(x)}{\int_x \tilde{p}_\theta(x)}$$

For all but a select few (overly simple) distributions, this normalization is intractable.

All generative models are essentially different ways of sidestepping the normalization constant and/or easing sampling.

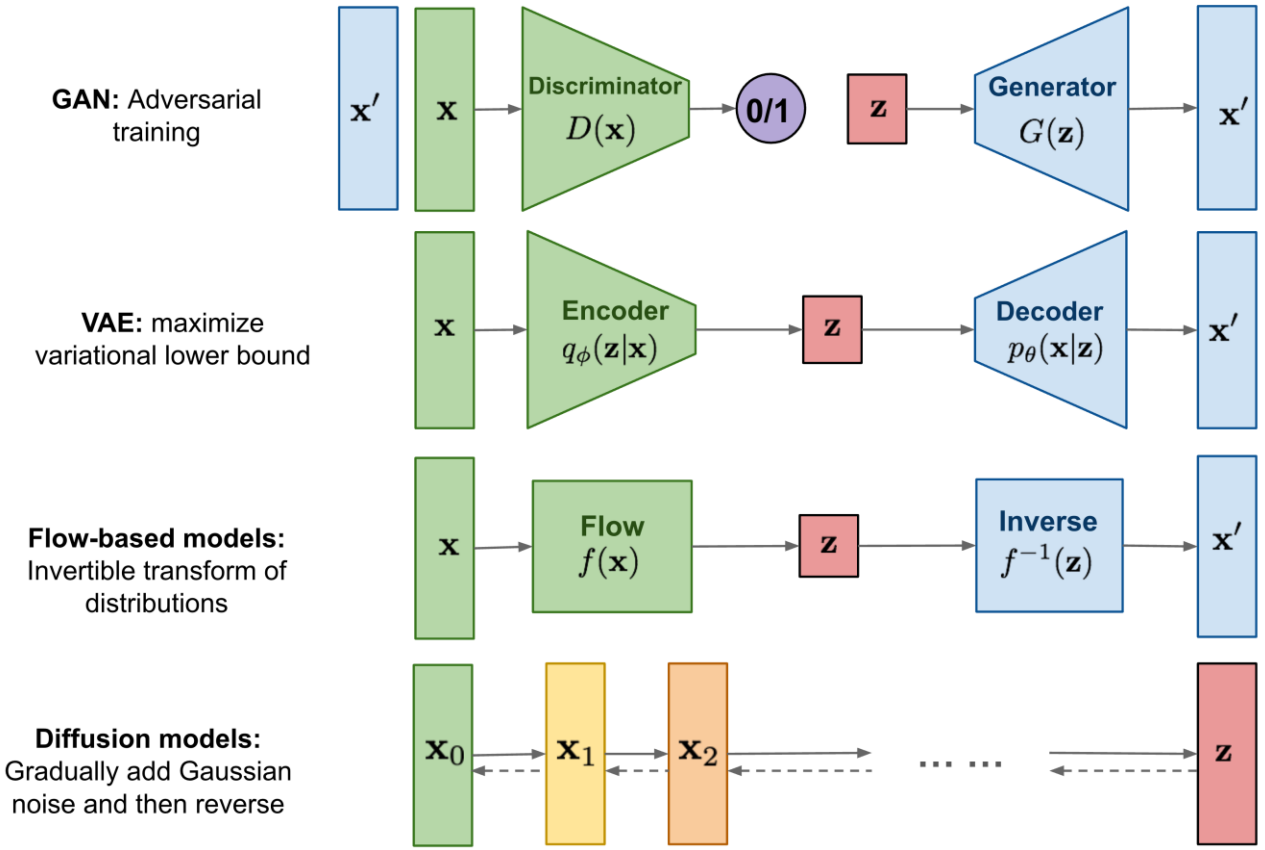
Existing Generative Models

A common underlying principle – *transforming simple probability distributions:*

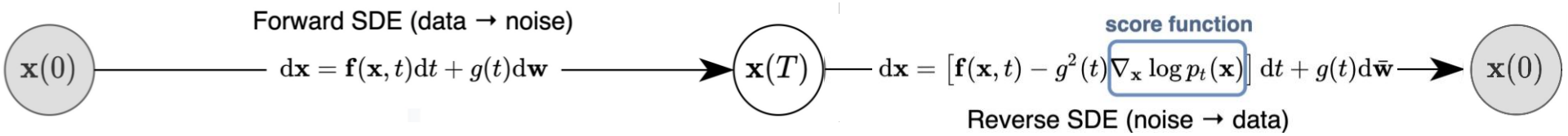
$$x = f_\theta(z), \text{where } z \sim \mathcal{N}(0, I)$$

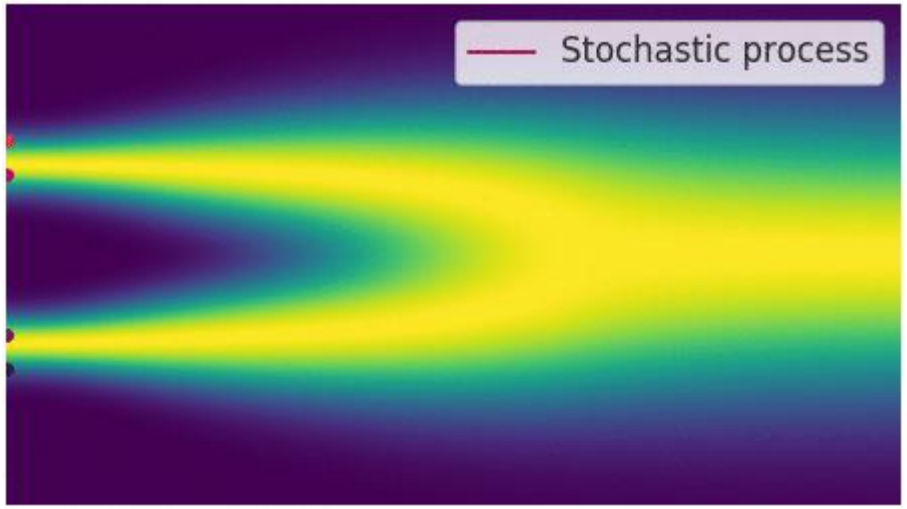
f_θ is a *functional* transformation (typically a neural network), parameterised by θ.

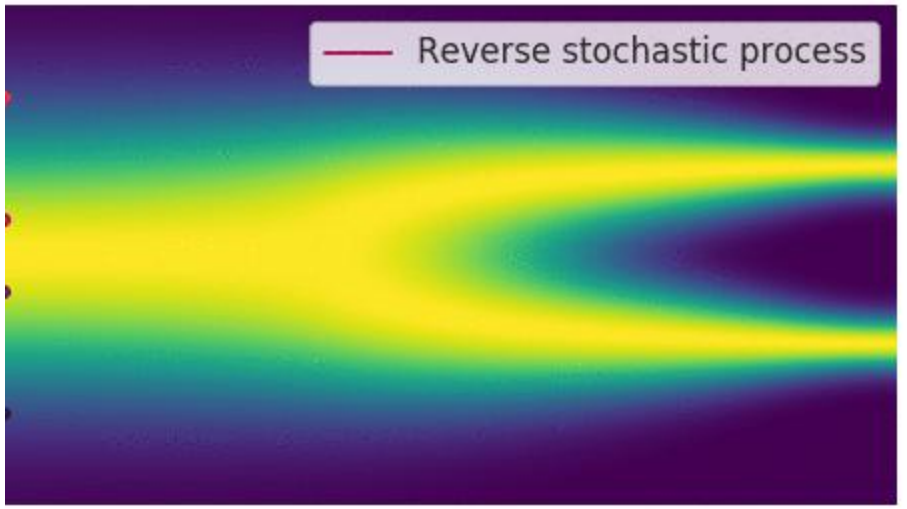
Different models mainly differ by *how they are trained*.

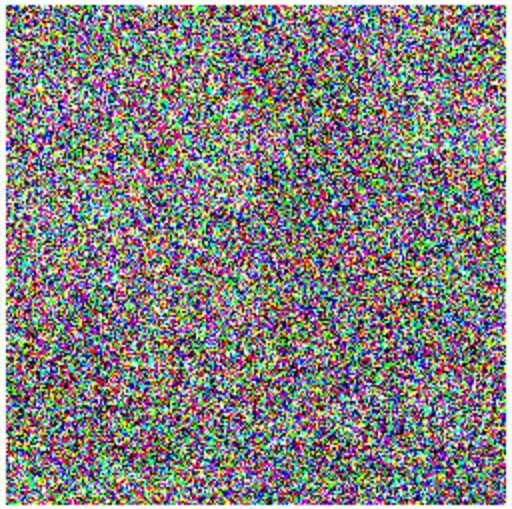


https://lilianweng.github.io/posts/2021-07-11-diffusion-models/


Forward SDE (data → noise)
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$
$\mathbf{x}(T)$
score function
$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\,\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$
$\mathbf{x}(0)$
Reverse SDE (noise → data)


Stochastic process


Reverse stochastic process



The “Score”

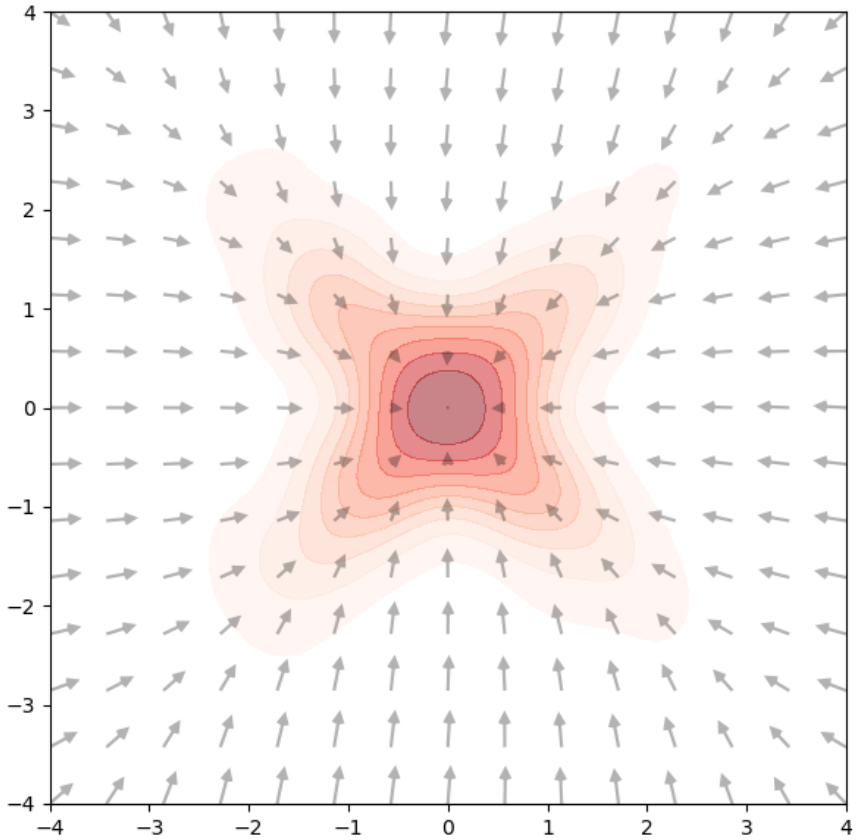
https://iclr-blogposts.github.io/2024/blog/diffusion-theory-from-scratch/

In diffusion models, we try to model the *true score* of the data distribution:

$$\nabla_x \log q_{data}(x) \triangleq s(x)$$

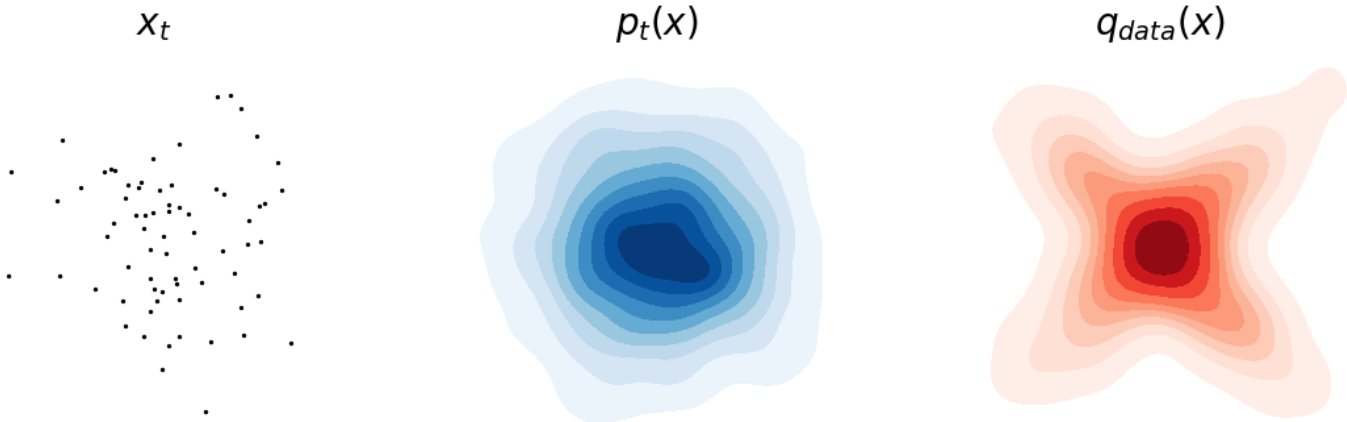
This is generally unknown but refers to the *direction of steepest increase* in log-likelihood in data-space.

Importantly, it does not depend on the normalization!



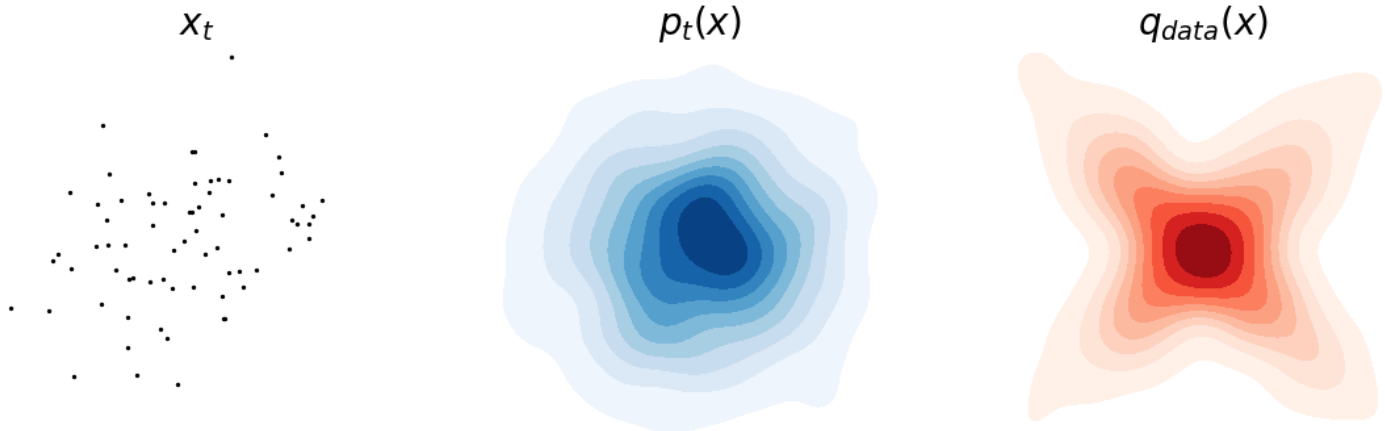
$$x' = x + \delta \cdot \nabla_x \log q_{data}(x)|_{x=x}$$

The *Reverse* Process: Sampling



Given the score, direct gradient ascent is **greedy** - converges to the *modes*.

$$dx = \nabla_x \log q_{data}(x)dt$$



By adding noise, we prevent mode collapse and can better sample the distribution.

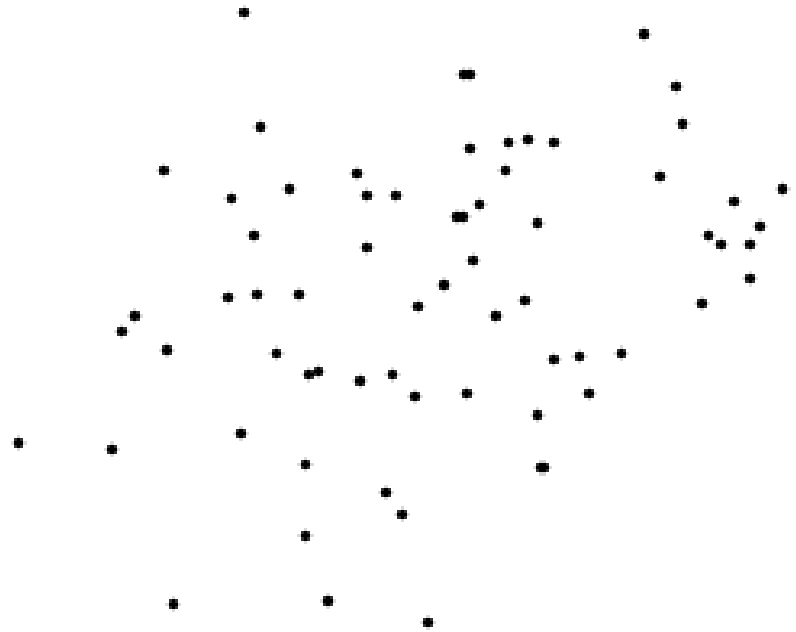
$$dx = \nabla_x \log q_{data}(x)dt + \sqrt{2}dB_t$$

Langevin Dynamics & Brownian Motion

In **molecular dynamics** the particles move according to a *potential energy field* $U(x)$:

$$dx = -\nabla_x U(x)dt + \sqrt{2}dB_t$$

dB_t is "Brownian Motion" and adds normally-distributed noise.



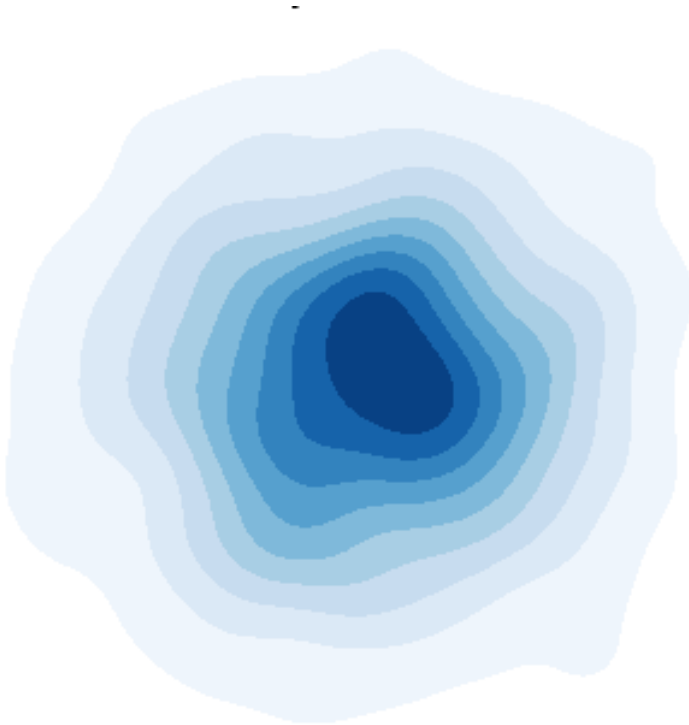
The Langevin equation allows exploration to visit states with probability:

$$q_{data}(x) = e^{-U(x)}/Z \implies \log q_{data}(x) = -U(x) + Z \implies \nabla_x \log q_{data}(x) = -\nabla_x U(x)$$

Fokker-Planck Equation

The stochastic process $dx = \nabla_x \log q_{data}(x)dt + \sigma_t(x)dB_t$ induces a time-varying distribution:

$$\frac{\partial}{\partial t}p_t(x) = -\frac{\partial}{\partial x}[p_t(x)\mu_t(x)] + \frac{1}{2}\frac{\partial^2}{\partial x^2}[p_t(x)\sigma_t^2(x)]$$



This ODE provably converges $p_t(x)$ to the data distribution in the infinite time limit $p_\infty(x) = q_{data}(x)$.

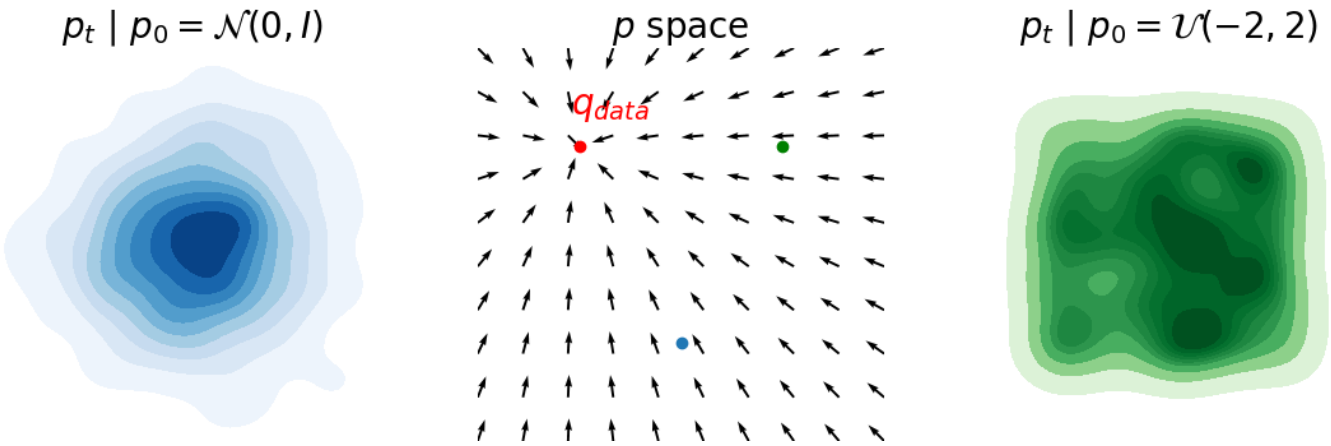
The distribution “stabilizes” like how molecular dynamics approaches thermal equilibrium.

Probability “Flows”

Transforming our data-points induces a flow along “paths” in the *space of probability distributions*.

Any initial distribution p_0 will “flow” and converge on the data distribution $p_\infty = q_{data}$.

We usually choose a simple initial distribution – Isotropic Gaussian $\mathcal{N}(0, I)$.

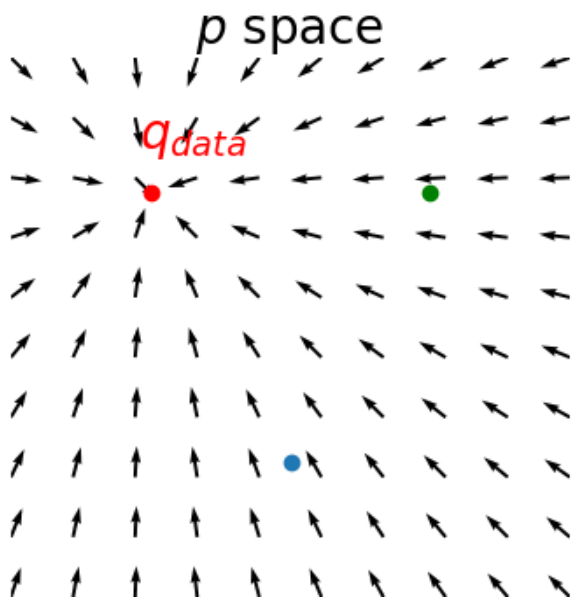


Estimating the Score

Estimating the score $\nabla_x \log q_{data}(x)|_{x=x_t}$ with a NN $s_\theta(x = x_t)$ is **very hard**.

One NN is not enough, so two options:

1. Make the NN *expressive enough*
2. **Only learn where it is needed.**



Approach: fix the probability path and learn the score *only on that path*. We specialize the neural network $s_\theta(x_t, t)$ over $t \in [0, \infty)$.

We need to get samples x_t for all t – the *forward process*.

The *Forward* Process

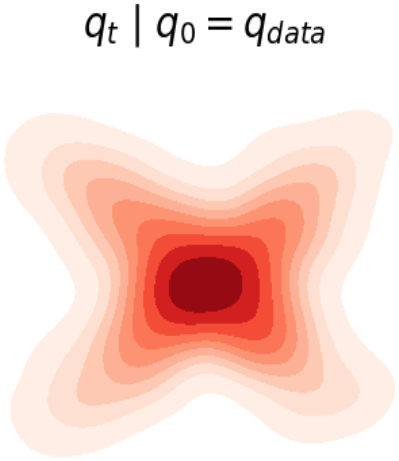
We run a simulation to go from $q_{data}(x)$ at $t = 0$ to $q_\infty = \mathcal{N}(0, I)$ at $t = \infty$:

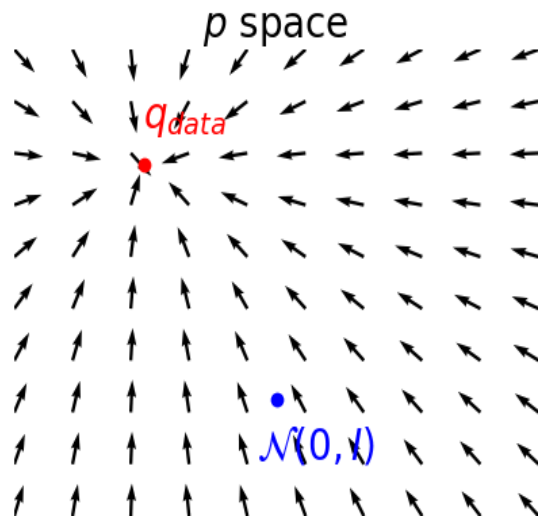
$$\begin{aligned} dx &= \nabla_x \log \mathcal{N}(0, I)\, dt + \sqrt{2} dB_t \\ &= -x dt + \sqrt{2dt} z \end{aligned}$$

q_∞ is known in closed form, so we know the score $-x$.

The discretised SDE:

$$x_{t+dt} = (1 - dt)x_t + \sqrt{2dt} z$$





While the reverse process is sequential due to the score, here we can sample ***any*** $x_t \sim q_t$ because $q_t(x_t|x_0)$ is Gaussian!

Finite time and Scheduling

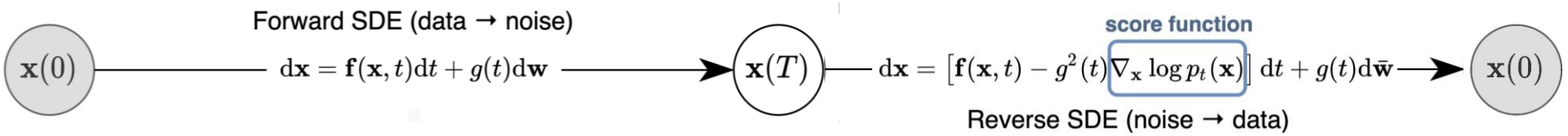
Langevin results in *infinite* time to reach our target state.

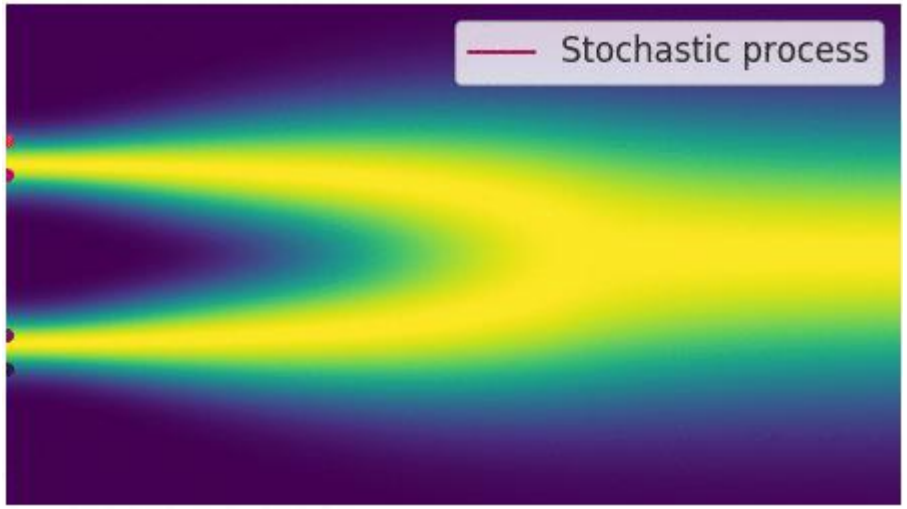
Redefine "time" to be *finite*: $t' = \mathcal{T}(t) = 1 - e^{-t} \in [0,1]$.

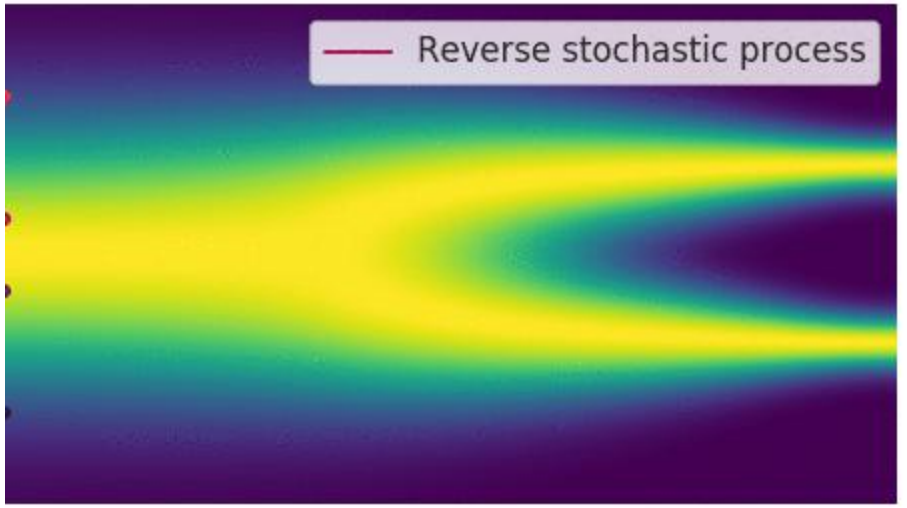
The forward equation: $x_{t'+dt'} = (1 - e^t dt')x_t + \sqrt{2e^t dt'}\, z$ where $t' = 0 \to 1$.

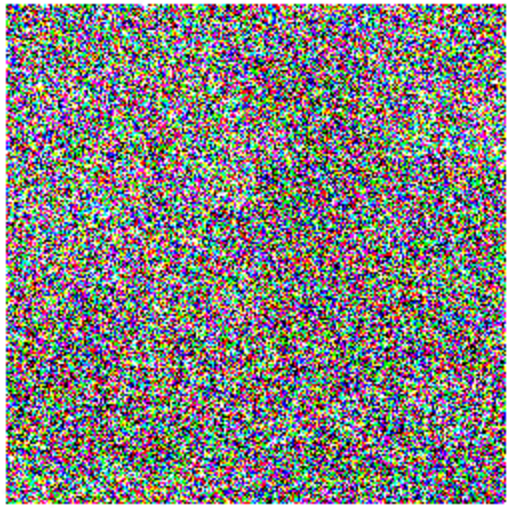
$dt = \mathcal{T}'(t)^{-1} dt' = e^t dt'$ is called a *"schedule"*. This is a design choice.

Different choices include: linear, cosine, exponential etc. These do not change the path taken in probability space, just the rate at which we move between states.


Forward SDE (data → noise)
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$
$\mathbf{x}(T)$
score function
$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\,\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$
$\mathbf{x}(0)$
Reverse SDE (noise → data)

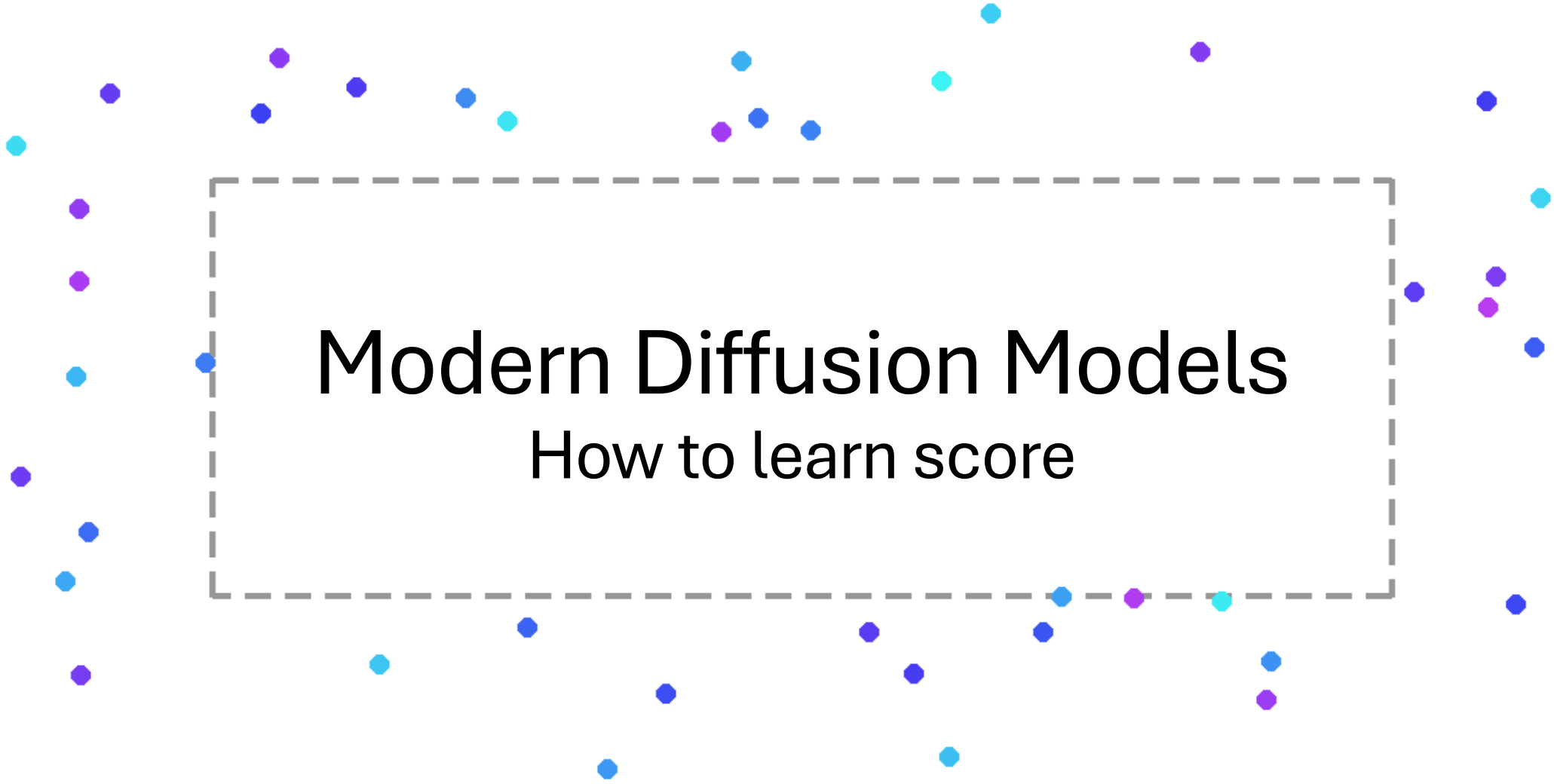

Stochastic process


Reverse stochastic process



Useful analogy to chemistry/physics

Brownian dynamics	⟷	forward noising process
thermodynamic relaxation	⟷	reverse denoising sampler
force / drift field	⟷	score or noise predictor
temperature / noise scale	⟷	time step or noise level t
equilibrium distribution	⟷	data distribution

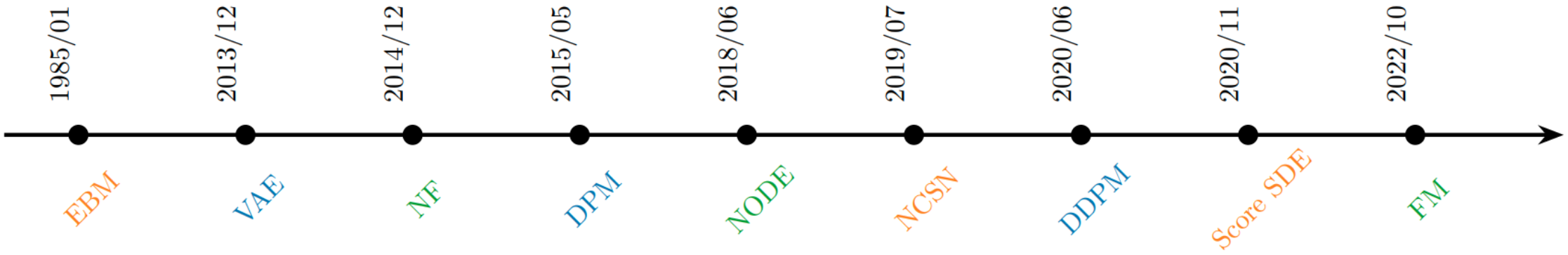


Modern Diffusion Models

How to learn score

Timeline for Diffusion Models

https://arxiv.org/abs/2510.21890



Variational Autoencoder (VAE) (Kingma and Welling, 2013) → Diffusion Probabilistic Models (DPM) (Sohl-Dickstein et al., 2015) → DDPM (Ho et al., 2020)

Energy-Based Model (EBM) (Ackley et al., 1985) → Noise Conditional Score Network (NCSN) (Song and Ermon, 2019) → Score SDE (Song et al., 2020c)

Normalizing Flow (NF) (Rezende and Mohamed, 2015) → Neural ODE (NODE) (Chen et al., 2018) → Flow Matching (FM) (Lipman et al., 2022)

Implicit Score Matching – Hyvärinen 2005

Estimator of the true score: $J(\theta) = \frac{1}{2}\mathbb{E}_{x\sim q_{data}(x)}[||s_\theta(x) - \nabla_x \log q_{data}(x)||^2]$

Implicit score matching shows that this loss function is equivalent to:

$$J_{\mathrm{I}}(\theta) = \mathbb{E}_{x\sim q_{data}(x)}\left[\mathrm{Tr}\left(\nabla_x s_\theta(x)\right) + \frac{1}{2}||s_\theta(x)||^2\right]$$

via integration-by-parts.

The trace of the Hessian $\mathrm{Tr}\left(\nabla_x s_\theta(x)\right)$ is expensive though.

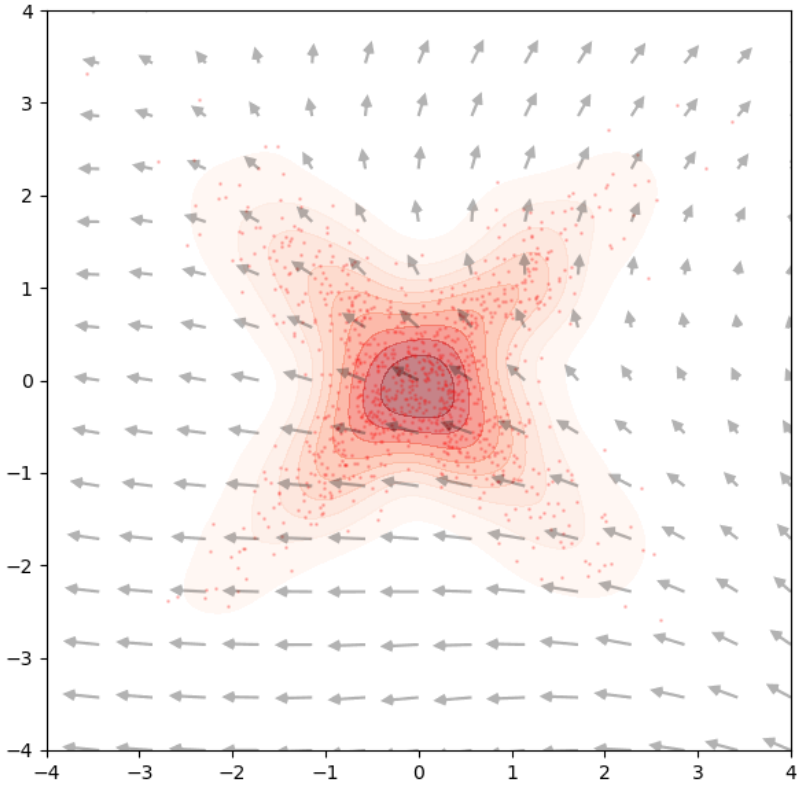
Denoising Score Matching – Pascal 2011

Another equivalent objective:

$$J_{\mathrm{D}}(\theta) = \mathbb{E}_{x\sim q_{data}(x),\epsilon\sim\mathcal{N}(0,I)}\left[\frac{1}{2\sigma^2}\left|\left|\epsilon_\theta(\tilde{x}) - \epsilon\right|\right|^2\right]$$

Here the "noise estimator" learns *just* the original Gaussian noise vector ϵ that was added to craft the noisy sample.

The network ϵ_θ learns roughly a unit variance direction that points towards clean samples.



Denoising Score Matching

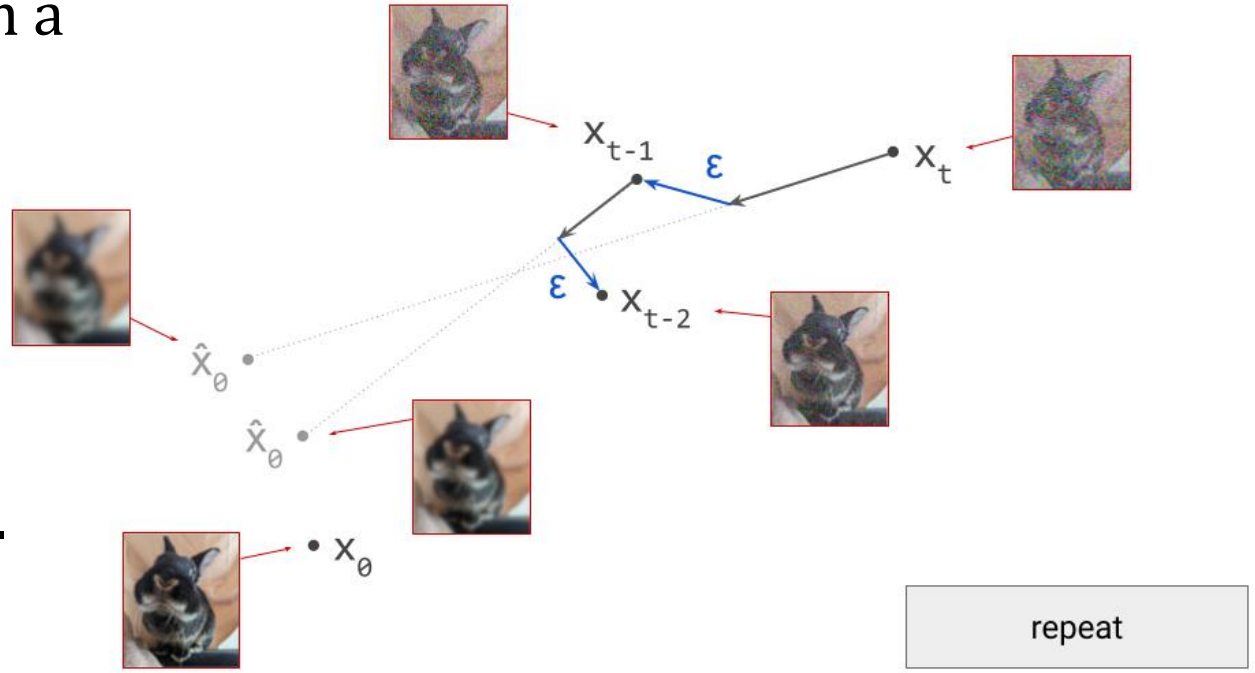
A clean x and noisy $\tilde{x}$ sample come from a joint distribution:

$$q(x, \tilde{x}) = q(x|\tilde{x})q(\tilde{x})$$

$q(x|\tilde{x})$ is the distribution of all clean samples which could've led to a given $\tilde{x}$.

We can write the objective as:

$$J_{\mathrm{D}}(\theta) = \mathbb{E}_{\tilde{x}\sim q(\tilde{x})}\left[\frac{1}{2\sigma^2}\left\|\epsilon_\theta(\tilde{x}) - \frac{\tilde{x} - \mathbb{E}_{x\sim q(x|\tilde{x})}[x]}{\sigma}\right\|^2\right]$$



https://sander.ai/2023/08/28/geometry.html

Incorporating Time

Excluded time sampling for brevity, but the score must be estimated along each timestep of the forward process.

We add the time variable $t \in \mathcal{U}[0,1]$ to the denoising loss:

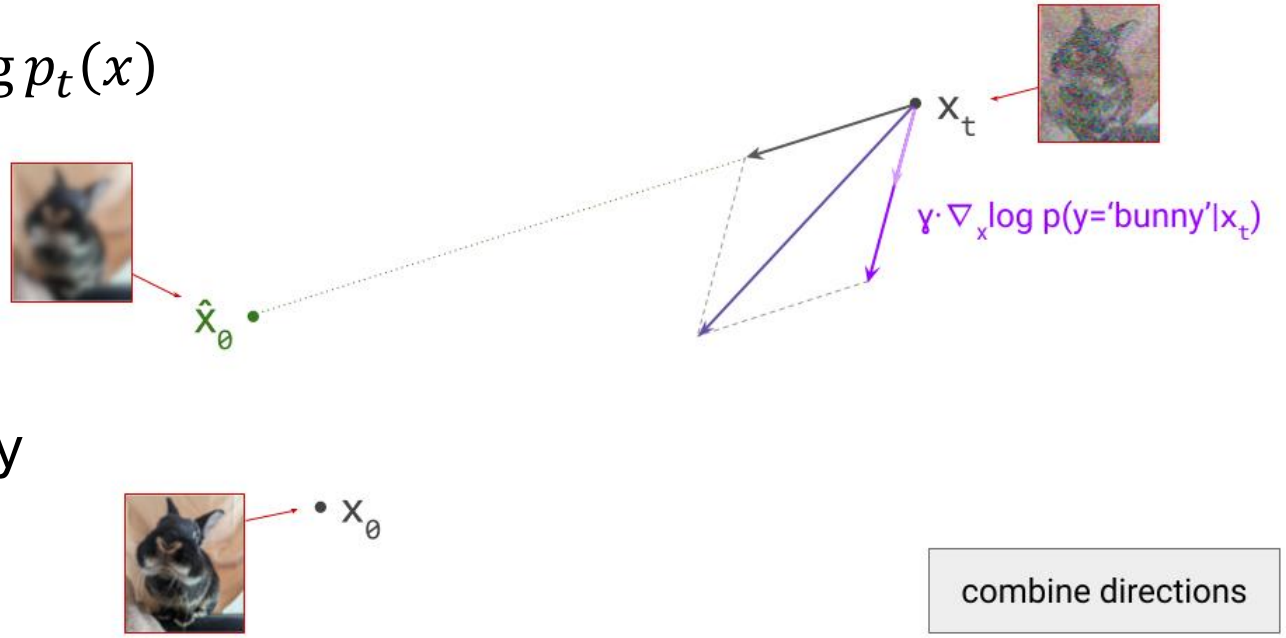
$$J_{\mathrm{D}}(\theta) = \mathbb{E}_{x_0,\, \epsilon,\, t\sim\mathcal{U}[0,1],\, x_t\sim q_t(x_t|x_0)} \left[\frac{1}{2} \left\| s_\theta(x_t, t) - \left(-\frac{\epsilon}{\sigma_t} \right) \right\|^2 \right]$$

Classifier Guidance

Obtain conditional score function as sum of unconditional score and a conditional term:

$$\nabla_x \log p_t(x|y) = \nabla_x \log p_t(y|x) + \nabla_x \log p_t(x)$$

Just need a classifier that maps x to target labels y.



We can tune the amount of guidance by setting a "*temperature*":

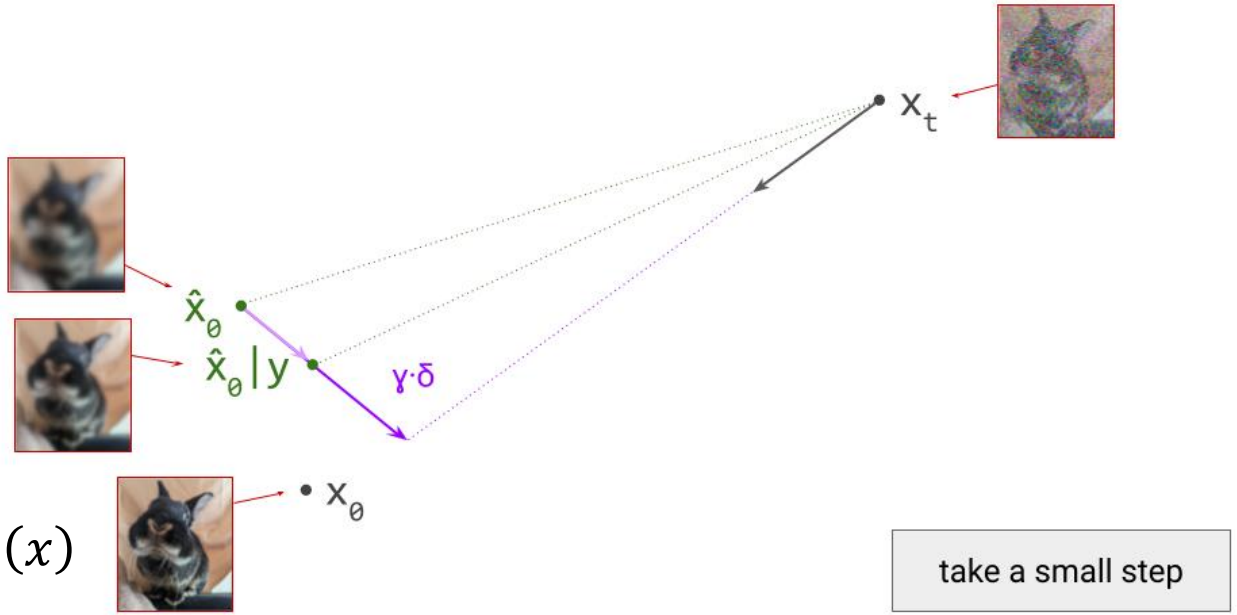
$$\nabla_x \log p_t^{\gamma}(x|y) = \gamma \nabla_x \log p_t(y|x) + \nabla_x \log p_t(x)$$

https://sander.ai/2023/08/28/geometry.html

Classifier-*free* Guidance

Does not require training separate classifier.

Instead, we train *one* conditional diffusion model $p(x|y)$ with **conditioning dropout**. Conditioning information y is absent/*null* ~ 10-20% of the time.



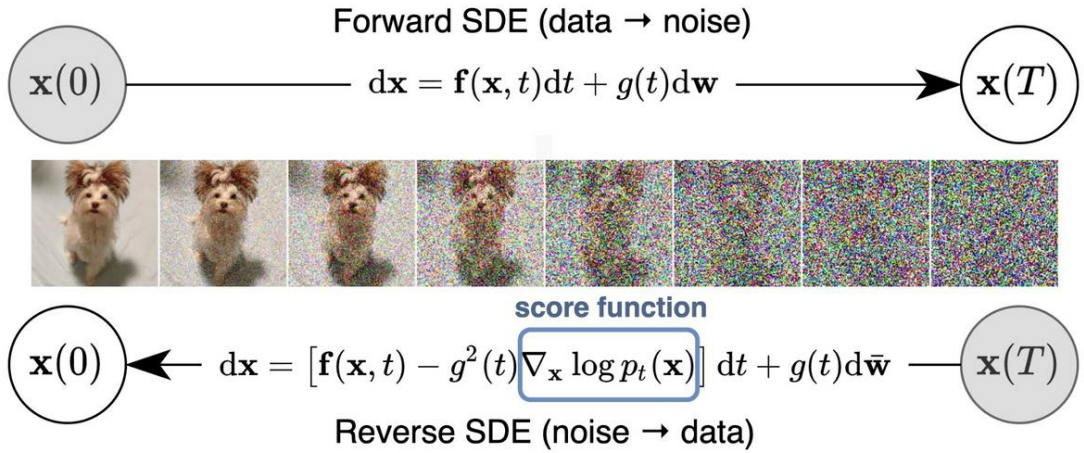
Use Bayes theorem for the classifier score:

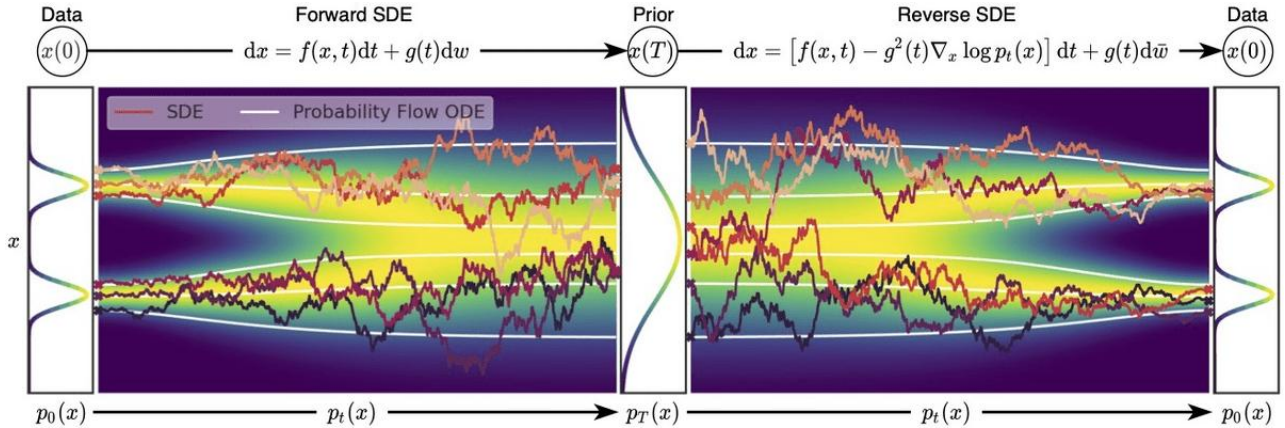
$$\nabla_x \log p_t(y|x) = \nabla_x \log p_t(x|y) - \nabla_x \log p_t(x)$$

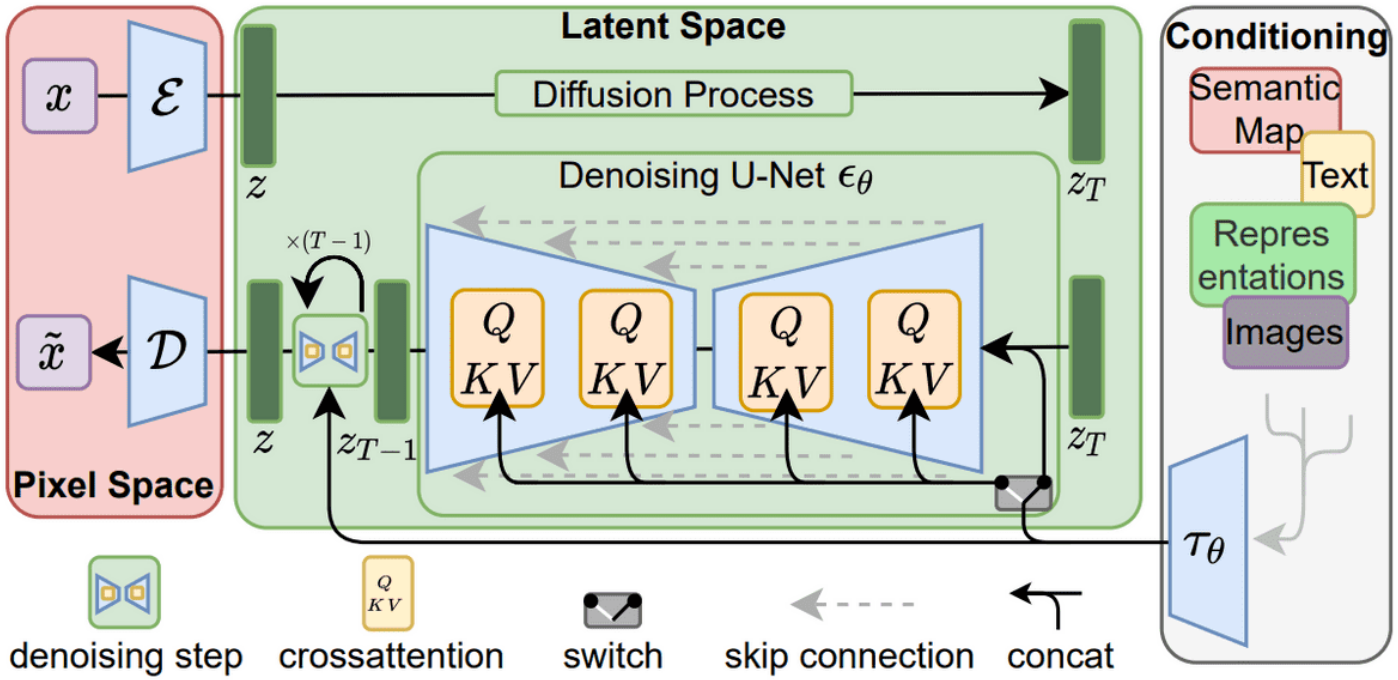
Get barycentric combination of conditional and unconditional scores:

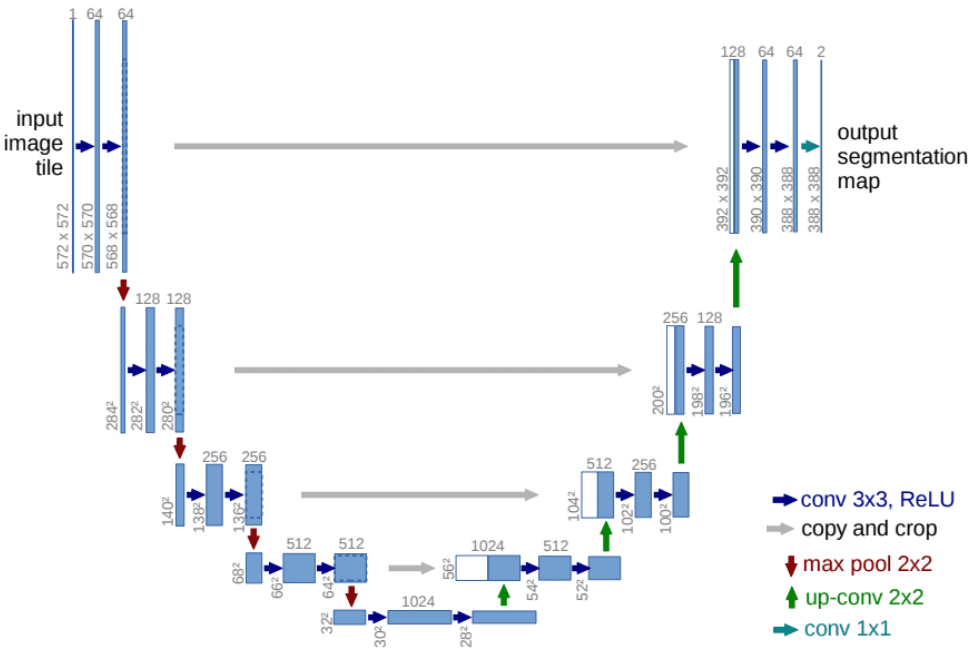
$$\nabla_x \log p_t(x|y) = \gamma \nabla_x \log p_t(x|y) + (1-\gamma) \nabla_x \log p_t(x)$$

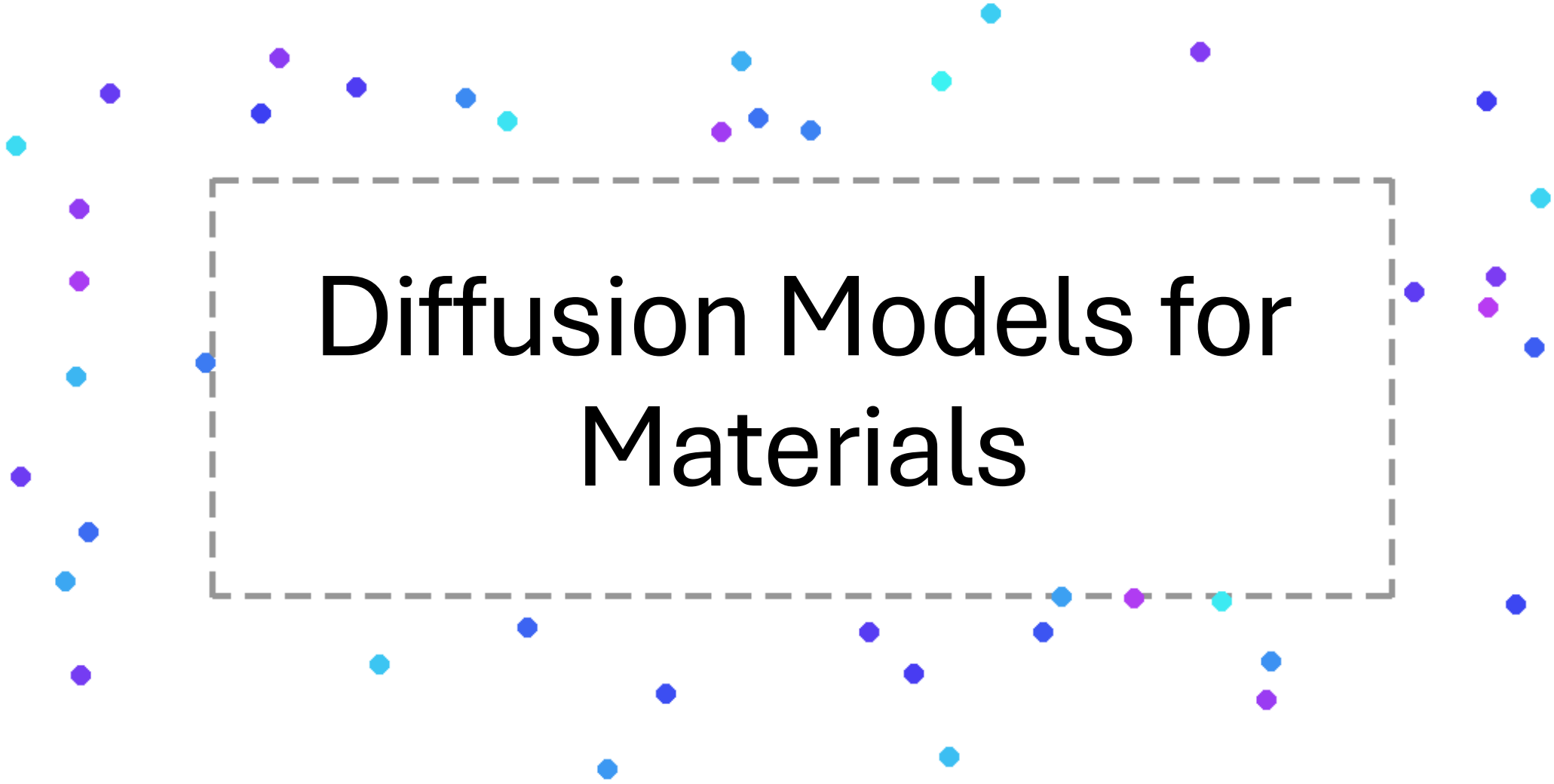
https://sander.ai/2023/08/28/geometry.html


Forward SDE (data → noise)
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$
$\mathbf{x}(T)$
score function
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$
$\mathbf{x}(T)$
Reverse SDE (noise → data)


Data
Forward SDE
Prior
Reverse SDE
Data
$x(0)$
$dx = f(x,t)dt + g(t)dw$
$x(T)$
$dx = \left[f(x,t) - g^2(t)\nabla_x \log p_t(x)\right] dt + g(t)d\bar{w}$
$x(0)$
SDE
Probability Flow ODE
x
$p_0(x)$
$p_t(x)$
$p_T(x)$
$p_t(x)$
$p_0(x)$


Latent Space
Diffusion Process
Denoising U-Net ϵ_θ
$\times(T-1)$
x
$\mathcal{E}$
z
z_T
$\tilde{x}$
$\mathcal{D}$
z
z_{T-1}
z_T
Q
K V
Pixel Space
Conditioning
Semantic Map
Text
Repres entations
Images
τ_θ
denoising step
crossattention
switch
skip connection
concat


input image tile
output segmentation map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1



Diffusion Models for Materials

Challenges for generating crystals

Periodicity

Atoms exist in a repeating unit cell. Coordinates are defined modulo lattice vectors. The diffusion process must respect periodic boundary conditions.

Permutation invariance

Swapping atom labels in the unit cell shouldn't change the crystal. The model must be invariant to atom re-ordering.

SE(3) / E(3) equivariance

Physical properties don't change under rotation, translation, or reflection. The score network must respect these symmetries.

Mixed representations

A crystal = lattice (6 continuous params) + fractional coords (3N continuous) + atom types (N discrete). Need joint diffusion over all three.

Representing periodic material

$\boldsymbol{M = (X, A, L)}$:

atom types

discrete

$\boldsymbol{A \in \mathbb{A}^n}$

fractional coordinates

continuous

$\boldsymbol{X \in [0, 1)^{3\times n}}$

lattice parameters

continuous

$\boldsymbol{L = (l_1, l_2, l_3) \in \mathbb{R}^{3\times 3}}$

This is why crystal generators often need hybrid corruption processes or carefully coupled heads.

A "good" crystal model must coordinate chemistry, geometry, and periodicity simultaneously.

Modern update: **Wyckoff positions** (special symmetric sites within each space group) are now used as a descriptor in models like DiffCSP++ (ICLR 2024), constraining atom positions to symmetry-allowed sites during diffusion.

Discrete Diffusion

The **problem**: Standard diffusion adds Gaussian noise to continuous variables (coordinates, latent vectors). But many quantities in chemistry are inherently discrete:

Atom types

H, C, N, O, S, ...
Categorical, not continuous

Space groups

230 crystallographic symmetry classes

Bond orders

Single, double, triple
Discrete integers

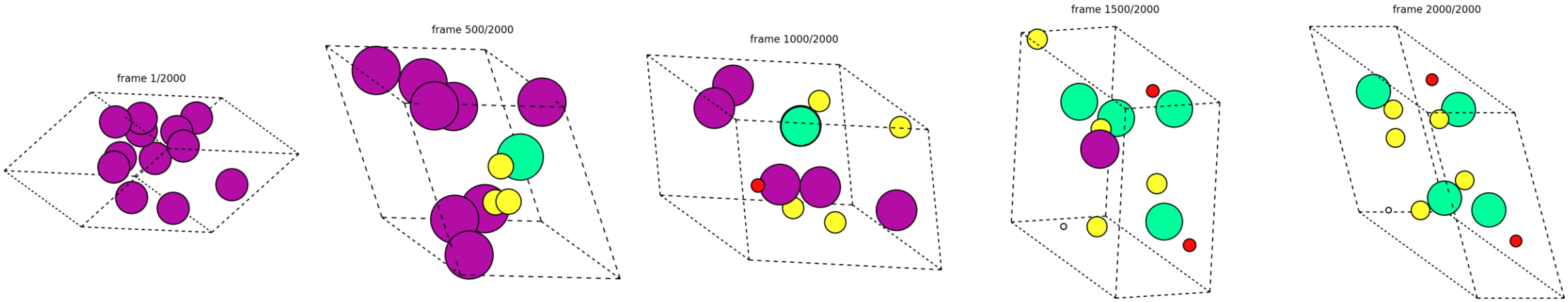
Key idea: Replace Gaussian noise with transition matrices on categorical states. Instead of x + noise, we randomly flip categories according to a Markov chain.

D3PM – Discrete Denoising Diffusion (Austin et al. 2021)

Forward process: Corrupt discrete tokens via transform matrices.

$$q(x_t|x_{t-1}) = \mathrm{Cat}(x_t; Q_t x_{t-1})$$

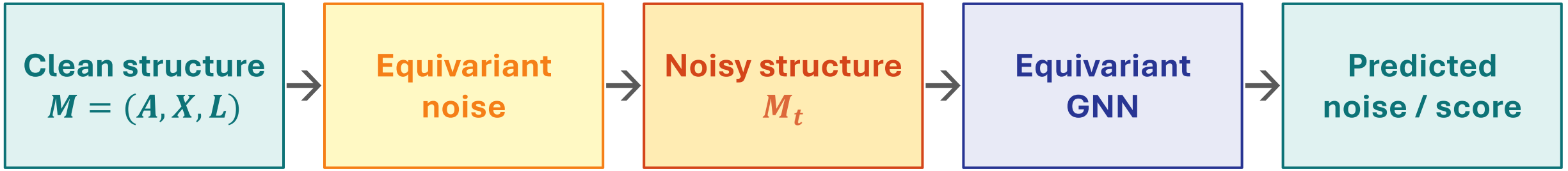
where Q_t is a $K \times K$ transition matrix (K = number of categories)



Reverse: Train a neural network to predict $p(x_{t-1}|x_t)$, i.e. which category each token should revert to. Loss is a cross-entropy or variational bound analogous to continuous DDPM.

Equivariant Diffusion

Key insight (Hoogeboom et al., 2022; EDM): If the forward noising process is equivariant and the denoising network is equivariant, the entire generative model is equivariant.



For coordinates: Add isotropic Gaussian noise (equivariant under rotation). For periodic systems, noise is added in fractional coordinates and wrapped.

For atom types: Use discrete diffusion (D3PM / absorbing states). Already permutation-invariant by construction.

For lattice: Diffuse lattice matrix L directly (or its representation). Must handle the gauge freedom: many L matrices describe the same lattice.

Handling Periodicity: Wrapped Diffusion

Fractional coordinates live on a torus: $x_i \in [0,1)^3$ *with periodic boundaries.*

Approach 1: Wrap after noise

Add Gaussian noise, then take $\mathrm{mod}\ 1$.

$$x_t = (x_{t-1} + \mathrm{noise}) \bmod 1$$

Simple but the score near boundaries becomes discontinuous.

Used in CDVAE (Xie et al., 2022)

Approach 2: Wrapped normal

Use the wrapped normal distribution on the torus.

$$\mathcal{N}_w(x;\mu,\sigma) = \sum_k \mathcal{N}(x+k;\mu,\sigma)$$

Proper probability distribution on $[0,1)^3$. Score is well-defined everywhere.

Used in DiffCSP (2023), MatterGen (2025)

Both approaches converge to a uniform distribution on the torus as $t \to T$, which is the natural "maximally noisy" state for periodic coordinates.

Landscape of Crystal Diffusion Models

2022	**CDVAE**	*Xie et al.:* VAE + diffusion on coordinates. First to generate periodic crystals. Uses wrap-around for periodicity.
2023	**DiffCSP**	*Jiao et al.:* Score-based diffusion for crystal structure prediction (given composition). Joint lattice + coordinate diffusion.
2025	**MatterGen**	*Zeni et al.:* Full generative model: lattice + coords + atom types. Property-guided generation. Microsoft.
2025	**Chemeleon**	*Park et al.:* Fully discrete (tokenized) approach. Transformer backbone. Unified masked diffusion.
2026	**ScoreMD**	*Plainer et al.:* A single unified model to perform both independent sampling via diffusion denoising and continuous molecular dynamics simulations.

MatterGen (Zeni et al. Microsoft, 2025)

A diffusion model for inorganic material generation with property-guided sampling.

Architecture Overview

- Joint diffusion over atom types (discrete), fractional coordinates (periodic continuous), and lattice matrix (continuous)
- GNN backbone based on equivariant message passing (adapted from DimeNet++/GemNet-style interactions)
- Atom types: absorbing-state discrete diffusion (tokens -> [MASK])
- Coordinates: wrapped Gaussian diffusion on the torus $[0,1)^3$
- Lattice: direct diffusion on the 3x3 lattice matrix with appropriate invariances
- Classifier-free guidance for conditional generation on chemistry / target properties

Key results: Generates stable, novel crystal structures across diverse chemistries. Can be conditioned on target properties (bulk modulus, band gap, magnetic density) to steer generation.

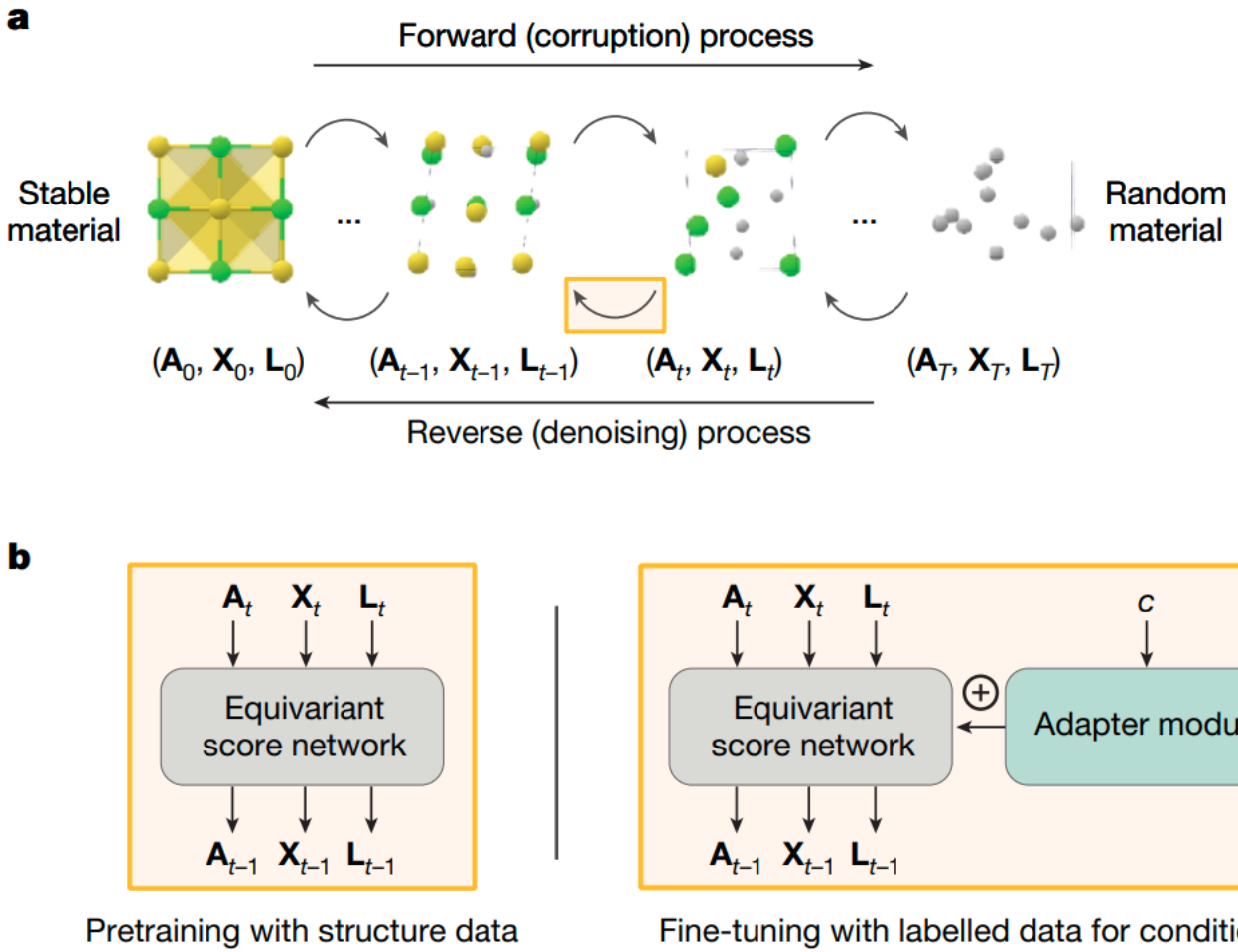
MatterGen (Zeni et al. Microsoft, 2025)

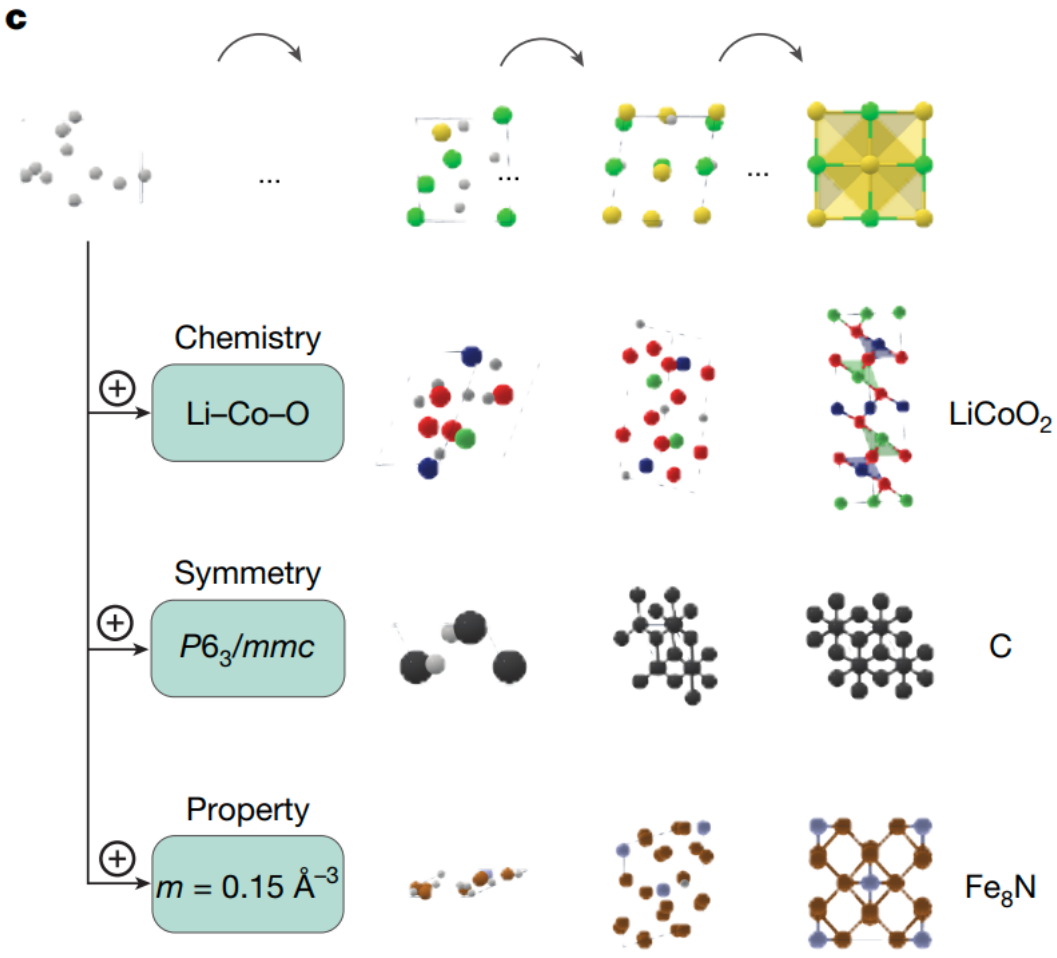
Conditioning types supported:

Chemistry	Specify allowed elements (e.g., "only Li, Mn, O"). Achieved by masking the discrete diffusion to a subset of atom types.
Symmetry	Target a specific space group or crystal system. Constraint applied during the denoising steps.
Scalar properties	Target bulk modulus, band gap, magnetic moment, etc. Uses classifier-free guidance with a property regressor.
Composition	Fix the stoichiometry (e.g., generate all ABX3 perovskites). Constrains number of atoms and type counts.

Workflow: Generate candidates → DFT relaxation → Filter by stability (energy above hull) → Experimental validation. Microsoft reported synthesizing novel materials discovered this way.

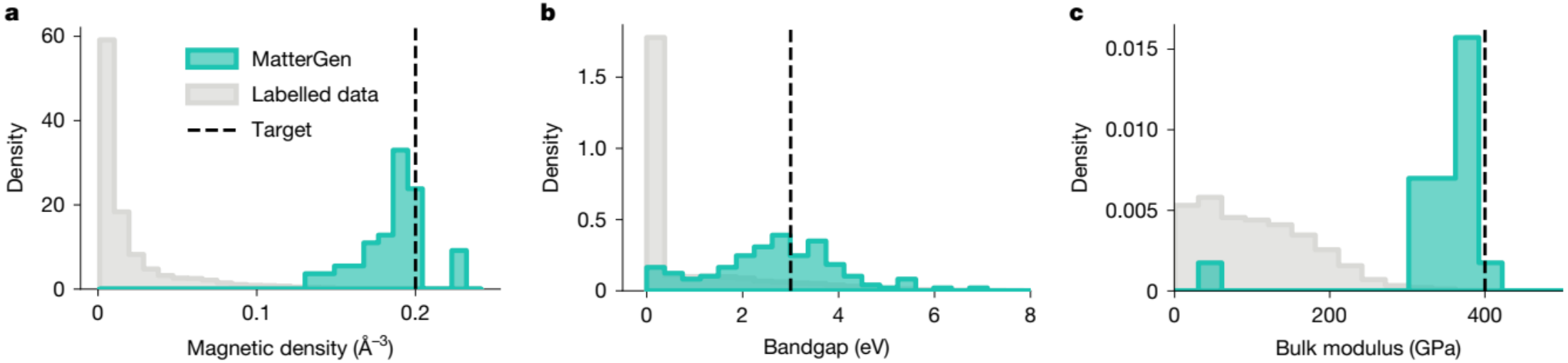
MatterGen (Zeni et al. Microsoft, 2025)





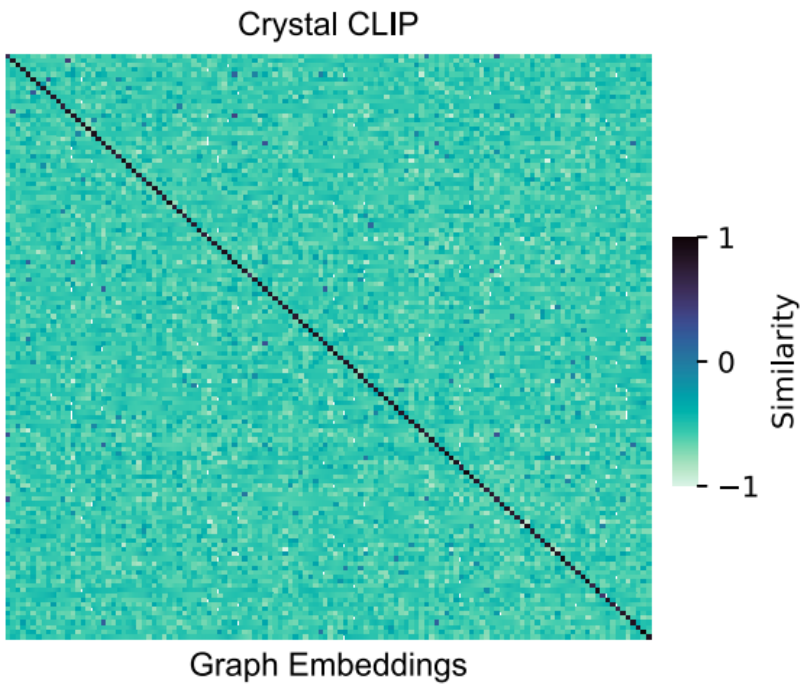
MatterGen (Zeni et al. Microsoft, 2025)

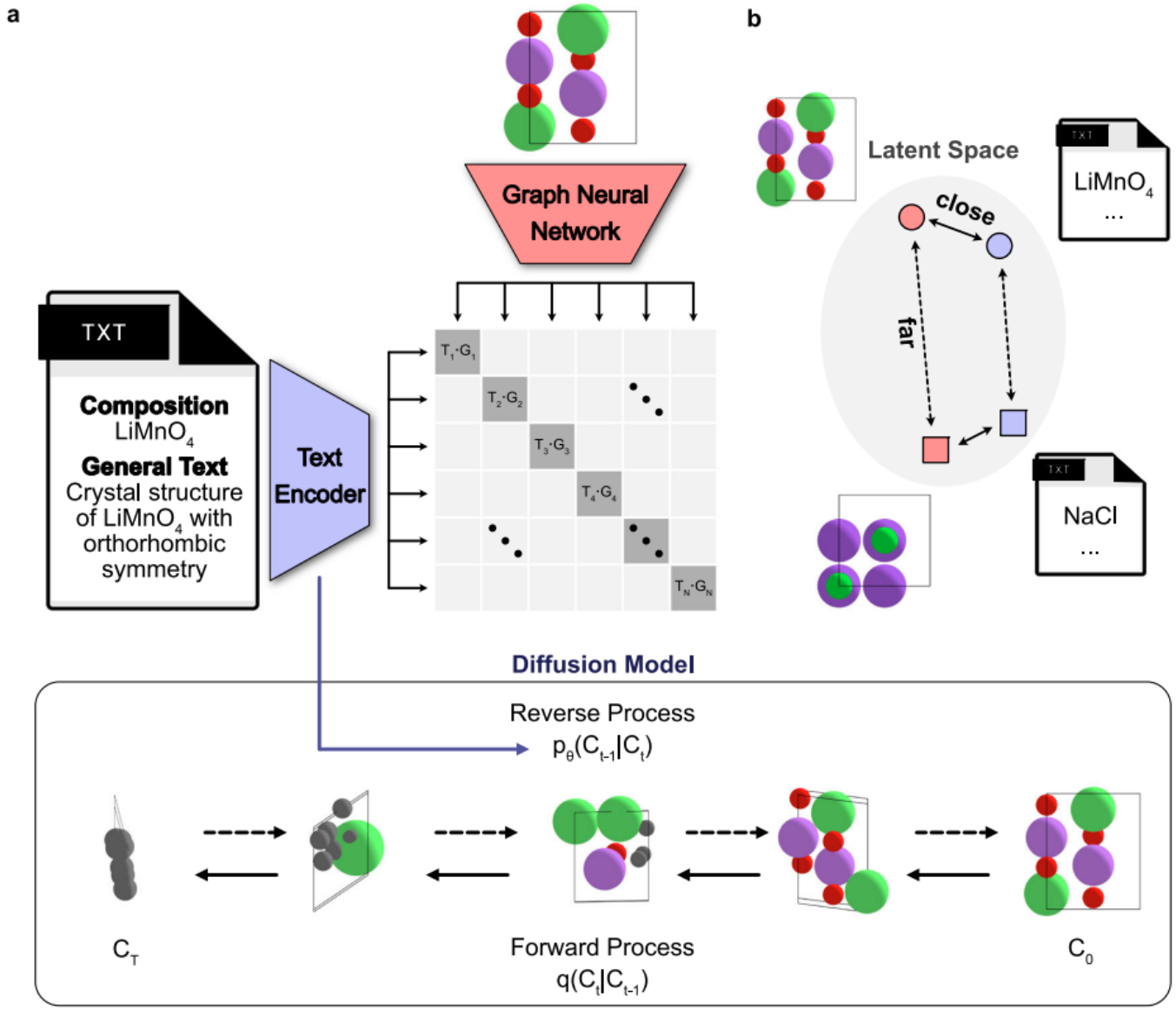
Distribution shift from conditioning:



Chemeleon (Park et al. 2025)

A unified framework for crystal generation using language-model-inspired discrete diffusion.





Practical Considerations for Material Diffusion

From generated structures to real materials:

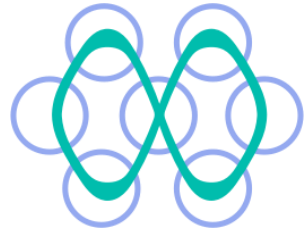
Validation pipeline	Generated structures need DFT relaxation, stability checks (energy above convex hull), and comparison to known databases (Materials Project, ICSD, AFLOW).
Evaluation metrics	Validity (does it relax to a sensible structure?), uniqueness, novelty (is it in the training set?), coverage and recall of known stable phases, match rate.
Training data	Typically trained on Materials Project (~150k structures) or ICSD. Data augmentation via symmetry operations. Careful treatment of prototype structures.
Sampling speed	Crystal diffusion models typically need 500-1000 denoising steps. Accelerated sampling (DDIM-like) is an active area. Much cheaper than random DFT screening.
Integration with MLIPs	Generated structures can be relaxed with MACE/CHGNet before expensive DFT, creating a fast screening pipeline. Links to earlier lectures!

Summary & Key Takeaways

1. **Score function:** The gradient of log-density avoids the normalization problem and enables sampling via Langevin dynamics.
2. **Forward / Reverse:** Gradually corrupt data with noise (forward SDE); learn to reverse it (denoising score matching).
3. **Guidance:** Classifier and classifier-free guidance enable conditional generation without retraining.
4. **Discrete diffusion:** Transition matrices on categories extend diffusion to atom types, space groups, and tokens.
5. **Crystal generation:** Must handle periodicity (torus), equivariance (E(3)), and joint continuous + discrete variables.
6. **State of the art:** MatterGen, Chemeleon, DiffCSP: different architectures, same goal - generate stable novel materials.

Hands-on!

https://next-gen.materialsproject.org/



There are **4** notebooks available:

1. **Diffusion Fundamentals** – investigate simple diffusion processes and common models on toy systems
2. **Crystal Diffusion From Scratch** – build a compact crystal diffusion model from scratch from a Materials Project dataset*
3. **MatterGen In Practice** – run a pretrained MatterGen diffusion model for crystal generation and relax with MACE
4. **Chemeleon-DNG In Practice –** run a pretrained Chemeleon-DNG model, relax with MACE, explore formation energy for fractional compositions