Deep Learning in Life Sciences

Diffusion models for protein design

Elodie Laine

Laboratory of Computational and Quantitative Biology

e-mail: elodie.laine@sorbonne-universite.fr

@LaineElodie

CHARLES UNIVERSITY

SORBONNE UNIVERSITÉ

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

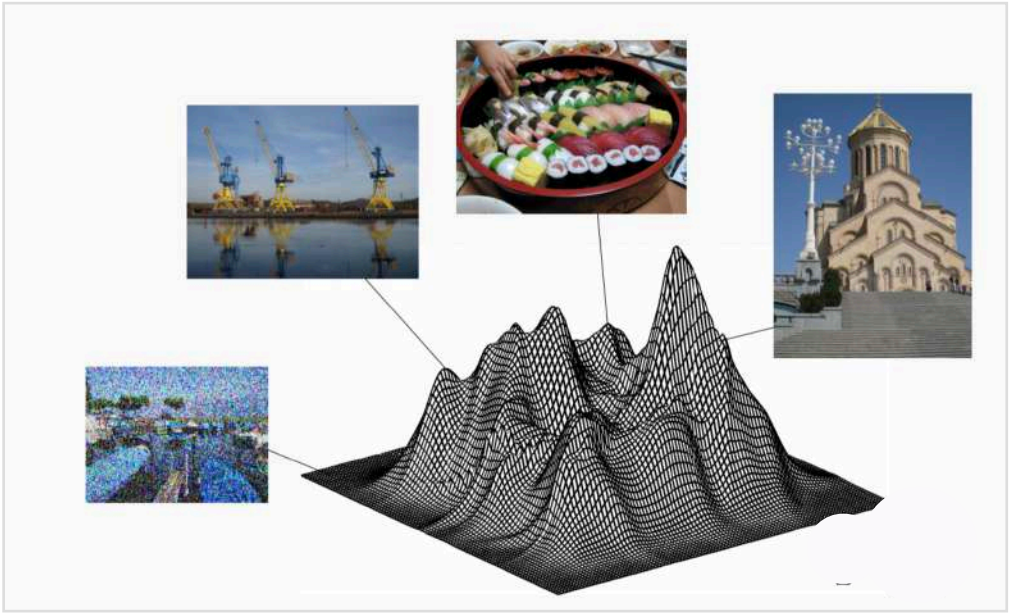
UNIVERSITY OF WARSAW

UNIVERSITÀ DEGLI STUDI DI MILANO

4eu+ EUROPEAN UNIVERSITY ALLIANCE

The problem of generative modelling

*Parametric density estimation: we want to learn the distribution underlying the training samples to generate new **plausible** samples that are similar but not identical.*



credits: Charles Deledalle

What do we have?

A finite set of samples, $\{x^{(n)}\}_{n=1}^{N}$, from a true but unknown data distribution $q_{data}(x)$.

What do we want?

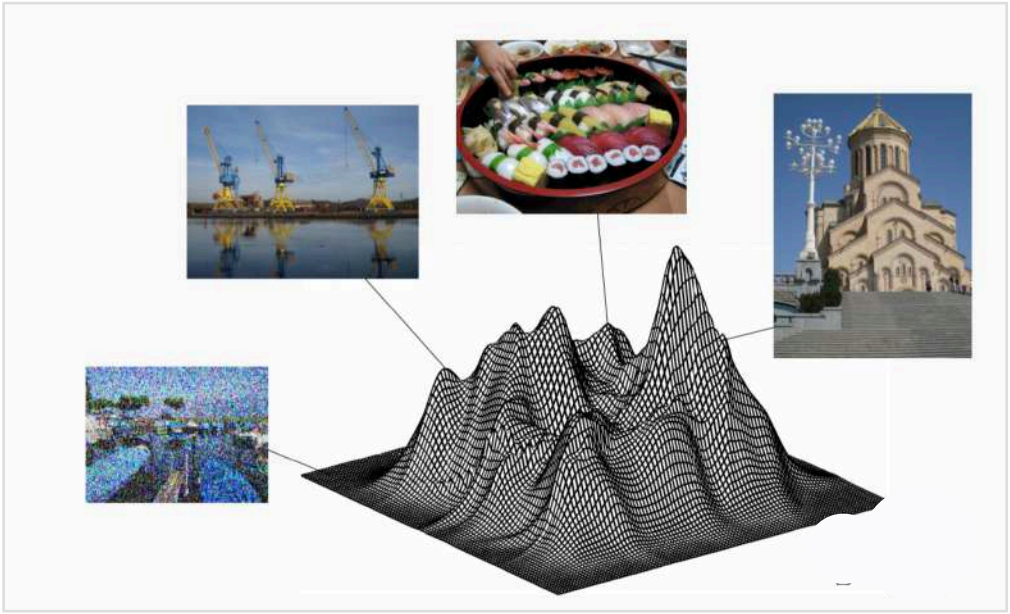
Estimate $q_{data}(x)$ with a suitable model family $p_\theta(x)$, with unknown parameters θ.

The problem boils down to maximising the average likelihood (w.r.t. θ) of all the samples under the model,

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

The problem of generative modelling

*Parametric density estimation: we want to learn the distribution underlying the training samples to generate new **plausible** samples that are similar but not identical.*



credits: Charles Deledalle

Is it simple?

No! The likelihood is intractable because one has to consider all the data points in the distribution to estimate any point's probability.

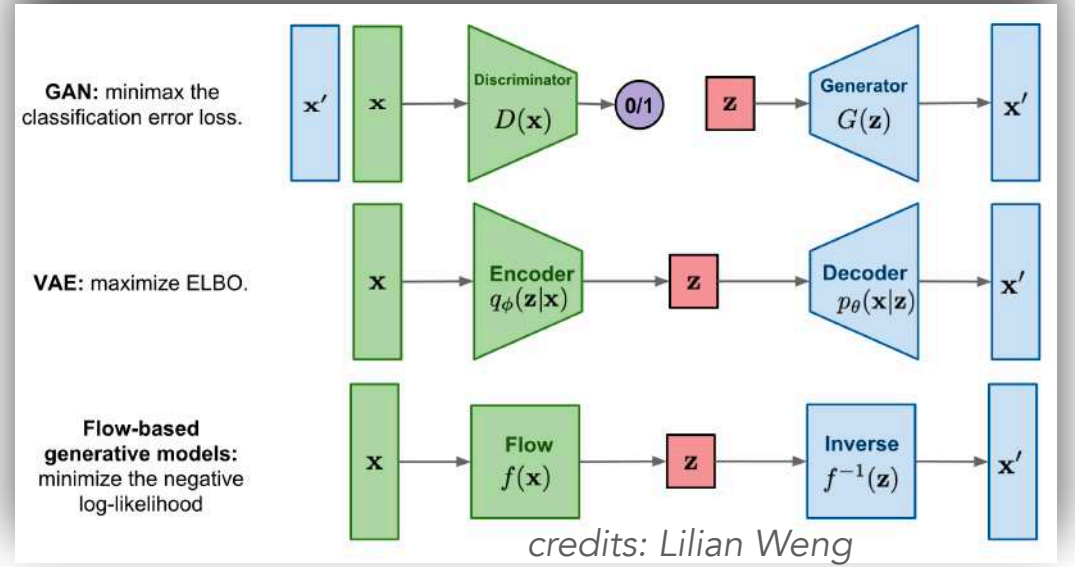
$$p_\theta(x) = \frac{\tilde{p}_\theta(x)}{\int_x \tilde{p}_\theta(x)}$$

Normalisation constant

What can we do?

- Restrict the architecture to ensure tractability
- Rely on surrogate objectives
- Avoid evaluating probabilities (implicit models)

Sampling is essential!



credits: Lilian Weng

Sampling-centric generative models operate by transforming simple probabilistic densities.

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

Functional transformation (neural network)

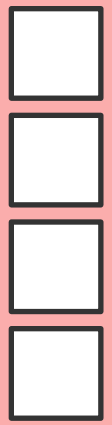
Simple density (often standard normal)

In **diffusion models**, the transformation f_θ is a sequence of invocations of a neural function, denoted as s_θ, along with some additional computations, denoted as $g(\,.\,)$.

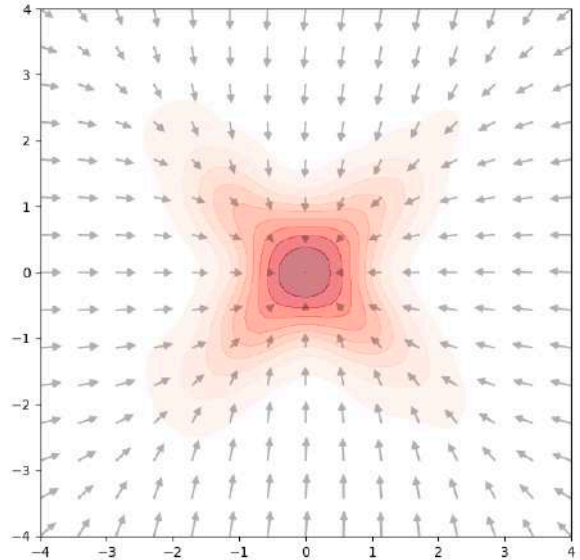
$$x = g_1(g_2(g_3(\cdots z \cdots, s_\theta), s_\theta), s_\theta), \text{ where } z \sim \mathcal{N}(0,I)$$

Outline

- Why the score s_θ?
- Why diffusion? (backward)
- Why forward?
- Why so popular?



Why the score?



Vector field representing the score of the true distribution at any point

credits: Ayan Das

Diffusion models aim at estimating the (Stein) **score function**, defined as the **gradient of the log of the density of the data**,

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

It reflects the direction of steepest increase in log-likelihood at any given point in the data space.

At a point x, the **score** gives us the best direction to step into (with little step size δ) if we would like to see a point x' with slightly higher likelihood,

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

Why is it more convenient to estimate the score instead of the density?

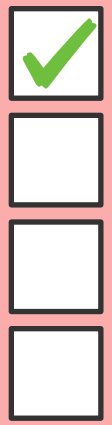
$\Longrightarrow$ because the score **does not depend** on the normalising constant!

$$\mathbf{s}_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} \tilde{p}_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log \int_x \tilde{p}_\theta(x)}_{=0} = \nabla_{\mathbf{x}} \tilde{p}_\theta \,.$$

Outline

DeepLearning inLifeSciences

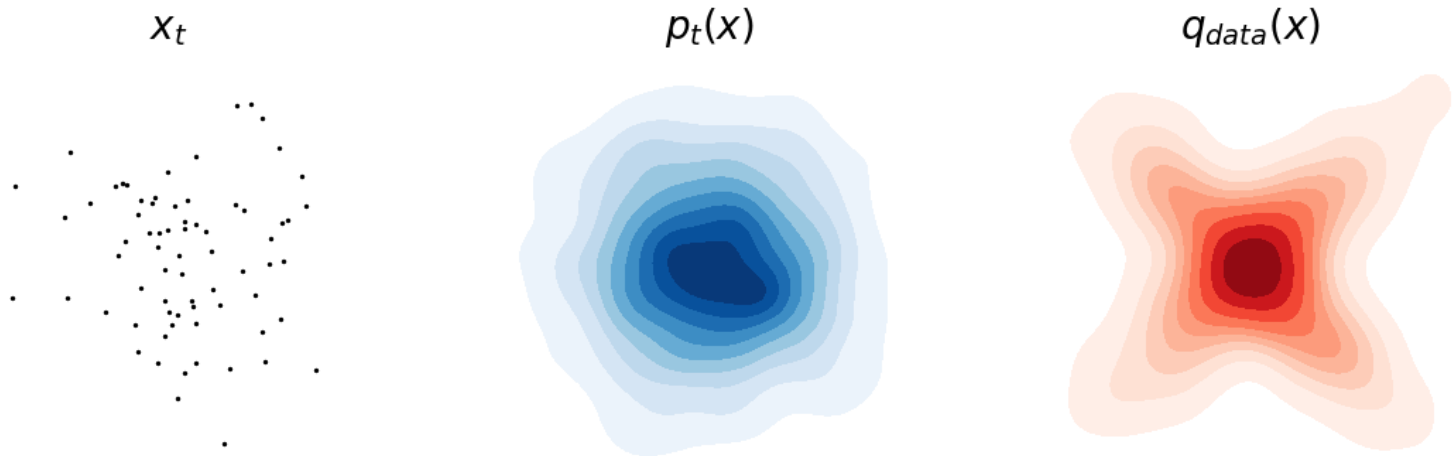
- Why the score s_θ?
- Why diffusion? (backward)
- Why forward?
- Why so popular?



Sampling from the true score

$$x' = x + \delta \cdot \nabla_x \log q_{data}(x)\Big|_{x=x} \xrightarrow{\delta \to 0} dx = \nabla_x \log q_{data}(x) dt$$

Iteratively updating x from a normal distribution along the score **concentrate** the points in the most likely region.



credits: Ayan Das

Langevin dynamics

French physicist *Paul Langevin* modelled the evolution of the positions of particles over time t under a potential energy field $U(x)$ as, $dx = -\nabla_x U(x)dt + \sqrt{2}dB_t$

Brownian motion (noise)

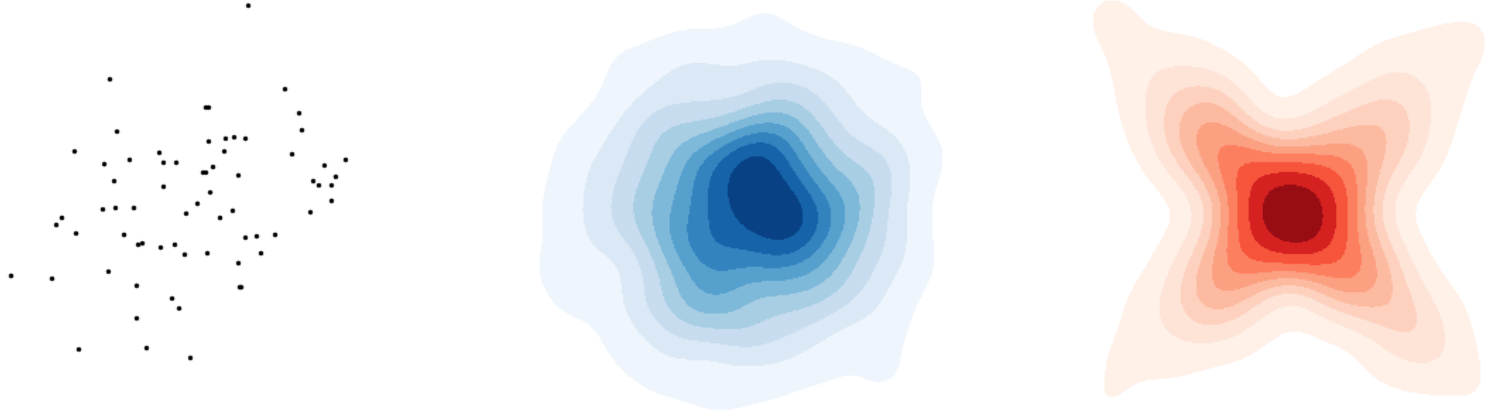
$dB_t = \mathcal{N}(0,dt) \implies dB_t = \sqrt{dt} \cdot z,$

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

The energy is linked to the probability density,

$$q_{data}(x) = \frac{e^{-U(x)}}{Z} \quad \text{s.t.} \quad dx = \nabla_x \log q_{data}(x)dt + \sqrt{2}dB_t$$

x_t $p_t(x)$ $q_{data}(x)$



credits: Ayan Das

Langevin dynamics

French physicist *Paul Langevin* modelled the evolution of the positions of particles over time t under a potential energy field $U(x)$ as, $dx = -\nabla_x U(x)dt + \sqrt{2}dB_t$

Brownian motion (noise)

$dB_t = \mathcal{N}(0,dt) \implies dB_t = \sqrt{dt} \cdot z,$

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

The energy is linked to the probability density,

$$q_{data}(x) = \frac{e^{-U(x)}}{Z} \quad \text{s.t.} \quad dx = \nabla_x \log q_{data}(x)dt + \sqrt{2}dB_t$$

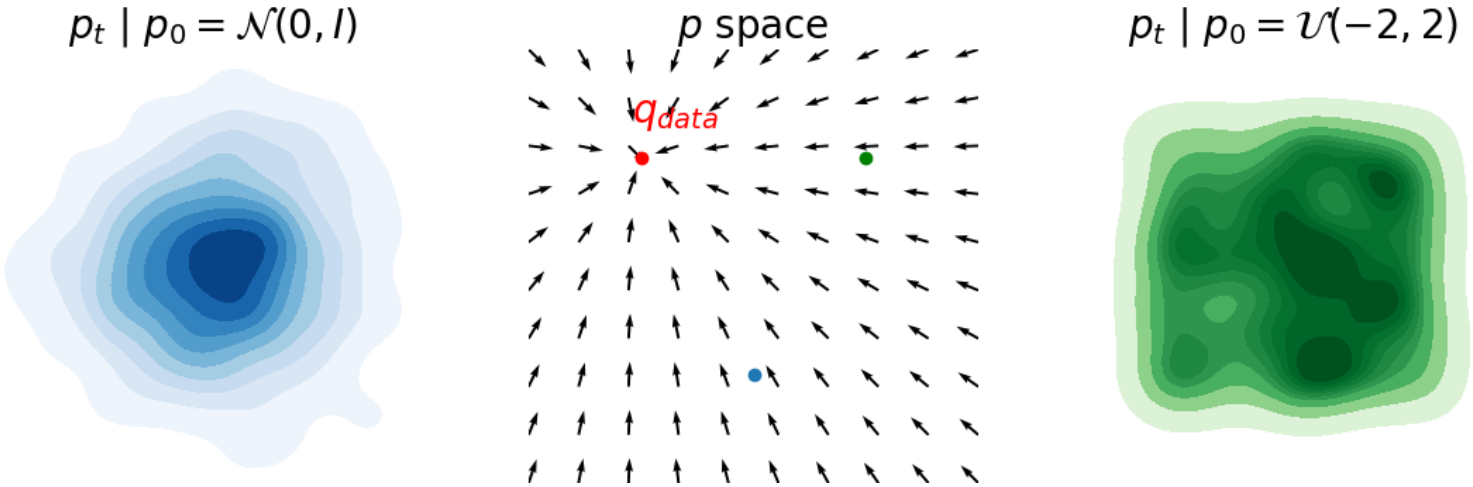
This stochastic process induces a time-varying distribution $p_t(x)$ described by the *Fokker-Planck* equation,

$$\frac{\partial}{\partial t}p_t(x) = -\frac{\partial}{\partial x}\left[p_t(x)\nabla_x \log q_{data}(x)\right] + \frac{1}{2}\frac{\partial^2}{\partial x^2}\left[p_t(x)\sqrt{2}\right]$$

The probability distribution **provably converges** to the data distribution $p_\infty(x) = q_{data}(x)$.

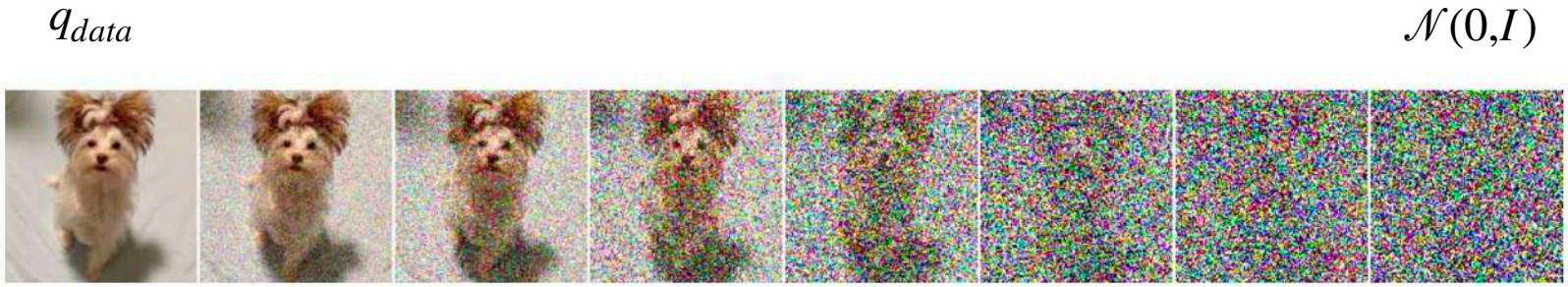
credits: Ayan Das

Illustration of the convergence



credits: Ayan Das

Going backward



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

Backward process

Outline

DeepLearning inLifeSciences

- Why the score s_θ?
- Why diffusion? (backward)
- Why forward?
- Why so popular?

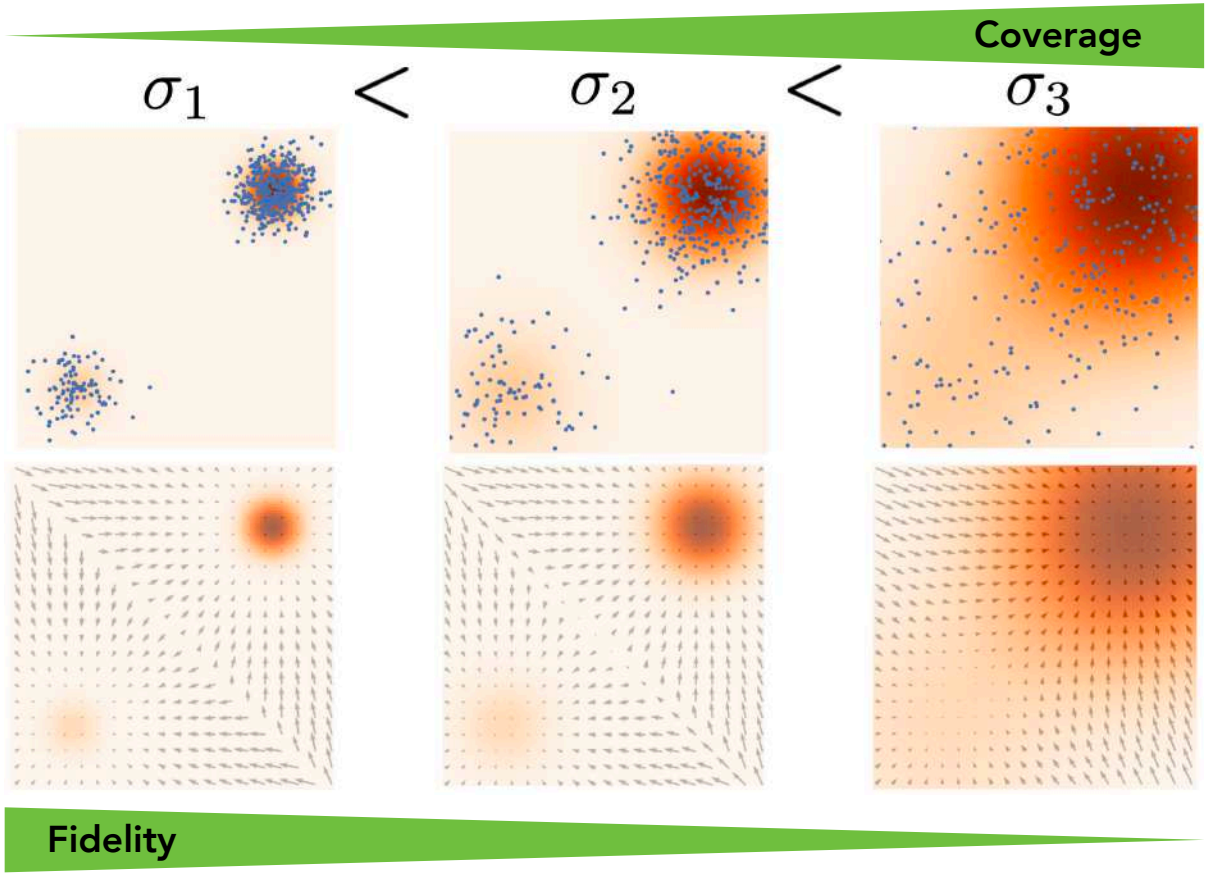
Score matching

To estimate the score, we would ideally minimise a loss function that reflects the error to the true score.

$$J(\theta) = \frac{1}{2}\mathbb{E}_{x\sim q_{data}(x)}\left[|| s_\theta(x) - \nabla_x \log q_{data}(x) ||^2 \right]$$

Problem: we do not have access to the true score.

Augmenting data with noise



credits: Yang Song

Estimating empirical scores at different scales of noise, improves the accuracy of the score estimation!

Score matching

To estimate the score, we would ideally minimise a loss function that reflects the error to the true score.

$$J(\theta) = \frac{1}{2}\mathbb{E}_{x\sim q_{data}(x)}\left[\,||\,s_\theta(x) - \nabla_x \log q_{data}(x)\,||^2\right]$$

Problem: we do not have access to the true score.

Pascal Vincent established in 2011 the connection between score matching and denoising autoencoders,

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

We estimate the score at a noisy version of x

We compare with the noise added to x (which we know by design!)

A connection between score matching and denoising autoencoders
P. Vincent. Neural computation. MIT Press. 2011.

Going forward

DeepLearning inLifeSciences

Forward process

$$dx = \nabla_x \log \mathcal{N}(0,I)dt + \sqrt{2}dB_t$$

$$dx = -xdt + \sqrt{2dt}z$$

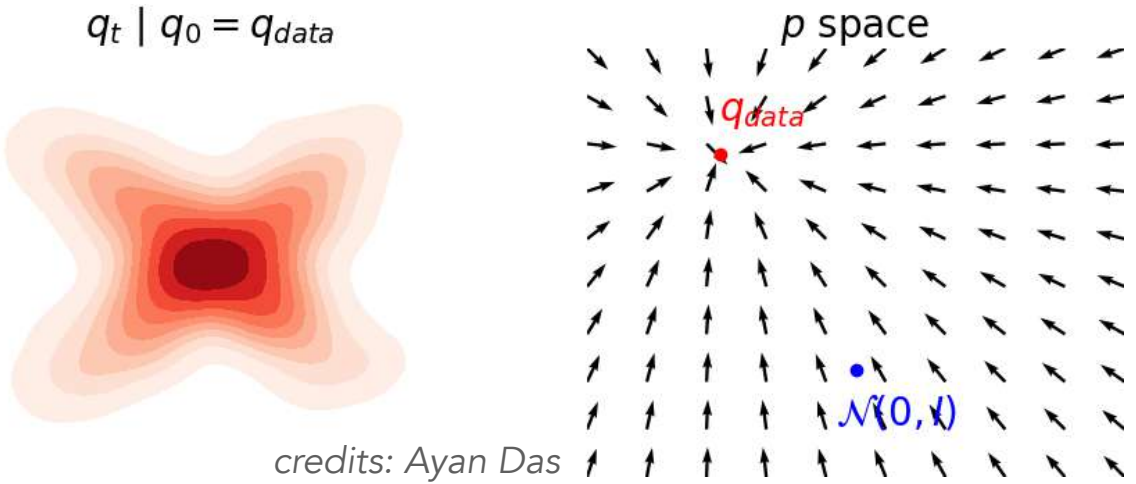
q_{data} → $\mathcal{N}(0,I)$

$x(0)$ ← $x(T)$

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

Backward process

Illustration of the convergence



credits: Ayan Das

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

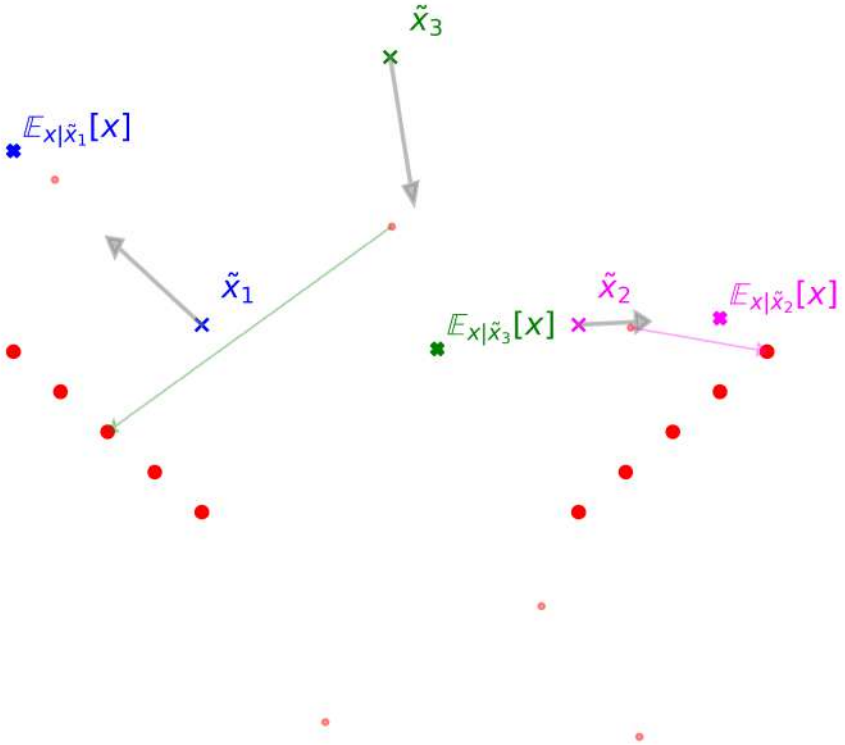
Convergence is guaranteed at infinite time. We may re-parametrize time to escalade the forward process (noise schedule).

Since the process is fully known, one can jump to any time without going through the sequential process.

$$x_{t+2dt} = (1 - dt)x_{t+dt} + \sqrt{2dt}\ z_2 = (1 - dt)[(1 - dt)x_t + \sqrt{2dt}\ z_1] + \sqrt{2dt}\ z_2$$

$$\implies x_{t+2dt} \sim \mathcal{N}((1 - 2 \cdot dt)x_t, 2 \cdot 2dtI)$$

Illustration of the denoising



credits: Ayan Das

Outline

- Why the score s_θ? ☑
- Why diffusion? (backward) ☑
- Why forward? ☑
- Why so popular? ☐

Bayesian inversion

According to *Bayes* theorem, $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t \mid y) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p(y \mid \mathbf{x}_t)$

Score of the unconditional denoising

Score of a classifier of regressor of a given property y

Generalised to many properties,

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t \mid y_1, y_2, \ldots, y_M) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \sum_{i=1}^{M} \nabla_{\mathbf{x}_t} \log p(y_i \mid \mathbf{x}_t)$$

Provided classifiers able to predict the properties based on noisy samples, the diffusion model can be used for conditional generation **without re-training**.

Ingraham, John B., et al. "Illuminating protein space with a programmable generative model." *Nature* 623.7989 (2023): 1070-1078.

Outline

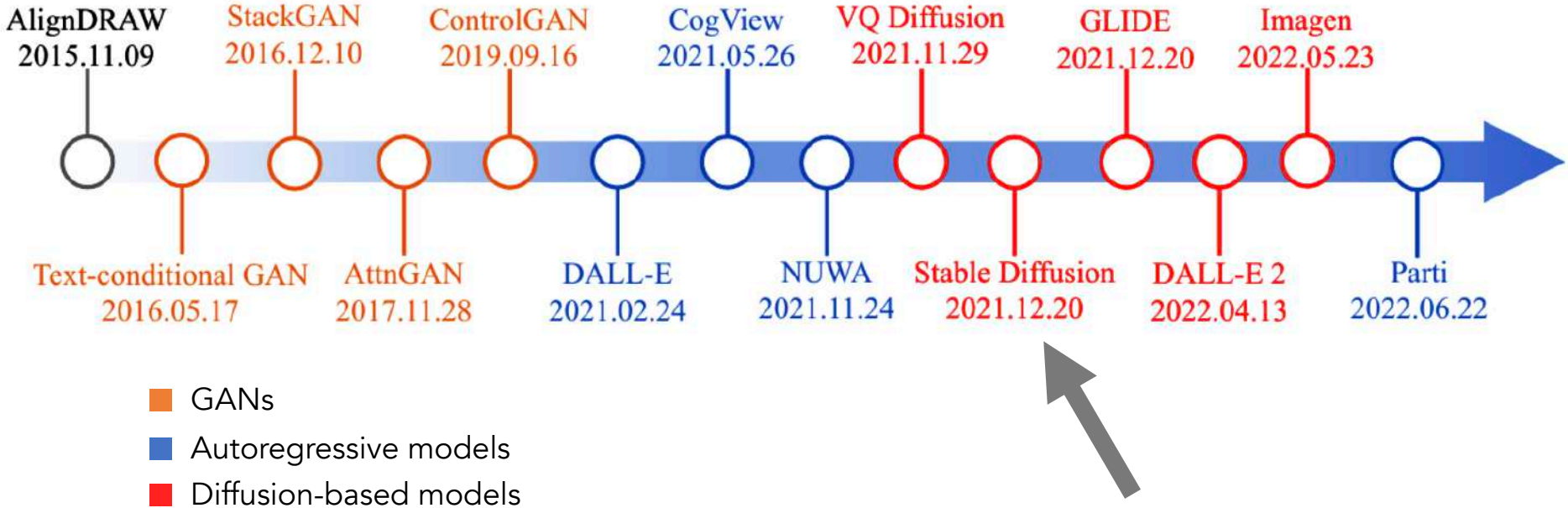
- Why the score s_θ?
- Why diffusion? (backward)
- Why forward?
- Why so popular?

Applications

DeepLearning inLifeSciences

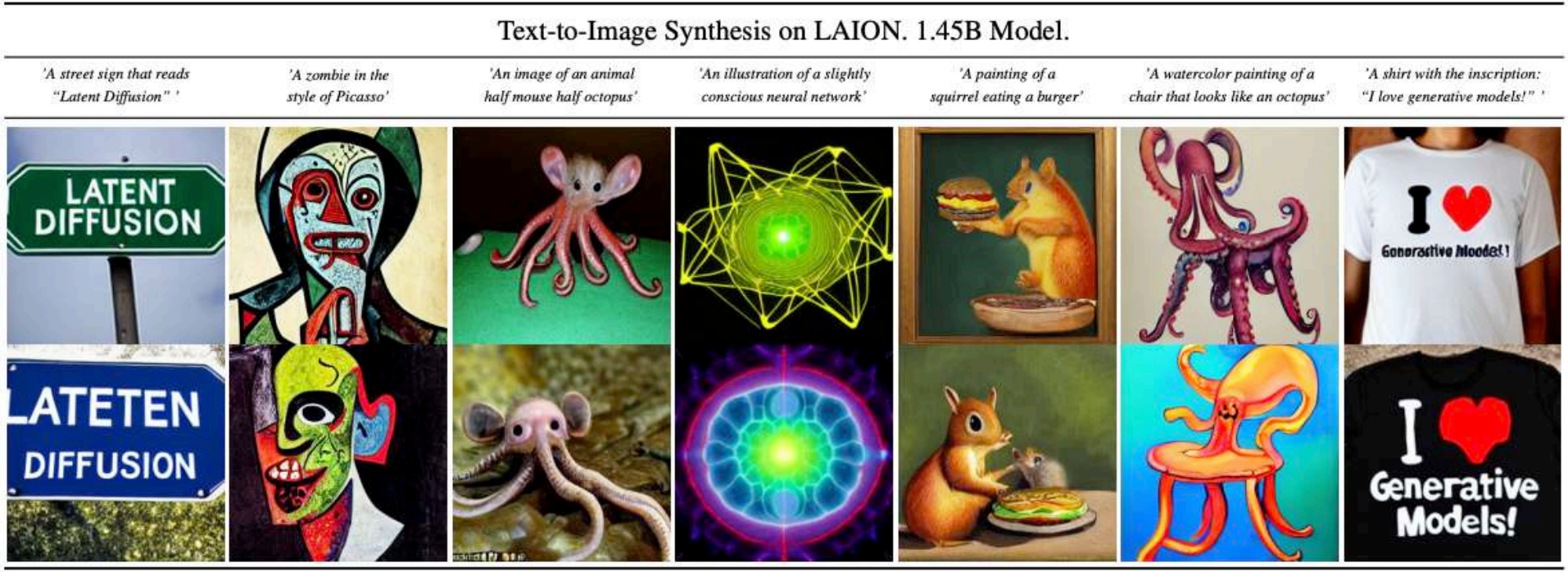
- **From text to images (Stable Diffusion)**
- Generating proteins
  - RFDiffusion
  - CHROMA
  - AlphaFold3…

Text-to-Image architectures



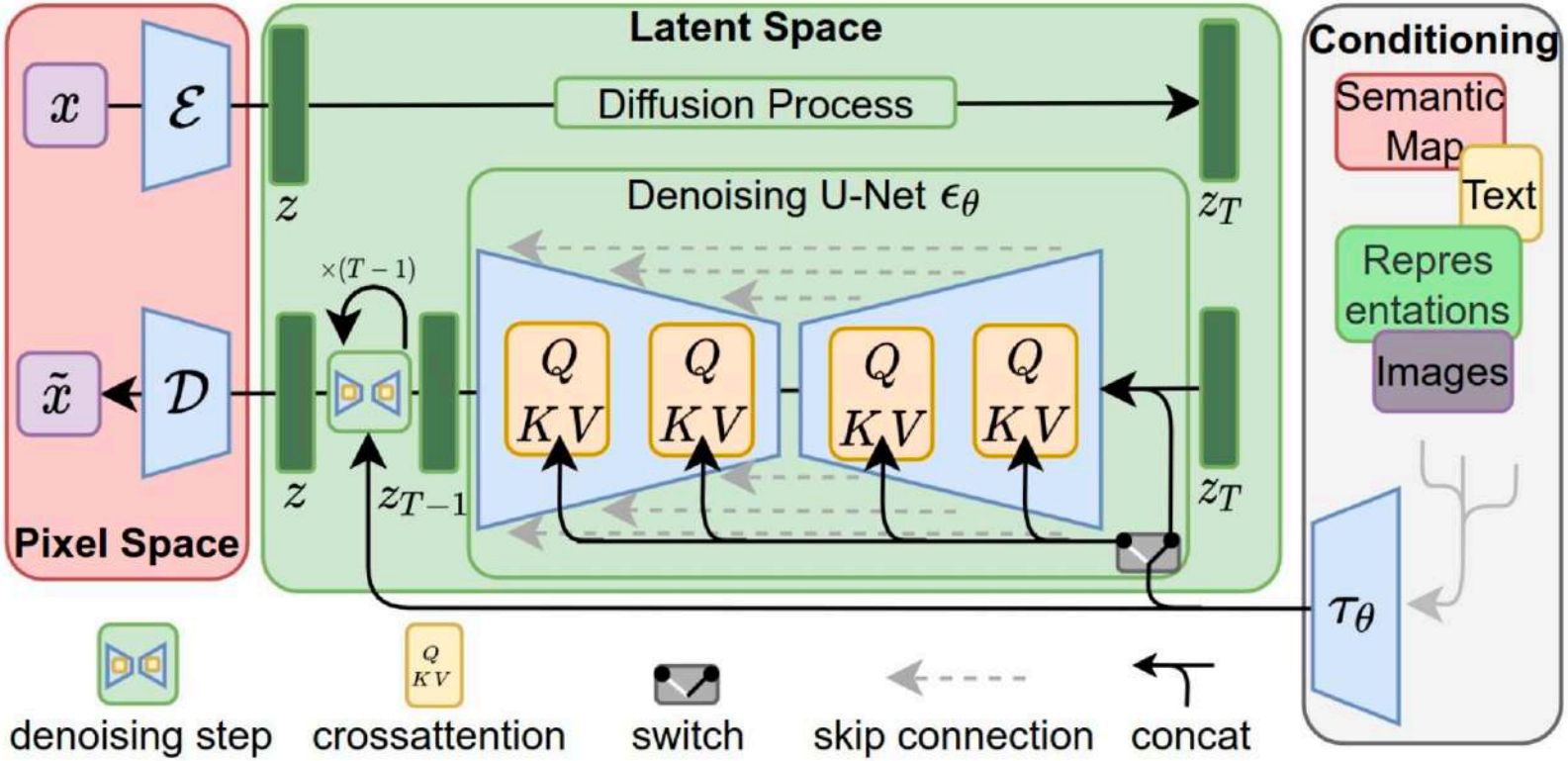
Zhang, Chenshuang, et al. "Text-to-image diffusion model in generative ai: A survey." *arXiv preprint arXiv:2303.07909* (2023).

Stable diffusion



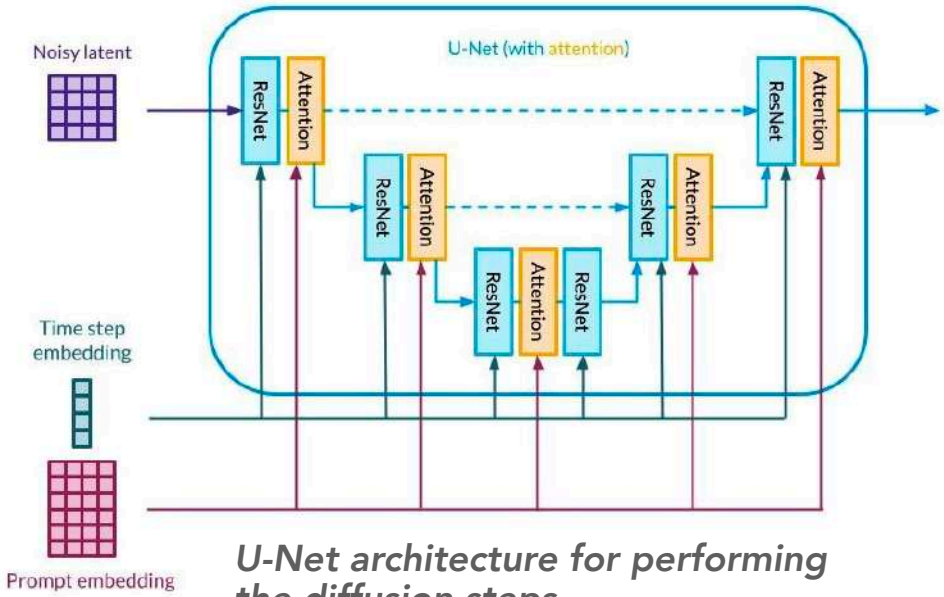
Rombach, Robin, et al. "High-resolution image synthesis with latent diffusion models." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022.

Stable diffusion



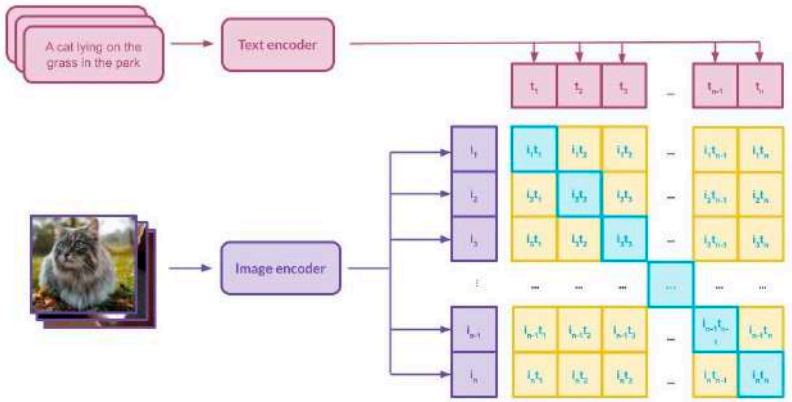
Rombach, Robin, et al. "High-resolution image synthesis with latent diffusion models." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022.

Stable diffusion



U-Net architecture for performing the diffusion steps

CLIP model for encoding the conditional prompt



$$L_{LDM} = \mathbb{E}_{\varepsilon(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\,||\,\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\,||_2^2\right]$$

credits: Jarosław Kochanowicz, Maciej Domagała, Dawid Stachowiak and Krzysztof Dziedzic

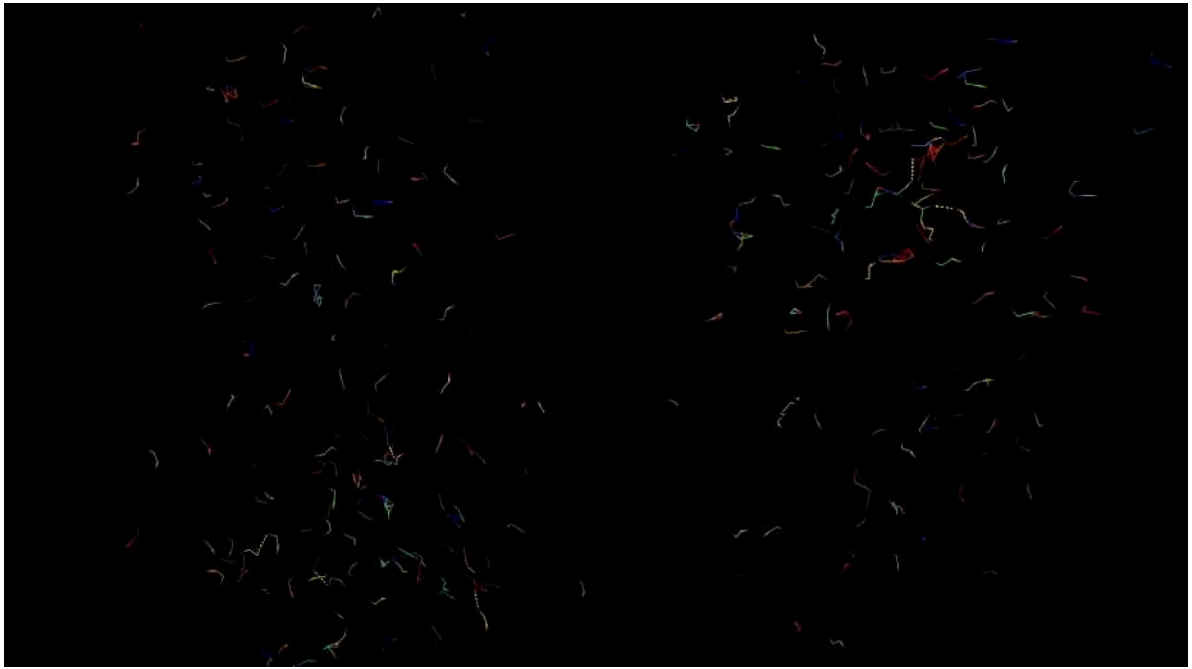
Rombach, Robin, et al. "High-resolution image synthesis with latent diffusion models." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022.

Applications

DeepLearning inLifeSciences

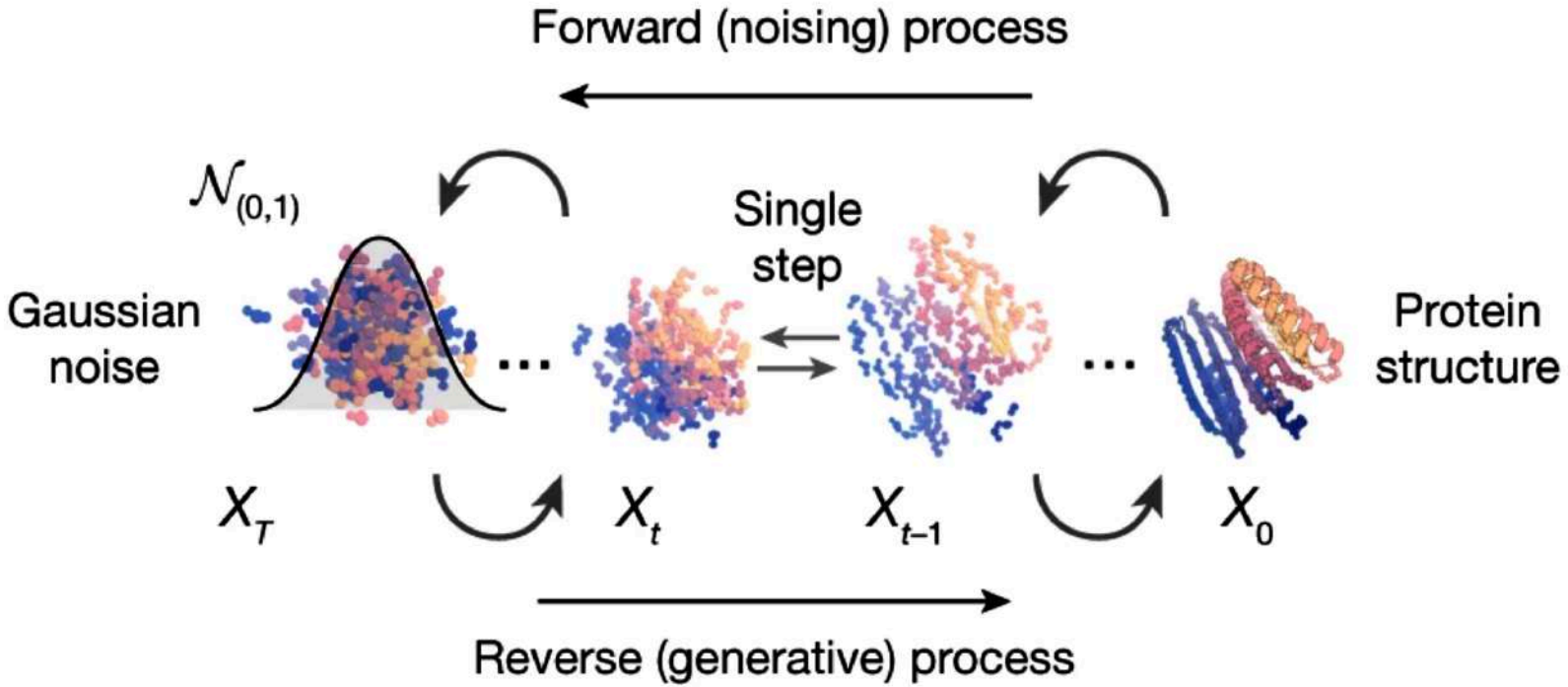
- From text to images (Stable Diffusion)
- **Generating proteins**
  - **RFDiffusion**
  - **CHROMA**
  - **AlphaFold3…**

Applications



Anand, Namrata, and Tudor Achim. "Protein structure and sequence generation with equivariant denoising diffusion probabilistic models." *arXiv preprint arXiv:2205.15019* (2022).

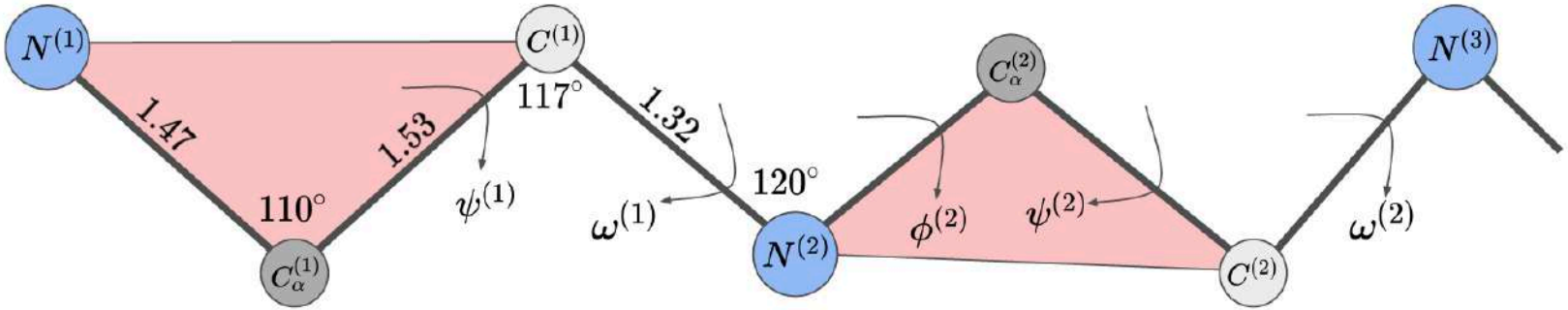
RosettaFold *Diffusion*



https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Protein backbone



Bond lengths (in Å) and angles (in degree) between neighbouring atoms are fixed.

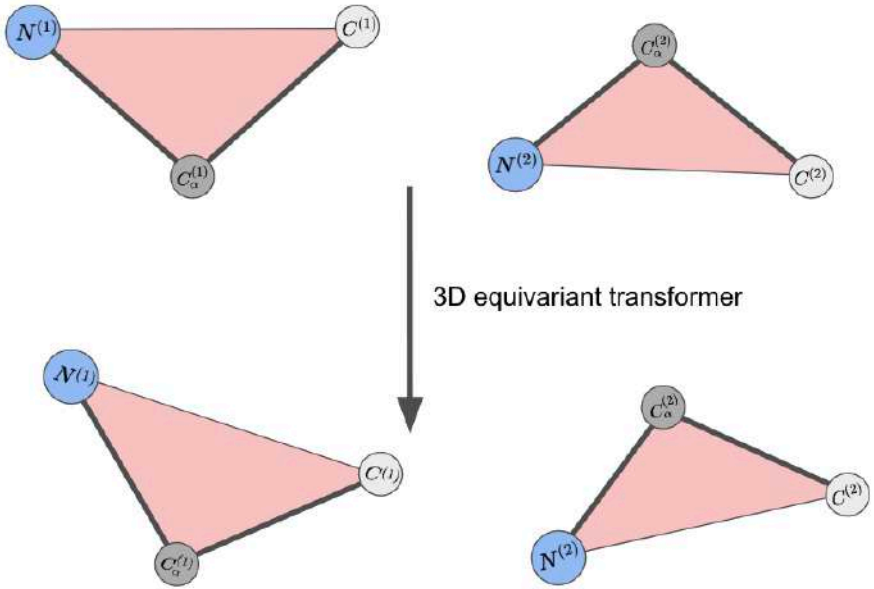
The torsion angles $\psi^{(1)}, \omega^{(1)}, \phi^{(2)}, \psi^{(2)}, \omega^{(2)}$ are the degrees of freedom.

credits: Justas. Dauparas

https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Protein backbone representation



credits: Justas. Dauparas

Euclidian transform from the **local frame** to the global reference frame

$$x_l = (r_l, z_l)$$

Coordinates of the lth residue

3x3 rotation matrix

Translation (coordinates of the Cα)

https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Independent diffusion processes

$$x_l = (r_l, z_l)$$

$$q(z^{(t)} \mid z^{(0)}) = \mathcal{N}\left(z^{(t)}; \sqrt{\bar{\alpha}^{(t)}} z^{(0)}, (1 - \bar{\alpha}^{(t)}) I_3\right)$$

$$\bar{\alpha}^{(t)} = \Pi_{s=1}^{t} \alpha^{(t)}, \quad \alpha^{(t)} = 1 - \beta^{(t)}, \quad \beta^{(1)}, \beta^{(2)}, \ldots, \beta^{(T)} \in [0,1]$$

$$p(z^{(t-1)} \mid x^{(t)}) = \mathcal{N}(z^{(t)}; \hat{\mu}(x^{(t)}), \beta^{(t)} I_3),$$

$$\text{with} \quad \hat{\mu}(x^{(t)}) = \frac{\sqrt{\bar{\alpha}^{(t-1)}} \beta^{(t)}}{1 - \bar{\alpha}^{(t)}} \hat{z}^{(0)}(x^{(t)}) + \frac{\sqrt{\alpha^{(t)}} (1 - \bar{\alpha}^{(t-1)})}{1 - \bar{\alpha}^{(t)}} z^{(t)}$$

Trippe, Brian L., et al. "Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem." *arXiv preprint arXiv:2206.04119* (2022).

Check out Julien's notebook!

https://github.com/RosettaCommons/RFdiffusion

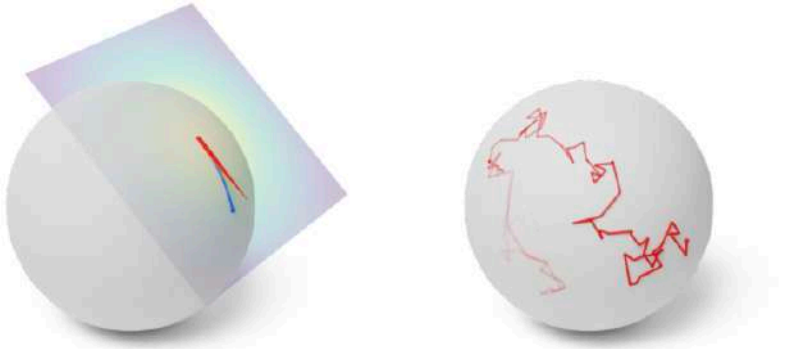
Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Independent diffusion processes

$$x_l = (r_l, z_l)$$

$$q(r^{(t)} \mid r^{(0)}) = \mathcal{IG}_{SO(3)}(r^{(t)}; r^{(0)}, \sigma_t^2)$$

Which is the Isotropic Gaussian distribution on $SO(3)$.



(a) A single step of a Geodesic Random Walk. (b) Many steps yield an approximate trajectory.

De Bortoli, Valentin, et al. "Riemannian score-based generative modelling." *Advances in Neural Information Processing Systems* 35 (2022): 2406-2422.

$$r^{(t-1)} = \exp_{r^{(t)}} \left\{ (\sigma_t^2 - \sigma_{t-1}^2) \nabla_{r^{(t)}} \log q(x^{(t)}) + \sqrt{\sigma_t^2 - \sigma_{t-1}^2} \sum_{d=1}^{3} \epsilon_d r^{(t)} f_d \right\}$$

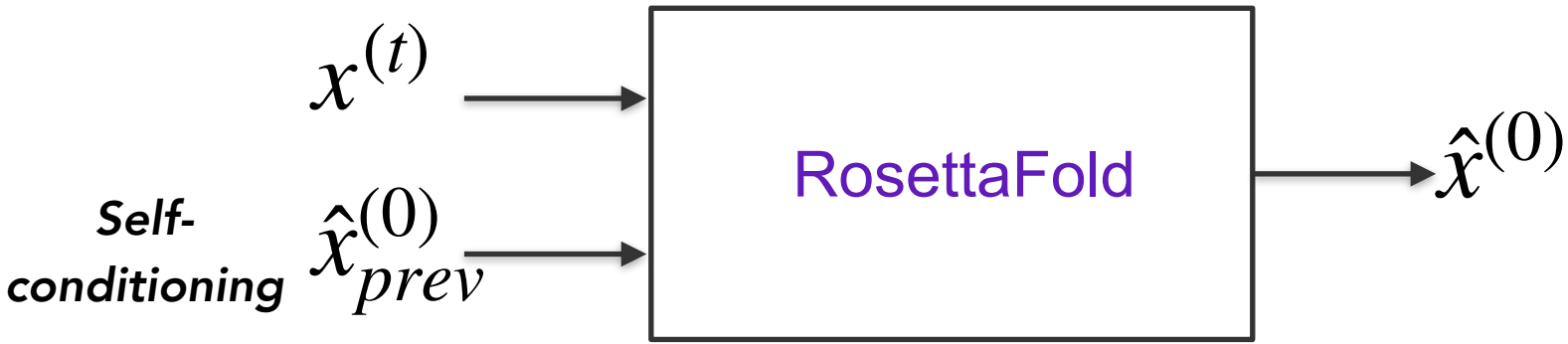
with $r^{(t)}f_1, r^{(t)}f_2, r^{(t)}f_3$ the orthonormal basis of the tangent space of $r^{(t)}$

$$\nabla_{r^{(t)}} \log q(x^{(t)}) = \nabla_{r^{(t)}} \log \mathcal{IG}_{SO(3)}(r^{(t)}; \hat{r}^{(0)}, \sigma_t^2)$$

https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Denoising structures



Loss

$$\mathrm{MSE}_{\mathrm{Frame}} = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_q[d_{\mathrm{frame}}(x^{(0)}, \hat{x}^{(0)}(x^{(t)}))^2],$$

$$d_{\mathrm{frame}}(x^{(0)}, \hat{x}^{(0)}) = \sqrt{\frac{1}{L}\sum_{l=1}^{L}\|z_l^{(0)} - \hat{z}_l^{(0)}\|_2^2 + \|I_3 - \hat{r}_l^{(0)\top} r_l^{(0)}\|_F^2},$$

https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Training procedure

Algorithm 3 RFdiffusion Training

function FORWARDNOISE($x^{(0)}, t$)
  for $l = 1, \ldots, L$ **do**
    $(r_l^{(0)}, z_l^{(0)}) = x_l^{(0)}$
    $z_l^{(t)} \sim \mathcal{N}\left(\sqrt{\bar{\alpha}^{(t)}} z_l^{(0)}, (1 - \bar{\alpha}^{(t)}) I_3\right)$
    $r_l^{(t)} \sim \mathcal{IG}_{SO(3)}\left(r_l^{(0)}, \sigma_t^2\right)$
    $x_l^{(t)} = (r_l^{(t)}, z_l^{(t)})$
  end for
return $x^{(t)}$
end function

function TRAIN
  while not converged **do**
    $x^{(0)} \sim$ TrainingSet
    $t \sim$ Uniform($\{1, \ldots, T\}$)
    if Uniform$(0, 1.0) < 0.5$ **or** $t = T$ **then**
      ▷ Train step without self-conditioning
      $x^{(t)} = \text{ForwardNoise}(x^{(0)}, t)$
      $\hat{x}^{(0)}_{\text{prev.}} = \vec{0}$
    else
      ▷ Train step with self-conditioning
      $x^{(t+1)} = \text{ForwardNoise}(x^{(0)}, t+1)$
      $x^{(t)} = \text{ReverseStep}(x^{(t+1)}, x^{(0)})$

      ▷ Compute self-conditioning input
      $\hat{x}^{(0)}_{\text{prev.}} = RFdiffusion(x^{(t+1)}, \vec{0})$
      $\hat{x}^{(0)}_{\text{prev.}} = \text{StopGradient}(\hat{x}^{(0)}_{\text{prev.}})$
    end if
    $\hat{x}^{(0)} = RFdiffusion(x^{(t)}, \hat{x}^{(0)}_{\text{prev.}})$
    Take gradient step on $d_{\text{frame}}\left(x^{(0)}, \hat{x}^{(0)}\right)^2$
  end while
end function

https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Inference

Algorithm 2 RFdiffusion generation

function SAMPLEREFERENCE(L)
  ▷ Random initial structure for L residues
  for $l = 1, \ldots, L$ **do**
    $r_l^{(T)} \sim \text{Uniform}(SO(3))$
    $z_l^{(T)} \sim \mathcal{N}(0, I_3)$
    $x_l^{(T)} = (r_l^{(T)}, x_l^{(T)})$
  end for
return $x^{(T)}$
end function

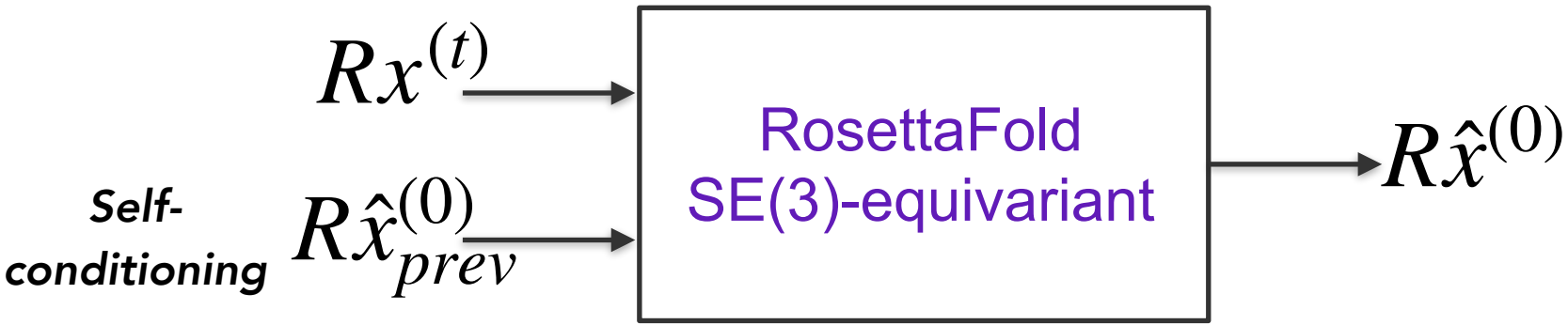
function SAMPLE(L)
  ▷ RFdiffusion generation of L-residue backbone structure
  $x^{(T)} = \text{SampleReference}(L)$
  $\hat{x}^{(0)}_{\text{prev.}} = \vec{0}$
  for $t = T, \ldots, 1$ **do**
    $\hat{x}^{(0)} = \text{RFDIFFUSION}(x^{(t)}, \hat{x}^{(0)}_{\text{prev.}})$
    $x^{(t-1)} = \text{REVERSESTEP}(x^{(t)}, \hat{x}^{(0)})$
    $\hat{x}^{(0)}_{\text{prev.}} = \hat{x}^{(0)}$
  end for
return $\hat{x}^{(0)}$
end function

function REVERSESTEP($x^{(t)}, \hat{x}^{(0)}$)
  ▷ One step of reverse diffusion
  for $l = 1, \ldots, L$ **do**
    $(r_l^{(t)}, z_l^{(t)}) = x_l^{(t)}$
    $(\hat{r}_l^{(0)}, \hat{z}_l^{(0)}) = \hat{x}_l^{(0)}$
    ▷ Update translations
    $z_l^{(t-1)} \sim \mathcal{N}(\frac{\sqrt{\bar{\alpha}^{(t-1)}}\beta^{(t)}}{1-\bar{\alpha}^{(t)}}\hat{z}_l^{(0)} + \frac{\sqrt{\alpha^{(t)}}(1-\bar{\alpha}^{(t-1)})}{1-\bar{\alpha}^{(t)}} z_l^{(t)}, \beta^{(t)} I_3)$

    ▷ Update rotations
    $s_l = \text{ROTATIONSCOREAPPROXIMATION}(r_l^{(t)}, \hat{r}_l^{(0)}, \sigma_t^2)$
    $\epsilon_{l,1}, \epsilon_{l,2}, \epsilon_{l,3} \overset{iid}{\sim} \mathcal{N}(0,1)$
    $r_l^{(t-1)} = r_l^{(t)} \exp_{I_3}\left\{(\sigma_t^2 - \sigma_{t-1}^2) r_l^{(t)\top} s_l + \sqrt{\sigma_t^2 - \sigma_{t-1}^2} \sum_{d=1}^{3} \epsilon_{l,d} f_d\right\}$
    $x_l^{(t-1)} = (r_l^{(t-1)}, z_l^{(t-1)})$
  end for
return $x^{(t-1)}$
end function

https://github.com/RosettaCommons/RFdiffusion
Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Invariance & equivariance in 3D



Loss Depends on the choice of the global reference frame!

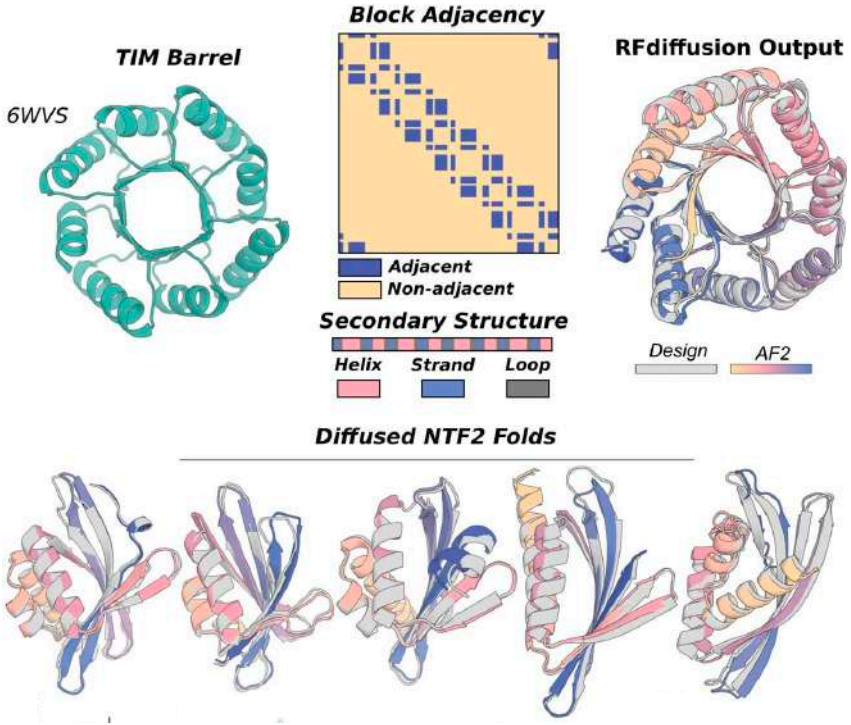
$$\mathrm{MSE}_{\mathrm{Frame}} = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_q[d_{\mathrm{frame}}(x^{(0)}, \hat{x}^{(0)}(x^{(t)}))^2],$$

$$d_{\mathrm{frame}}(x^{(0)}, \hat{x}^{(0)}) = \sqrt{\frac{1}{L}\sum_{l=1}^{L}\|z_l^{(0)} - \hat{z}_l^{(0)}\|_2^2 + \|I_3 - \hat{r}_l^{(0)\top} r_l^{(0)}\|_F^2},$$

https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Fold-specific generation



RF*Diffusion* is fine-tuned with extended input features specifying

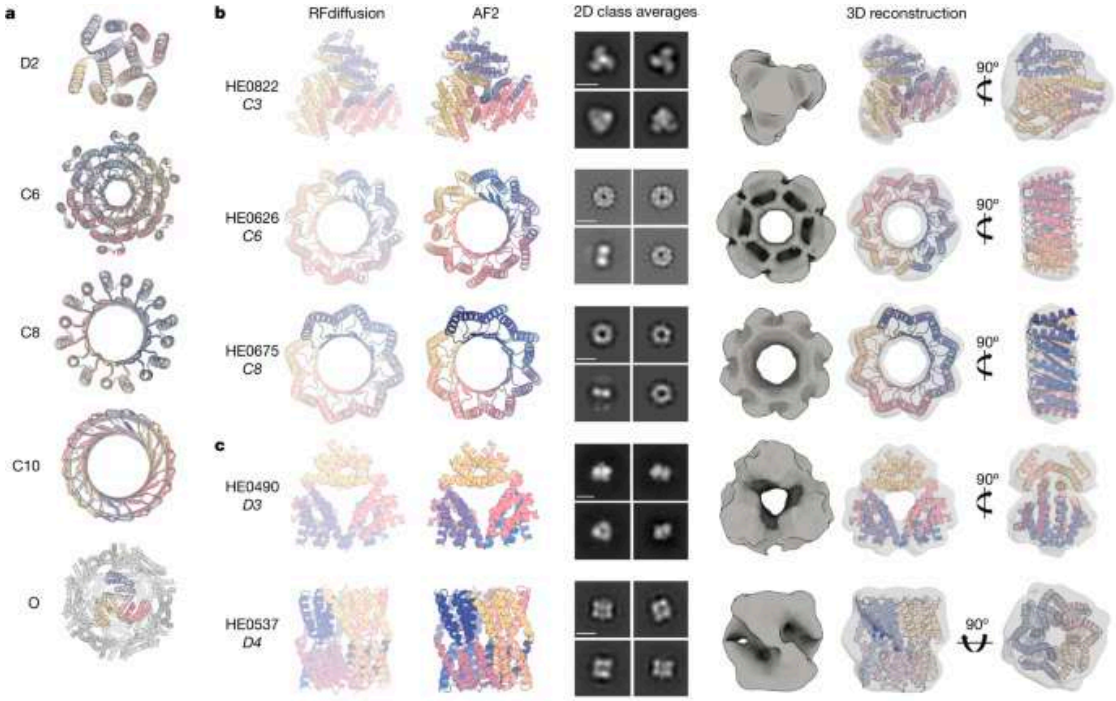
- the secondary structure type of each residue,
- the adjacency matrix of the secondary structure blocks.

AlphaFold (AF2) is used as external validation of the plausibility or realism of the produced structures. The designs are diverse and closely recapitulated by AF2.

https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

Symmetric oligomers



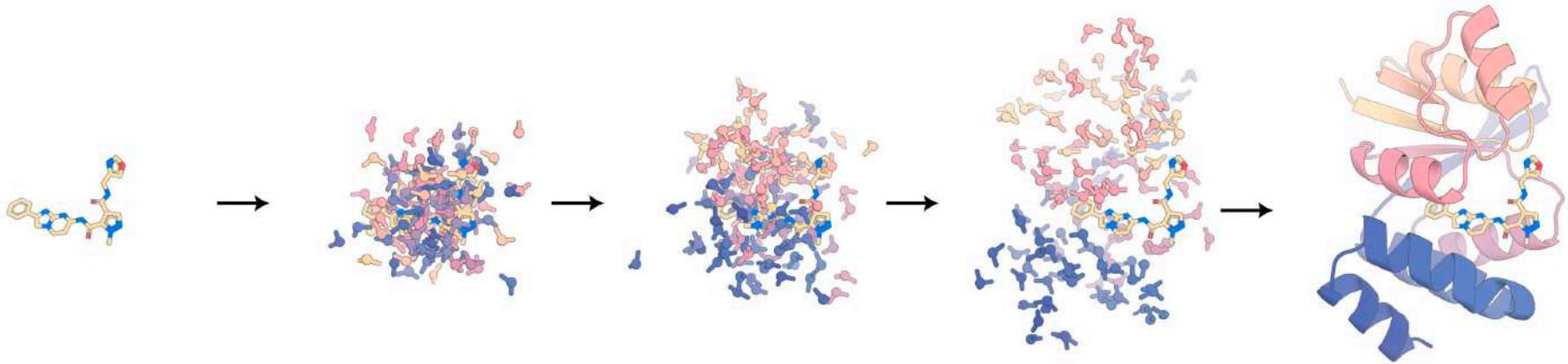
RF*Diffusion* is symmetry preserving thanks to its rotation and permutation invariance.

Explicit symmetrisation is still applied at each demonising step.

https://github.com/RosettaCommons/RFdiffusion

Watson, Joseph L., et al. "De novo design of protein structure and function with RFdiffusion." *Nature* 620.7976 (2023): 1089-1100.

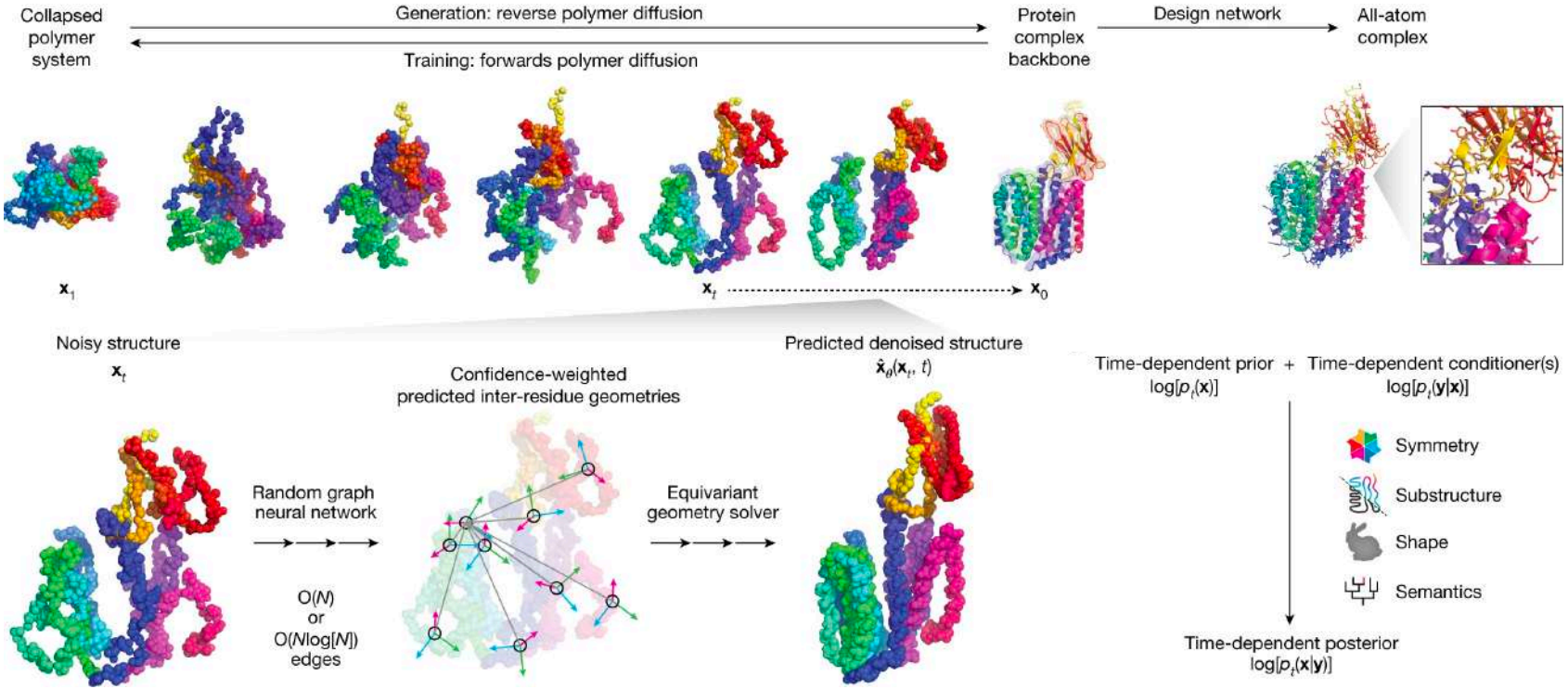
Small molecule binder design



RF*Diffusion All-Atom* is explicitly trained in a conditioned way by demonising proteins bound to small molecules. The model is additionally trained to generated scaffolds conditionally on fixed motifs (with or without small molecules) by setting the time to zero for the motifs and maintaining them in original version (without any noise).

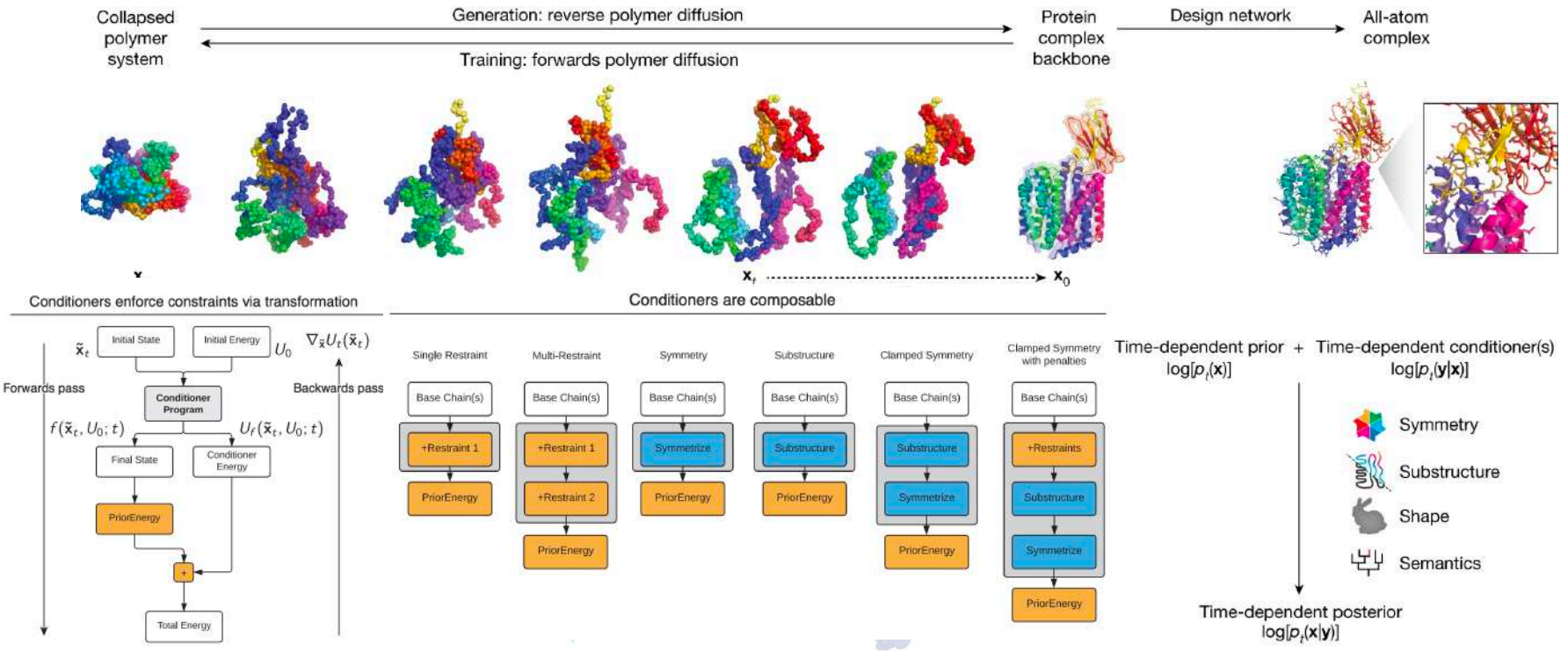
Krishna, Rohith, et al. "Generalized biomolecular modeling and design with RoseTTAFold All-Atom." *Science* (2024): eadl2528.

CHROMA



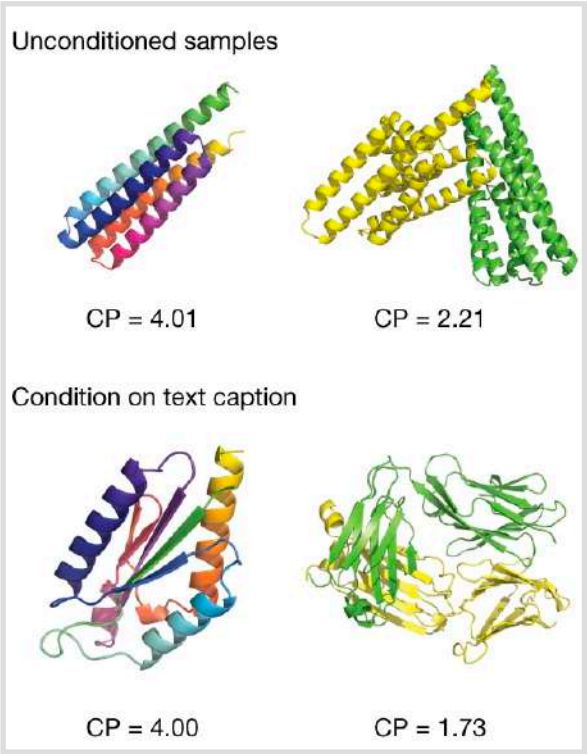
Ingraham, John B., et al. "Illuminating protein space with a programmable generative model." *Nature* 623.7989 (2023): 1070-1078.

CHROMA



Ingraham, John B., et al. "Illuminating protein space with a programmable generative model." *Nature* 623.7989 (2023): 1070-1078.

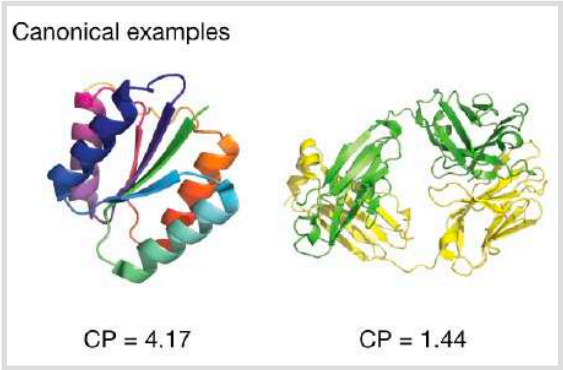
From text to protein structures



Left: *crystal structure of a Rossmann fold*

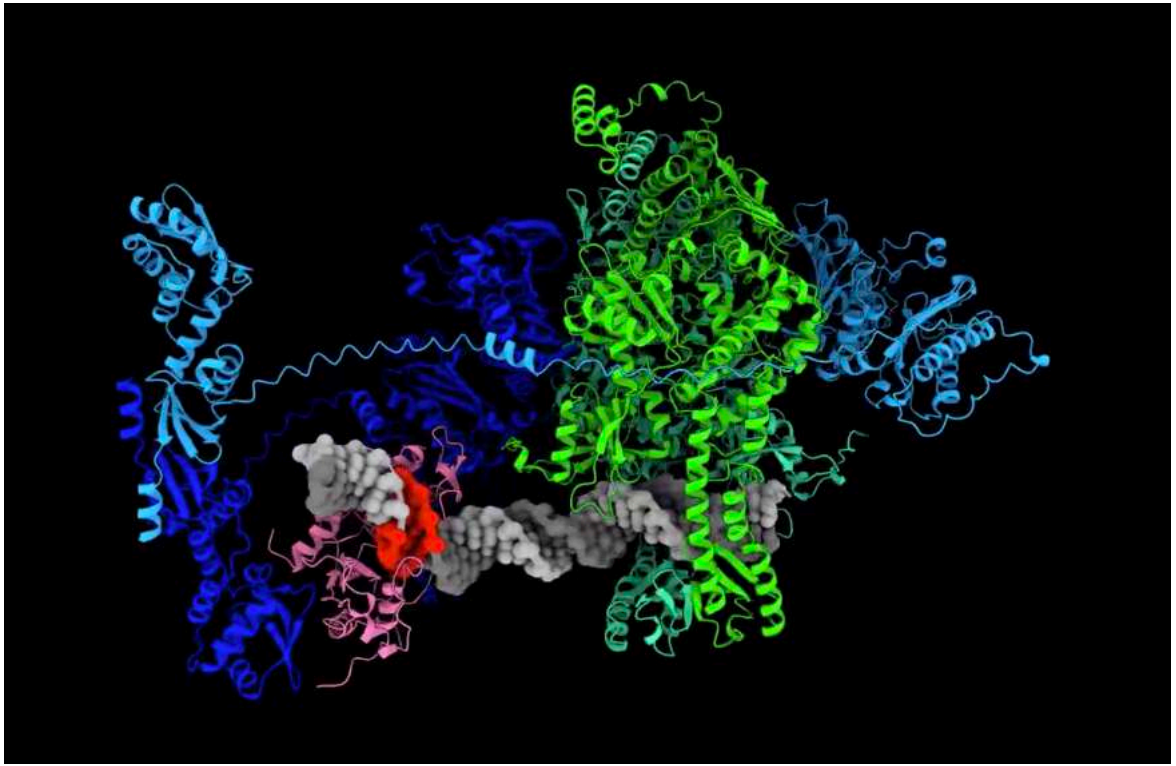
Right: *crystal structure of a Fab antibody fragment*

Fine-tuning a multi-label predictor to bias a pretrained large language model into a structure caption predictor can enable natural language conditioning.



Ingraham, John B., et al. "Illuminating protein space with a programmable generative model." *Nature* 623.7989 (2023): 1070-1078.

AlphaFold 3



By Jan Kosinski:

MutS protein (green) recognizes base mismatches in DNA (gray), after which it slides off to find MutL matchmaker protein (blue), which positions and activates MutH nuclease (pink) to nick a hemimethylated GATC sequence (red) to activate removal and repair of the mismatched strand.

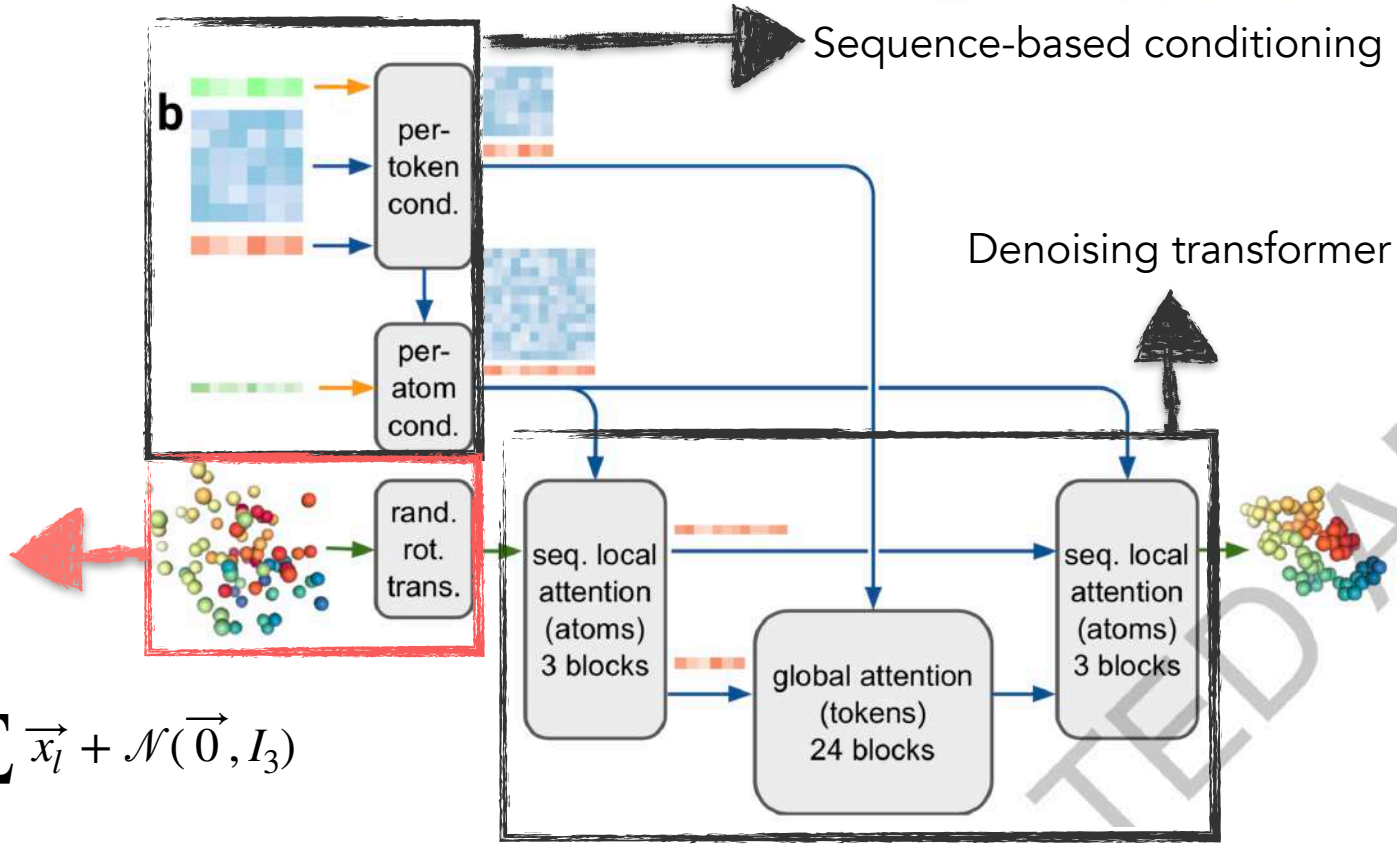
Abramson, Josh, et al. "Accurate structure prediction of biomolecular interactions with AlphaFold 3." *Nature* (2024): 1-3.

AlphaFold 3 diffusion module

No more frames! The network operates directly on raw coordinates.

Data augmentation: get 48 samples from 1 through random rot. & trans.

$$\vec{x}_{l\,c}^{\,aug} = R \cdot \vec{x}_l - \frac{1}{L}\sum_l \vec{x}_l + \mathcal{N}(\vec{0}, I_3)$$



Abramson, Josh, et al. "Accurate structure prediction of biomolecular interactions with AlphaFold 3." *Nature* (2024): 1-3.

AlphaFold 3 diffusion process

DeepLearning inLifeSciences

Algorithm 18 Sample Diffusion

def SampleDiffusion($\{\mathbf{f}^*\}, \{\mathbf{s}_i^{\text{inputs}}\}, \{\mathbf{s}_i^{\text{trunk}}\}, \{\mathbf{z}_{ij}^{\text{trunk}}\}$, Noise Schedule $[c_0, c_1, \ldots, c_T]$, $\gamma_0 = 0.8$, $\gamma_{\min} = 1.0$, noise scale $\lambda = 1.003$, step scale $\eta = 1.5$):

1: $\vec{\mathbf{x}}_l \sim c_0 \cdot \mathcal{N}(\vec{\mathbf{0}}, \mathbf{I}_3)$ ⟹ *start from pure noise* $\vec{\mathbf{x}}_l \in \mathbb{R}^3$

2: **for all** $c_\tau \in [c_1, \ldots, c_T]$ **do** ⟹ *go over noise scheduler*

3: $\{\vec{\mathbf{x}}_l\} \leftarrow \text{CentreRandomAugmentation}(\{\vec{\mathbf{x}}_l\})$ ⟹ *augment the data (for batch size increase)*

4: $\gamma = \gamma_0$ if $c_\tau > \gamma_{\min}$ else 0

5: $\hat{t} = c_{\tau-1}(\gamma + 1)$ } ⟹ *determine the amount of noise*

6: $\vec{\xi}_l = \lambda\sqrt{\hat{t}^2 - c_{\tau-1}^2} \cdot \mathcal{N}(\vec{\mathbf{0}}, \mathbf{I}_3)$ ⟹ *compute the noise vector* $\vec{\xi}_l \in \mathbb{R}^3$

7: $\vec{\mathbf{x}}_l^{\text{noisy}} = \vec{\mathbf{x}}_l + \vec{\xi}_l$ ⟹ *corrupt the coordinates with the noise vector*

8: $\{\vec{\mathbf{x}}_l^{\text{denoised}}\} = \text{DiffusionModule}(\{\vec{\mathbf{x}}_l^{\text{noisy}}\}, \hat{t}, \{\mathbf{f}^*\}, \{\mathbf{s}_i^{\text{inputs}}\}, \{\mathbf{s}_i^{\text{trunk}}\}, \{\mathbf{z}_{ij}^{\text{trunk}}\})$ ⟹ *denoise conditionally on the sequence*

9: $\vec{\delta}_l = (\vec{\mathbf{x}}_l - \vec{\mathbf{x}}_l^{\text{denoised}})/\hat{t}$ ⟹ *compute the denoising gradient*

10: $dt = c_\tau - \hat{t}$

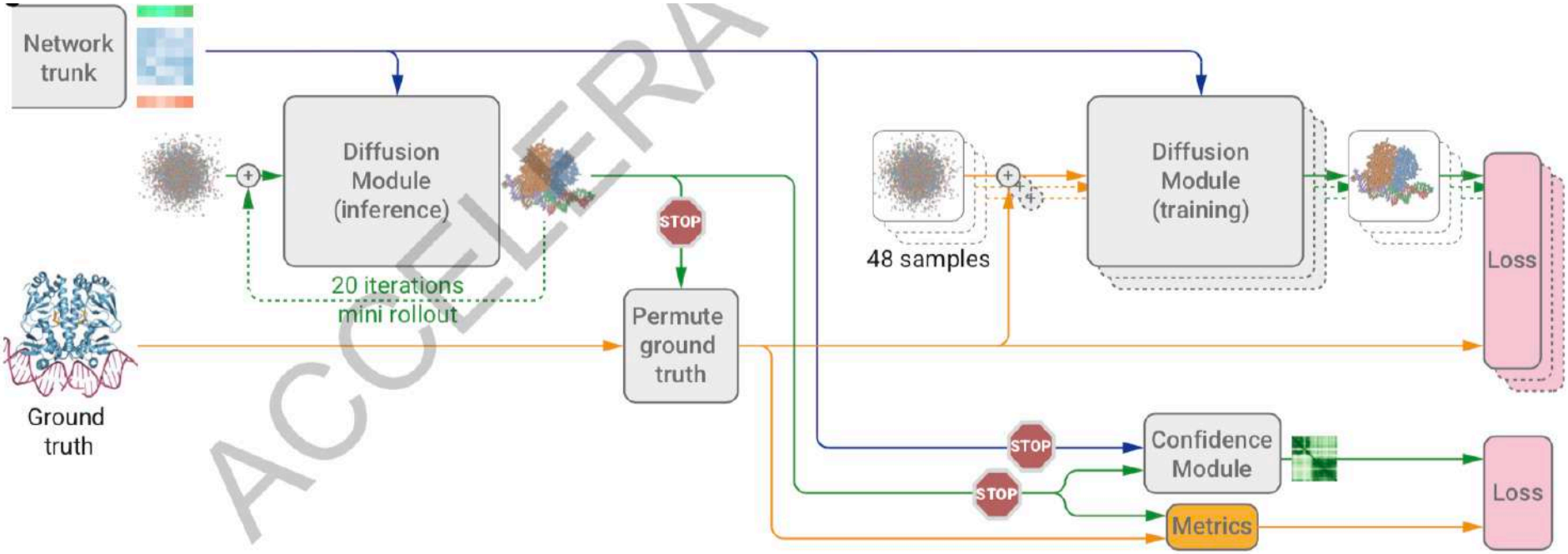
11: $\vec{\mathbf{x}}_l \leftarrow \vec{\mathbf{x}}_l^{\text{noisy}} + \eta \cdot dt \cdot \vec{\delta}_l$ ⟹ *update the coordinates from their noisy version in the denoising gradient direction*

12: **end for**

13: **return** $\{\vec{\mathbf{x}}_l\}$

Abramson, Josh, et al. "Accurate structure prediction of biomolecular interactions with AlphaFold 3." *Nature* (2024): 1-3.

AlphaFold 3 training and inference



Abramson, Josh, et al. "Accurate structure prediction of biomolecular interactions with AlphaFold 3." *Nature* (2024): 1-3.

Taking another perspective

DeepLearning inLifeSciences

- Do we really need Langevin dynamics?
- What if we do not reach pure noise?
- What is we want to sample from another distribution?

Matching flows instead of scores

DeepLearning inLifeSciences

Flow Matching

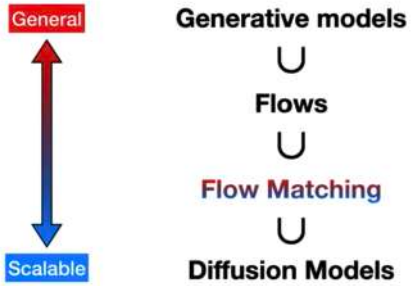
Simplifying and Generalizing Diffusion Models

Yaron Lipman

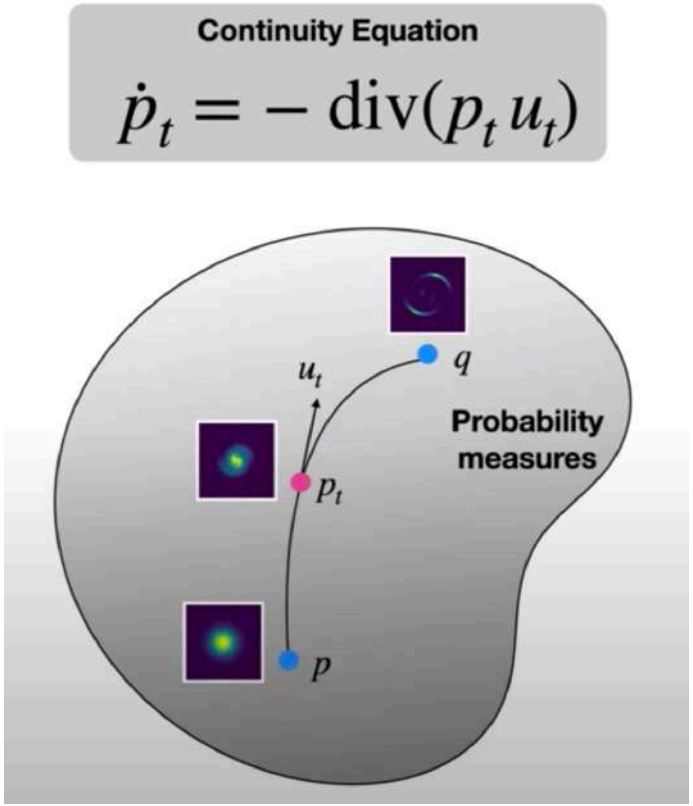
https://www.youtube.com/watch?v=5ZSwYogAxYg

Meta

מכון ויצמן למדע
WEIZMANN INSTITUTE OF SCIENCE



Matching flows instead of scores



p_t lies on a time-evolving probability path from
- p: prior probability density (e.g. noise), to
- q: target probability density (the true density of the data)

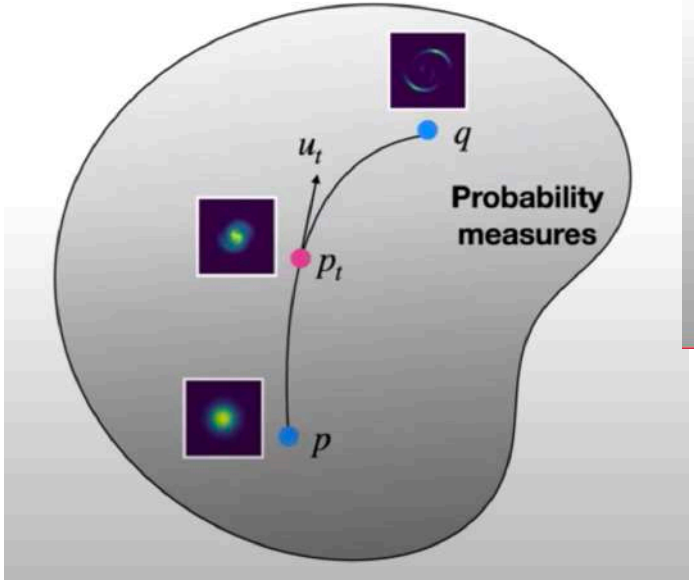
u_t is the time-dependent vector field $[0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ defining the dynamics along the path, according to the Ordinary Differential Equation: $\dot{x} = u_t(x_t)$

The **continuity equation** states that the change of probability density $\dot{p}_t$ (e.g. seen as a fluid density) at any time point t is exactly counter-balanced by the change of flux volume in all directions $div(p_t u_t)$.

Compare velocities

Continuity Equation

$$\dot{p}_t = -\operatorname{div}(p_t u_t)$$



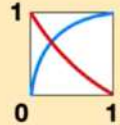
$$L_{\text{FM}}(q\|p_1) = \min \mathbb{E}_{t,\,p_t(x)} \|v_t(x) - u_t(x)\|^2$$

The FM loss (w. marginal vector field) is equivalent to the ***conditional*** *FM loss where we break down to one training sample at a time.*

$$L_{\text{CFM}}(q\|p_1) = \min \mathbb{E}_{t,\,q(x_1),\,p_t(x|x_1)} \|v_t(x) - u_t(x\,|\,x_1)\|^2$$

$$= \min \mathbb{E}_{t,\,q(x_1),\,p(x_0)} \|v_t(x_t) - \dot{x}_t\|^2$$

$x_t \sim p_t(x\,|\,x_1)$
$\dot{x}_t = u_t(x_t\,|\,x_1)$



$$x_t = \Psi_t(x_0\,|\,x_1) = \sigma_t x_0 + \alpha_t x_1$$

$$x_0 \sim p \Rightarrow x_t \propto p\left(\frac{x - \alpha_t x_1}{\sigma_t}\right)$$

A less restrictive framework

DeepLearning inLifeSciences

Algorithm 1: Flow Matching training.

Input : dataset q, noise p
Initialize v^θ
while *not converged* **do**
  $t \sim \mathcal{U}([0,1])$ ▷ sample time
  $x_1 \sim q(x_1)$ ▷ sample data
  $x_0 \sim p(x_0)$ ▷ sample noise
  $x_t = \Psi_t(x_0|x_1)$ ▷ conditional flow
  Gradient step with $\nabla_\theta \|v_t^\theta(x_t) - \dot{x}_t\|^2$
Output: v^θ

Algorithm 2: Diffusion training.

Input : dataset q, noise p
Initialize s^θ
while *not converged* **do**
  $t \sim \mathcal{U}([0,1])$ ▷ sample time
  $x_1 \sim q(x_1)$ ▷ sample data
  $x_t = p_t(x_t|x_1)$ ▷ sample conditional prob
  Gradient step with
  $\nabla_\theta \|s_t^\theta(x_t) - \nabla_{x_t} \log p_t(x_t|x_1)\|^2$
Output: v^θ

$p_t(x_t|x_1)$ general
$p(x_0)$ is general

$p_t(x_t|x_1)$ closed-form from of SDE $dx_t = f_t dt + g_t dw$

- *Variance Exploding*: $p_t(x|x_1) = \mathcal{N}(x|x_1, \sigma_{1-t}^2 I)$
- *Variance Preserving*: $p_t(x|x_1) = \mathcal{N}(x|\alpha_{1-t}x_1, (1-\alpha_{1-t}^2)I)$
  $\alpha_t = e^{-\frac{1}{2}T(t)}$

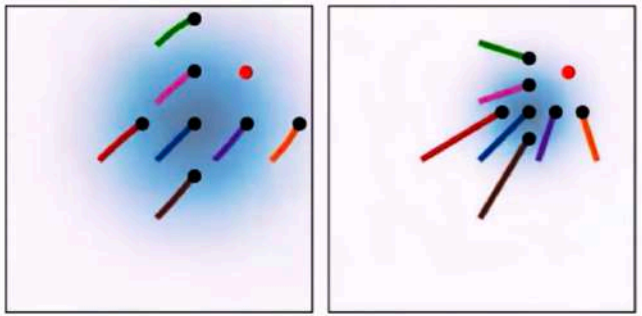
$p(x_0)$ is Gaussian
$p_0(\cdot|x_1) \approx p$

A less restrictive framework

- Example of non-diffusion choice

$$p_t(x \mid x_1), u_t(x \mid x_1) \quad \Leftrightarrow \quad \Psi_t(x_0 \mid x_1)$$

$$\mathcal{N}(tx_1, (1-t)^2 I), \frac{x_1 - x}{1-t} \qquad \Psi_t(x_0 \mid x_1) = (1-t)x_0 + tx_1$$



Diffusion (VP) Cond-OT

Take-home messages

- Diffusion models are generative models drawing inspiration from **Langevin** dynamics.
- They allow for **transforming noise** into any distribution we want to sample from. They are very flexible, allowing for conditioning on various properties.
- The learnable part of the process serves for **denoising** (estimating noise is equivalent to estimating the score) and for **conditioning** (with labels).
- Convergence to pure noise is critical, as well as defining an adequate **noise scheduler**.
- Some of the restrictions of diffusion models are alleviated by **flow-matching** models.

Resources

- https://iclr-blogposts.github.io/2024/blog/diffusion-theory-from-scratch/
- https://deepsense.ai/diffusion-models-in-practice-part-1-the-tools-of-the-trade/
- https://lilianweng.github.io/posts/2021-07-11-diffusion-models/
- https://yang-song.net/blog/2021/score/
- https://www.youtube.com/watch?v=a4Yfz2FxXiY&themeRefresh=1
- https://github.com/diff-usion/Awesome-Diffusion-Models?tab=readme-ov-file
- https://dauparas.github.io/post/af2/
- Thanks to Julien Nguyen Van, Roman Klypa and Sergei Grudinin for discussions on the topic and sharing of resources.

What now?

- Check out the Practical! [Made by Julien Nguyen Van]