Today: Diffusion Models (Last Lecture!)

Readings in Textbook: Ch 18 (Diffusion)
arxiv 2406.08929 from June 2024 (first sections only)

Announce:
Final Project Report: Extended Sun Dec 7
Poster Session Tue Dec 9th
Final Peer Review Thu Dec 11th
Final Final Report Sun Dec 15th
Final Exam: Wed Dec 17th 8-11AM
No Notes. Just Pencil.

Also: EECS Colloquium Next Week
Ece Kamar MSR
Dec 10th 4-5 PM

Diffusion Models for Generation... "Test-time compute"

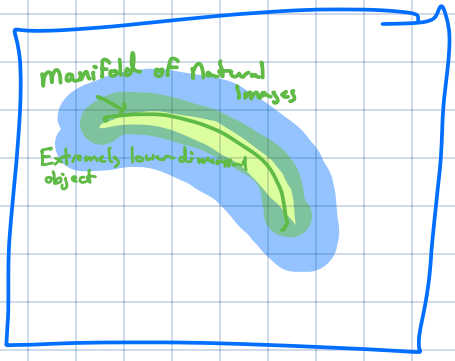
Recall from Lecture 18:

Ideas that don't work

A) Use a classifier

Try: Random Uniform Image followed by Gradient Ascent on the Cat Score

Result: Noise-like image that classifier confidently classifies as a cat.



Bring insights from VAE story: Need to see noisy things during training

Figure adapted from South Park to parody/meme

Why not everything in between too?

Can sample from new natural images by adding noise.

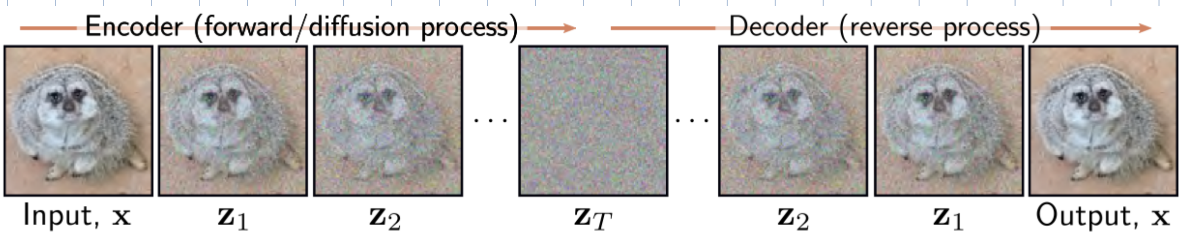
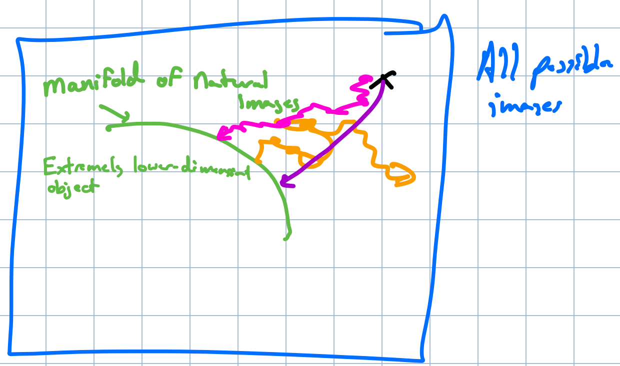


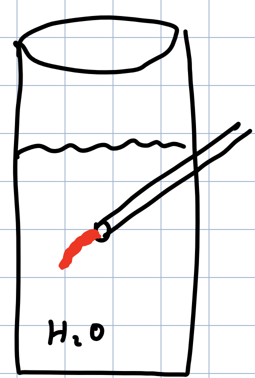
Fig 18.1 in Prince

$$
\begin{aligned}
\mathbf{z}_1 &= \sqrt{1-\beta_1}\cdot\mathbf{x} + \sqrt{\beta_1}\cdot\boldsymbol{\epsilon}_1 \\
\mathbf{z}_t &= \sqrt{1-\beta_t}\cdot\mathbf{z}_{t-1} + \sqrt{\beta_t}\cdot\boldsymbol{\epsilon}_t \qquad \forall\, t\in 2,\ldots,T,
\end{aligned}
$$

Traditional Perspective from Sohl-Dickstein, et. al.'s "Deep Unsupervised learning using Non-equilibrium Thermodynamics" in ICML 2015.

For clarity, we'll mostly follow arxiv 2406.08929 from June 2024: Nakkiran, et. al. "Step-by-step Diffusion: An elementary tutorial"





Idea: Take sample paths of forward diffusion starting at $\{x_{train,i}\}$

$$x_0 \to x_1 \to x_2 \to x_3 \to \cdots\cdots \; x_{T-1} \to x_T$$

Train a network to predict backwards in time $x_{t+1} \to \hat{x}_t$

Somehow use this learned network to simulate a path backwards from x_T to x_0.

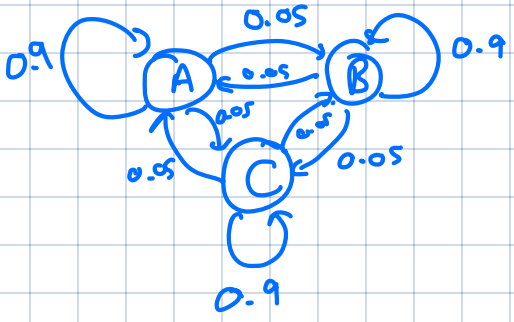
Key Design Question: Do I need randomness on the backward path

Two approaches: stochastic sampling (DDPM)

deterministic sampling (DDIM - next time)

Stochastic Case → Simulate Reverse Process

For Intuition. Consider discrete-time discrete-state



Stand $[\frac{1}{2}, 0, \frac{1}{2}]$ → $[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$
P(A) P(B) P(C)

(Assume $X_t \to X_{t+1}$ is conditionally Gaussian. e.g. $X_{t+1} = X_t + N_t$, $N_t \sim N(0, \sigma^2 I)$)

Is $p(X_{t-1} | X_t)$ like a Gaussian? Yes!

As long as density for x_0 sufficiently smooth, if σ^2 small enough,

$$p(x_{t-1} | x_t = \vec{z}) \approx N(\vec{\mu}_{t-1}(\vec{z}), \sigma^2 I)$$

Need to learn this from data.

Before we argue why (HW has you do it for a special case precisely)

Let's take a continuous-time perspective. $T \to 1$. Δt small step.

$$x_0 \to x_{\Delta t} \to x_{2\Delta t} \to x_{3\Delta t} \to \cdots \cdots x_{1-\Delta t} \to x_1$$

Noise added in one step is $N(0, \sigma^2 \Delta t)$ — very small

Big Tiny

To do generation... First draw $\hat{x}_1 \sim N(0, \sigma^2)$

Want to get $\hat{x}_{t-\Delta t}$ given $\hat{x}_t$ going backwards

To do this in DDPM, $\hat{x}_{t-\Delta t} = \mu_t(\hat{x}_t) + N(0, \sigma^2 \Delta t)$

Add noise going backwards

Now, return to claim...

From Paper...

Claim 1 (Informal). *Let $p_{t-\Delta t}(x)$ be an arbitrary, sufficiently-smooth density over $\mathbb{R}^d$. Consider the joint distribution of $(x_{t-\Delta t}, x_t)$, where $x_{t-\Delta t} \sim p_{t-\Delta t}$ and $x_t \sim x_{t-\Delta t} + \mathcal{N}(0, \sigma_q^2 \Delta t)$. Then, for sufficiently small Δt, the following holds. For all conditionings $z \in \mathbb{R}^d$, there exists μ_z such that:*

mean — Variance

$$p(x_{t-\Delta t} \mid x_t = z) \approx \mathcal{N}(x_{t-\Delta t};\ \mu_z,\ \sigma_q^2 \Delta t). \tag{16}$$

Idea: Bayes Rule → logs → drop terms that → 0 as $\Delta t \to 0$

$$p(x_{t-\Delta t}|x_t) = p(x_t|x_{t-\Delta t}) p_{t-\Delta t}(x_{t-\Delta t}) / p_t(x_t)$$

$$\log p(x_{t-\Delta t}|x_t) = \log p(x_t|x_{t-\Delta t}) + \log p_{t-\Delta t}(x_{t-\Delta t}) - \log p_t(x_t)$$

↖ Dropping temporarily because it is just normalizing factor

But $p_{t-\Delta t}(x_{t-\Delta t}) = p_t(x_{t-\Delta t}) + \Delta t \cdot \frac{\partial}{\partial t} p_t(\dots)$

So

$$= \log p(x_t|x_{t-\Delta t}) + \log p_t(x_{t-\Delta t}) + \mathcal{O}(\Delta t) \quad \text{+ Normalizer}$$

$$= -\frac{1}{2\sigma_q^2 \Delta t} ||x_{t-\Delta t} - x_t||_2^2 + \log p_t(x_{t-\Delta t}) \quad \text{+ dropped}$$

Gaussian Density of forward diffusion

Now, Taylor expand $\log p_t(\vec{x}_{t-\Delta t})$ around $\vec{x}_t$

$$= -\frac{1}{2\sigma_q^2 \Delta t} ||x_{t-\Delta t} - x_t||_2^2 + \log p_t(x_t) + \langle \nabla_x \log p_t(x_t), (x_{t-\Delta t} - x_t) \rangle + \mathcal{O}(\Delta t)$$

Normalizer — Const → 0

$$= -\frac{1}{2\sigma_q^2 \Delta t} \left(||x_{t-\Delta t} - x_t||_2^2 - 2\sigma_q^2 \Delta t \langle \nabla_x \log p_t(x_t), (x_{t-\Delta t} - x_t) \rangle \right)$$

"score"

$$= -\frac{1}{2\sigma_q^2 \Delta t} ||x_{t-\Delta t} - x_t - \sigma_q^2 \Delta t\ \nabla_x \log p_t(x_t)||_2^2 + C$$

Complete square

mean...

$$= -\frac{1}{2\sigma_q^2 \Delta t} ||x_{t-\Delta t} - \mu||_2^2$$

→ Variance

How to train $\vec{\mu}_t(\vec{x}_t)$ ← Deep Network

1) Pick $\vec{x}_0$ from training set
2) Pick time t in $[0,1]$
3) Construct $\vec{x}_{t-\Delta t} = \vec{x}_0 + N(0, \sigma^2(t-\Delta t))$

$\vec{x}_t = \vec{x}_{t-\Delta t} + N(0, \sigma^2 \Delta t)$

4) SGD step for $\vec{\mu}_t(\vec{x}_t)$ to predict $\vec{x}_{t-\Delta t}$ under squared-loss

Pseudocode 1: DDPM train loss

Input: Neural network f_θ; Sample-access to target distribution p.
Data: Terminal variance σ_q; step-size Δt.
Output: Stochastic loss L_θ

$x_0 \leftarrow \text{Sample}(p)$
$t \leftarrow \text{Unif}[0,1]$
$x_t \leftarrow x_0 + \mathcal{N}(0, \sigma_q^2 t)$
$x_{t+\Delta t} \leftarrow x_t + \mathcal{N}(0, \sigma_q^2 \Delta t)$
$L \leftarrow \|f_\theta(x_{t+\Delta t}, t+\Delta t) - x_t\|_2^2$ ($\mu_t(x_{t+\Delta t})$)
return L_θ

Pseudocode 2: DDPM sampling (Code for Algorithm 1)

Input: Trained model f_θ.
Data: Terminal variance σ_q; step-size Δt.
Output: x_0

$x_1 \leftarrow \mathcal{N}(0, \sigma_q^2)$
for $t = 1, (1-\Delta t), (1-2\Delta t), \ldots, \Delta t$ **do**
  $\eta \leftarrow \mathcal{N}(0, \sigma_q^2 \Delta t)$
  $x_{t-\Delta t} \leftarrow f_\theta(x_t, t) + \eta$
end
return x_0

Figure from Paper

Note: $\vec{\mu}_t(\vec{x}_t)$ ideally approximates $E[x_{t-\Delta t} \mid x_t]$

$=$

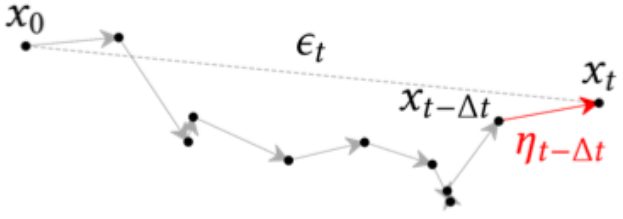
$E[x_{t-\Delta t} - x_t \mid x_t] + x_t$

So $\vec{\mu}_t(\vec{x}_t) - x_t$ is the expected negative noise.

negative of added noise in step from $t-\Delta t$ to t

By iid symmetry of added noise...

$$E[x_{t-\Delta t} - x_t \mid x_t] = \frac{\Delta t}{t} E[x_0 - x_t \mid x_t]$$



Can think of the network $\vec{\mu}_t(\vec{x}_t)$ as learning to predict original image.

i.e. $E[x_{t-\Delta t} \mid x_t] = \frac{\Delta t}{t} E[x_0 \mid x_t] + (1 - \frac{\Delta t}{t}) x_t$

Story seems great and complete.... So why/how can we sample deterministically??

DDPM-style sampling tries to walk (random) backwards through the joint distribution. That gives us the right marginals and hence a good p_0.

In Principle, we only need p_0 to be right

So walking backwards through the same marginals but a different joint distribution ok.

Example $N(0,1) \longleftrightarrow N(0,2)$
X Y

Stochastic a) $Y = X + N(0,1)$ independent $\Rightarrow$ Bayes Rule $P(x|y=y) = N(y/2, \frac{1}{2})$
i.e. sample X by $y/2 + N(0,1/2)$
Add Noise.

Deterministic b) $X \sim N(0,1)$ $Y = \sqrt{2} \cdot X$ (forward)
$X = \frac{Y}{\sqrt{2}}$

Historical Note: How did people discover this idea originally?
Continuous-time diffusion is understood using stochastic differential equations.
The probability distributions involved follow ODEs (Actually PDEs)
The smoothly changing probability distributions can be connected using transport maps that twonk one density into another. Deterministically.
Those transport maps can be ridden back from the end.

We will follow a different approach: Guess & Verify.

What should we guess??

$$\hat{x}_{t-\Delta t} = \hat{x}_t + \boxed{?} \cdot (\mu_t(x_t) - x_t)$$

Expected Negative Noise

What should $\boxed{?}$ be? Try 0: Use 1. $\Rightarrow$ Doesn't work (HW will make you see)

Try λ

General Strategy for such things. Try easiest example. That will give us a candidate.

What's the easiest example?

Scalar case. $P_0 = \delta_0$ i.e. $N(0,0)$ i.e. Always 0.

Now we know $x_t \sim N(0, t\sigma^2)$ and $\mu_t(z) = E[x_{t-\Delta t} \mid x_t = z]$

$N(0,(t-\Delta t)\sigma^2)$ $\quad$ $N(0, t\sigma^2)$

independent added noise.

Jointly normal! $\mu_t(z) = \frac{t-\Delta t}{t} z$

Plug in $x_{t-\Delta t} = x_t + \lambda\left(\frac{t-\Delta t}{t} x_t - x_t\right)$

$= x_t\left(1 - \lambda\frac{\Delta t}{t}\right)$

How to solve for λ?

Need distributions to match. $\mathrm{Var}(x_{t-\Delta t}) = \sigma^2(t-\Delta t)$

$$\cancel{\sigma^2}(t-\Delta t) = \cancel{\sigma^2} t\left(1 - \frac{\lambda\Delta t}{t}\right)^2$$

$$\sqrt{\frac{t-\Delta t}{t}} = 1 - \lambda\frac{\Delta t}{t} \Rightarrow \lambda\frac{\Delta t}{t} = 1 - \sqrt{\frac{t-\Delta t}{t}} \Rightarrow \lambda = \frac{t}{\Delta t}\left(1 - \sqrt{1-\frac{\Delta t}{t}}\right)$$

Alternate form $\lambda = \frac{t}{\Delta t}\left(1 - \frac{1}{\sqrt{t}}\sqrt{t-\Delta t}\right) = \frac{\sqrt{t}}{\Delta t}\left(\sqrt{t} - \sqrt{t-\Delta t}\right)$

$= \sqrt{t} \cdot \frac{\sqrt{t} - \sqrt{t-\Delta t}}{\Delta t}$

$= \frac{\sqrt{t}}{\sqrt{t} + \sqrt{t-\Delta t}}$

When t is "large" $\lambda \approx 1/2$ ie. Just go $1/2$ the way

At $t = \Delta t$ $\lambda = 1$
$= 2\Delta t$ $\lambda \approx 0.59$
$3\Delta t$ $\lambda \approx 0.55$
$\vdots$
$10\Delta t$ $\lambda \approx 0.51$

} Last few steps a bit diffent

But does our guess work in general? Yes!!!

See paper.

Pseudocode 2: DDPM sampling (Code for Algorithm 1)

Input: Trained model f_θ. ← μ
Data: Terminal variance σ_q; step-size Δt.
Output: x_0

$x_1 \leftarrow \mathcal{N}(0, \sigma_q^2)$
for $t = 1, (1-\Delta t), (1-2\Delta t), \ldots, \Delta t$ **do**
  $\eta \leftarrow \mathcal{N}(0, \sigma_q^2 \Delta t)$
  $x_{t-\Delta t} \leftarrow f_\theta(x_t, t) + \eta$
end
return x_0

Pseudocode 3: DDIM sampling (Code for Algorithm 2)

Input: Trained model f_θ ← μ
Data: Terminal variance σ_q; step-size Δt.
Output: x_0

$x_1 \leftarrow \mathcal{N}(0, \sigma_q^2)$
for $t = 1, (1-\Delta t), (1-2\Delta t), \ldots, \Delta t, 0$ **do**
  $\lambda \leftarrow \frac{\sqrt{t}}{\sqrt{t-\Delta t}+\sqrt{t}}$
  $x_{t-\Delta t} \leftarrow x_t + \lambda(f_\theta(x_t, t) - x_t)$
end
return x_0

ODE Perspective: Rewrite $\hat{x}_{t-\Delta t} = \hat{x}_t + \lambda(\mu_t(\hat{x}_t) - \hat{x}_t)$
$= \hat{x}_t + \Delta t \cdot \underbrace{v_t(x_t)}_{\text{Velocity Field}}$

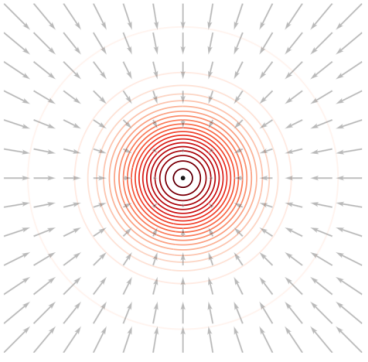


Figure 4: Velocity field v_t when $p_0 = \delta_{x_0}$, overlaid on the Gaussian distribution p_t.

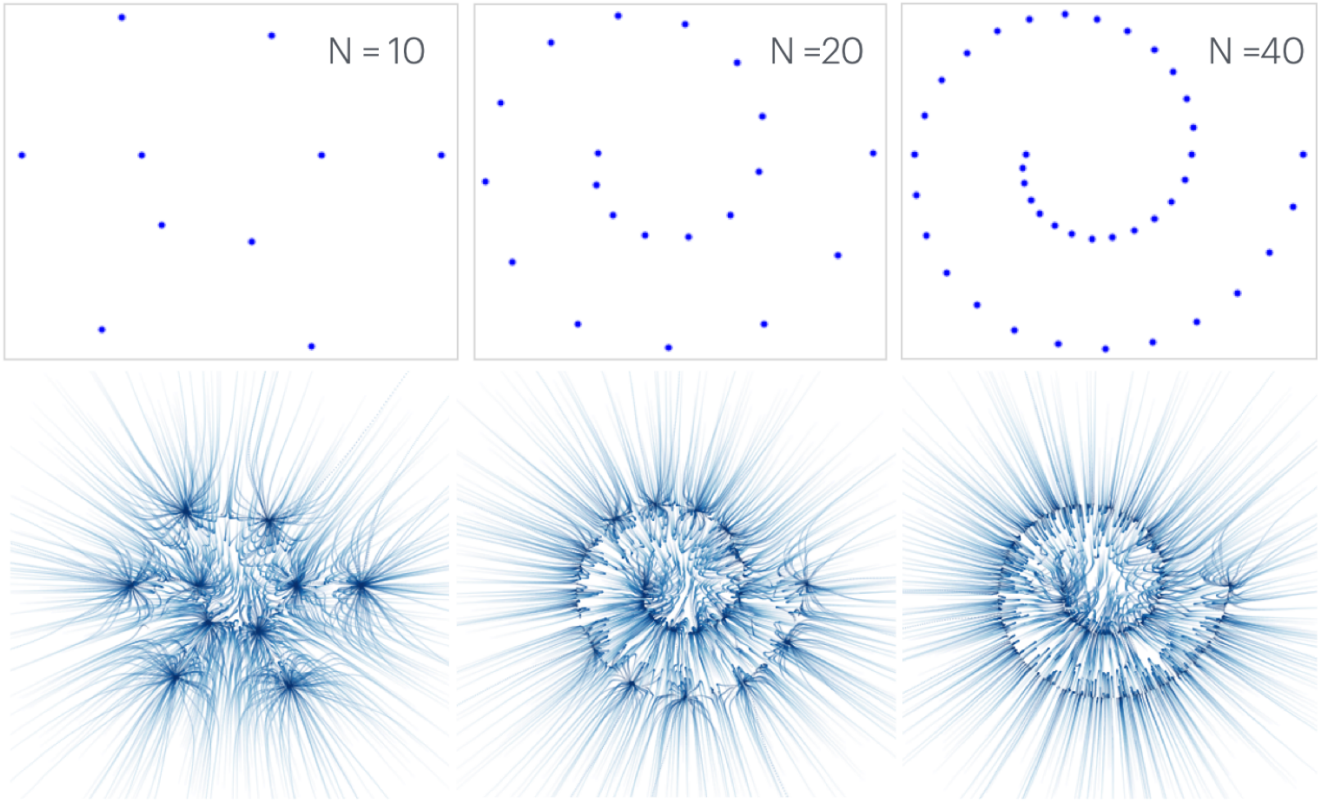


Figure 6: The DDIM trajectories (shaded by timestep t) for a spiral dataset. We compare the trajectories with $10, 20$, and 40 training samples. Note that as we add more training points (moving left to right) the diffusion algorithm begins to *learn* the underlying spiral and the trajectories look more perpendicular to the underlying manifold. The network used here is a 3 layer ReLU network with 128 neurons per layer.