L1 Overview & Generative Modeling Foundations

- Generative vs. Discriminative:
  - Discriminative Models: Learn the boundary between classes, modeling $P(Y|X)$. They map high-dimensional input to low-dimensional labels.
  - Generative Models: Learn the distribution of the data itself, modeling P(X) or the joint distribution P(X, Y). They allow for sampling new data points.
- Taxonomy of Generative Models: 1. Explicit Density: Models that explicitly define P(X) (e.g., VAEs, Diffusion, Normalizing Flows). 2. Implicit Density: Models that can sample from P(X) but don't explicitly define it (e.g., GANs).

L2 Probability, GMMs, and Expectation-Maximization (EM)

- Gaussian Mixture Models (GMM): A GMM models data as a weighted sum of K Gaussian distributions.
- Problem with MLE: We cannot easily use MLE because the summation inside the log-likelihood makes the derivative intractable.
- Latent Variables: We introduce a latent variable z (cluster assignment) to simplify the problem.
- **EM Algorithm**: An iterative method to find MLE when data has missing/latent variables.
  - E-Step: Guess the probability of the latent variables given the current parameters (calculate "responsibilities" or posterior P(z|x)).
  - M-Step: Update parameters (μ, Σ, π) to maximize the expected log-likelihood based on the responsibilities from the E-step.
  - Convergence: EM is guaranteed to monotonically increase the data likelihood.
- **Jensen's Inequality**: For a convex function f, $E[f(X)] \geq f(E[X])$. For a concave function like log, $E[\log(X)] \leq \log(E[X])$.
  - This inequality allows us to construct a lower bound on the log-likelihood, which leads to the ELBO concept used in VAEs.

L3 Dimensionality Reduction (PCA & Eigenfaces)

- Principal Component Analysis (PCA): Finds a low-dimensional subspace that maximizes the variance of the projected data. Equivalently, it minimizes the reconstruction error (L2 norm) between the original data and the projection.
- Eigenfaces: Applying PCA to face images. The "Eigenfaces" are the eigenvectors of the covariance matrix of the face dataset. Any face can be reconstructed as a linear combination of these eigenfaces.
- Connection to Generative Models: This introduces the concept of a "Latent Space" (z) where data is compressed. Standard PCA/Autoencoders have deterministic latent spaces, whereas VAEs make this probabilistic.

L4 Autoencoders (AE) and Variational Autoencoders (VAE)

- **Autoencoders (AE)**:
  - Consist of an Encoder $z = f(x)$ and Decoder $\widehat{x} = g(z)$.
  - Trained to minimize reconstruction loss: $\|x - \widehat{x}\|^2$.
  - Limitation: The latent space is not continuous or regularized, making it poor for generation (sampling random z often yields garbage).
- **Variational Autoencoders (VAE)**:
  - Probabilistic Encoder: Instead of a single point z, the encoder predicts a distribution $q_\phi(z|x) = \mathcal{N}(\mu, \sigma^2 I)$.
  - Probabilistic Decoder: $p_\theta(x|z)$ reconstructs data from the sampled z.
- **Reparameterization Trick**:
  - Problem: We cannot backpropagate gradients through a random sampling node $z \sim \mathcal{N}(\mu, \sigma)$.
  - Solution: $z = \mu + \sigma \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$.
  - This moves the randomness to an external input ϵ , making z differentiable w.r.t μ and σ.
- **The VAE Loss (ELBO)**:
  - We want to maximize $\log p(x)$, but it's intractable. Instead, we maximize the ELBO: $\mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{KL}(q(z|x)\|p(z))$
  - Term 1 (Reconstruction): Encourages the decoder to accurately reconstruct the input. Term 2 (Regularization): Forces the latent distribution q(z|x) to be close to the prior p(z) (usually $\mathcal{N}(0, I)$).

L5 VAE Analysis & The ELBO

The "Tight" ELBO Identity: The lecture proves the fundamental identity: $\log p_\theta(x) = \mathcal{L}(\theta, \phi; x) + D_{KL}(q_\phi(z|x)\|p_\theta(z|x))$

- Implication: Maximizing the ELBO ($\mathcal{L}$) achieves two things: It maximizes a lower bound on the model evidence $\log p_\theta(x)$; It minimizes the divergence between your approximate posterior q_ϕ and the true posterior$p_\theta(z|x)$.
- Tightness: The bound is "tight" only when $D_{KL} = 0$, meaning your encoder perfectly matches the true posterior.

The "Prior Hole" Problem:

- Mismatch: Even if the aggregate posterior $q(z)=E_{data}[q(z|x)]$ matches the prior p(z) generally, there can be regions in the latent space (holes) where p(z) has high density but no data maps to it.
- Consequence: Sampling from these "holes" in the prior generates poor, unrealistic samples.

Beta-VAE: Modifies the ELBO by weighting the regularization term: $\mathcal{L} = Recon - \beta \cdot D_{KL}$.

L6 Diffusion Models I (Fundamentals)

- **Forward(fixed encoder)**: Slowly destroy structure by adding noise until data becomes a standard Gaussian. **Reverse(learned decoder)**: Create structure by iteratively removing noise (denoising).
- Forward Process (The Markov Chain):
  - Defined as $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$.
  - **Markov Property**:

$$q(x_t|x_{t-1},x_0)=q(x_t|x_{t-1})=\frac{q(x_{t-1}|x_t,x_0)q(x_t|x_0)}{q(x_{t-1}|x_0)}$$

- Key Property: You can jump directly from x_0 to any step t without iterating: $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ where $\alpha_t=1-\beta_t$, and $\bar{\alpha}_t=\prod_{i=1}^{t}\alpha_i$
- Reverse Process: We want to sample from $q(x_{t-1}|x_t)$, but it requires the entire dataset (intractable).
- Solution: Learn a model $p_\theta(x_{t-1}|x_t)$ to approximate it. Since the forward steps are small Gaussians, the reverse steps are also approximately Gaussian.

L7 Diffusion Models II (Training & Math)

- The Variational Bound for Diffusion: Diffusion is trained by maximizing the ELBO, just like a VAE, but spread over T timesteps.
- The objective decomposes into:
  - L_T: Constant (prior matching); L_0: Reconstruction term.
  - L_{t-1} : Denoising matching terms (KL between forward posterior $q(x_{t-1}|x_t, x_0)$ and model $p_\theta(x_{t-1}|x_t)$).
- Simplifying the Loss: The complex KL divergence terms simplify algebraically into a Mean Squared Error (MSE) on the noise vectors.
- Instead of predicting the mean μ_θ, the network predicts the noise ϵ added at step t: $\mathcal{L}_{simple} = \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2\right]$
- Takeaway: The model learns to look at a noisy image and say "this is the noise that was added to it".
- Sampling (Langevin-like update):
  - To sample, we start with noise x_T and iterate backward:

$$x_{t-1}=\frac{1}{\sqrt{\alpha_t}}\left(x_t-\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t)\right)+\sigma_t z$$

- This equation is effectively: (Scale back) - (Predicted Noise direction) + (Random noise for exploration).

L8 Diffusion Models III (Guidance & Control)

- Conditioning: We want to generate specific data p(x|y) (e.g., "a dog") rather than just any data p(x).
- Classifier Guidance: Uses an explicit, separate classifier trained on noisy images. We shift the diffusion mean using the gradient of the classifier: $\nabla_{x_t}\log p(y|x_t)$.
- Drawback: Requires training a noise-robust classifier; adversarial attacks on the classifier can break generation.
- Classifier-Free Guidance (CFG): Avoids a separate classifier. Instead, trains a single noise predictor $\epsilon_\theta(x_t, c)$ that can handle both a prompt c and a "null" prompt $\emptyset$ (unconditional).
- **The CFG Formula:** $\tilde{\epsilon}_\theta(x_t, c) = (1+w)\epsilon_\theta(x_t, c) - w\epsilon_\theta(x_t, \emptyset)$
  - Interpretation: $\epsilon_\theta(x_t, \emptyset)$ "What a generic image looks like."
  - $\epsilon_\theta(x_t, c)$: "What the prompt image looks like."
  - Difference: The vector pointing specifically toward the prompt concepts.
  - w: The guidance scale. *low*->more diverse, higher quality samples; *higher*->adhere strictly to prompt, but reduced diversity and potential quality degradation(artifacts).

L9 Score-Based Generative Models

- **Langevin Dynamics** (Sampling): It allows sampling from a probability distribution p(x) using only its score function:

$$\nabla_x \log p_t(x)=\frac{\nabla_x p_t(x)}{p_t(x)}$$

  - Update Rule: $x_{k+1} = x_k + \tau\nabla_x \log p(x_k) + \sqrt{2\tau}z_k, z_k \sim \mathcal{N}(0, I)$
  - Interpretation: Gradient ascent on the log-density (moving towards high probability modes) + noise injection (to explore and prevent getting stuck in local optima).
- **Tweedie's Formula(denoise)**: Connects *denoising* to score matching.
  - The posterior mean of a signal observed in Gaussian noise can be estimated using the score of the marginal distribution.
  - Formula: $\mathbb{E}[x_0|x_t] = x_t + \sigma^2\nabla_{x_t}\log p(x_t)$.
  - Significance: This proves that learning to denoise (predict the mean) is mathematically equivalent to learning the score function.
- Explicit Score Matching(Intractable): $\mathcal{L}_{SM}=E_{x\sim p_t}[\|s_\theta(x)-\nabla_x \log p_t(x)\|^2]$
- **Conditional Score Matching**(Tractable):$\mathcal{L}_{CSM}=E_{x,z}[\|s_\theta(x)-\nabla_x \log p_t(x|z)\|^2]$

$$\nabla_x \log p_t(x|z) = -\frac{x-\alpha_t z}{\sigma_t^2} \approx -\frac{\epsilon}{\sigma}$$

  - $\nabla_\theta \mathcal{L}_{SM}=\nabla_\theta \mathcal{L}_{CSM}$ 最小化去噪误差等价于学习得分

L10 & 11: Guidance (Classifier & Classifier-Free)

Classifier Guidance (CBG): Uses a pre-trained image classifier $p(y|x)$ (trained on noisy images) to guide the generation.

- Modified Score: $\nabla_x \log p(x|y) = \nabla_x \log p(x) + s \cdot \nabla_x \log p(y|x)$.
- Mechanism: At each step, calculate the gradient of the class probability $\nabla_x \log p(y|x)$ and add it to the diffusion score.
- Scaling (s): When s > 1, it sharpens the distribution, forcing the model to pay more attention to the classifier label than the image prior.
- Drawback: Requires training a robust classifier on noisy data; prone to adversarial gradients.

Classifier-Free Guidance (CFG): Eliminate the separate classifier. Train a single denoiser $\epsilon_\theta(x, t, c)$ that can handle both conditional input (c) and unconditional input (∅).

  - Formula: $\tilde{\epsilon}_\theta = (1+s)\epsilon_\theta(x, t, c) - s\epsilon_\theta(x, t, \emptyset)$
  - Interpretation: This is a "Barycentric combination." The term $(\epsilon_\theta(x, c) - \epsilon_\theta(x, \emptyset))$ represents the vector direction specific to the text prompt. We scale this vector by s and add it to the unconditional estimate.

L12 CLIP, Latent Diffusion & Evaluation

- Evaluation Metrics: FID (Frechet Inception Distance): The standard metric for generative model quality.
- Process: Pass real (N) and generated (M) images through Inception-v3 to get feature vectors (2048-dim). Fit a Gaussian to both distributions: (μ_r, Σ_r) and (μ_g, Σ_g). Calculate the Wasserstein-2 distance between them.
  - Intuition: Lower FID = generated images are statistically similar to real images in high-level feature space.
  - CLIP Score: Measures semantic alignment between the generated image and the text prompt. higher is better，余弦相似度
- Text-to-Image Architectures:
  - Latent Diffusion Models (LDM)/Stable Diffusion: Performs diffusion in a compressed latent space (using VAE) rather than pixel space. Advantage: Drastically reduces computational cost while retaining high resolution.
- Dall-E 2: Uses CLIP embeddings; strong on concept composition.
- Imagen: Uses T5 text encoder; operates in pixel space; known for photorealism.

L13 Inverse Problems & Generative Priors

Problem: We want to recover a true state x from a noisy measurement y.

- Measurement Equation: $y = A(x) + n$
  - A(·): Forward operator (can be linear or non-linear).
  - n: Measurement noise.
  - Ill-posedness: Often, A is non-invertible (e.g., blurring, masking, downsampling), meaning multiple x's could map to the same y.

$$\log p(y|x)=const-\frac{1}{2\sigma_y^2}\|y-Ax\|^2$$

  - Gradient for Guidance:

$$\nabla_x \log p(y|x)\approx-\frac{1}{\sigma_y^2}A^\top(y-Ax)$$

  - Note: In DPS, we approximate this using $\hat{x}_0(x_t)$ instead of x_t.
- Classical Approach: $\widehat{x} = arg\min_x \|y - A(x)\|^2 + \lambda R(x)$
  - R(x): Regularizer (prior knowledge). Common choice: Total Variation (TV) (minimizing L1 norm of gradients to encourage smoothness).
- Generative Approach: Instead of a hand-crafted regularizer R(x), use a pre-trained diffusion model p(x) as the prior.
- Goal: Sample from the posterior $p(x|y) \propto p(y|x)p(x)$.

L14 Diffusion Posterior Sampling (DPS)

The Posterior Score:

- To sample from the posterior $p(x|y)$ using diffusion, we need its score:
- $\nabla_{x_t}\log p(x_t|y) = \nabla_{x_t}\log p(x_t) + \nabla_{x_t}\log p(y|x_t)$
- Term 1 (Prior): Given by the unconditional diffusion model $\epsilon_\theta(x_t, t)$.
- Term 2 (Likelihood): $\nabla_{x_t}\log p(y|x_t)$. This is hard to compute because $p(y|x_t)$ requires integrating over all possible x_0's derived from x_t.
- **The DPS Approximation**: We approximate the likelihood score by using the estimated mean $\widehat{x}_0(x_t)$ (derived via Tweedie’s formula): $\nabla_{x_t}\log p(y|x_t) \approx \nabla_{x_t}\log p(y|\widehat{x}_0(x_t))$
  - Implementation: Predict x_0 from noisy

$$x_t:\hat{x}_0=\frac{x_t-\sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t)}{\sqrt{\bar{\alpha}_t}}$$

    Pass$\widehat{x}_0$ through the forward operator A.
  - Compute error against $y: \|y - A(\widehat{x}_0)\|^2$.
  - Backpropagate the gradient of this error w.r.t. x_t.
  - In DPS, the reverse diffusion step is modified to sample from $p(x_{t-1}|x_t,y)\propto p_\theta(x_{t-1}|x_t)\,p(y|x_0)$, so that as $\sigma_y^2\to\infty$ (very noisy observations), DPS reduces to the standard unconditional diffusion process.
- Jensen's Gap Justification:
  - Why does this approximation work? When the noise in the forward process is high, the "Jensen's gap" is small, allowing us to approximate the expectation of the function with the function of the

expectation.

- Pi-GDM (Pseudoinverse Guided Diffusion): Another method, use the pseudoinverse $A^{\dagger}$ to guide the diffusion process for linear inverse problems.

L15 Applications (Audio & Motion)

ArrayDPS (Audio Source Separation):

- Problem: Separate K sources from a C-channel mixture x.
- Unknowns: The mixing matrix A (Room Impulse Responses - RIRs) is often unknown (Blind Source Separation).
- Algorithm: Use a diffusion model prior on the source spectrograms. Jointly estimate the sources S and the mixing parameters (RIRs/Green's function G).
- Forward Convolutive Prediction (FCP): A technique to estimate the mixing filter G dynamically during the sampling process.

L16 Differential Equations (ODEs) & Iterative Algorithms

Viewing iterative updates as steps in a continuous path.

- **ODEs as Vector Fields**: An Ordinary Differential Equation (ODE) describes how a state x changes over time: $dx/dt = f(x, t)$. Here, $f(x, t)$ acts as a *velocity vector field*, telling the particle which direction to move and how fast.
- **Iterative Algorithms as Discrete ODEs**: Many iterative algorithms (like Gradient Descent) are just *discretized versions* of continuous ODEs.
  - Gradient Descent: Can be seen as an Euler discretization of the "Gradient Flow" ODE: $dx/dt = -\nabla L(x)$.
  - Reversibility: You can reverse an ODE simply by flipping the sign of the time step or the vector field ($dx = -f(x, t)dt$).

L17 Stochastic Differential Equations (SDEs)

SDEs introduce randomness into the system, which is essential for Diffusion models.

- The SDE Formula: An SDE adds a "noise" term to the ODE: $dx = f(x,t)dt + g(t)dw$
  - $f(x, t)dt$: The Drift (deterministic part)
  - $g(t)dw$: The Diffusion (stochastic part), where w is a Wiener process (Brownian motion).
  - Sampling (Euler-Maruyama): Just as we use Euler's method for ODEs, we use the Euler-Maruyama method to solve SDEs numerically. It involves adding Gaussian noise scaled by $\sqrt{\Delta t}$ at each step.
- Reversing an SDE (Andersen's Theorem): Unlike an ODE, you can't just "flip the sign" to reverse an SDE because of the randomness.
- Andersen's Theorem:To reverse a diffusion process, you need a specific reverse SDE that involves the score function($\nabla \log p_t(x)$). The goal isn't to retrace the exact trajectory of a single particle, but to ensure the marginal distribution of the particles matches at every time step.

L18 Diffusion as SDEs

- Forward Process: The "noising" process in diffusion models (adding noise) is essentially a forward SDE that transforms data into a Gaussian distribution.
- Reverse Process: "Denoising" (generation) is sampling the reverse SDE.
- The Link: The reverse SDE requires knowing the score function. Since we don't know the true score, we train a neural network to estimate it (Score Matching) or predict the noise (DDPM), which are mathematically equivalent via Tweedie's Formula

L19 Flow Matching

Flow Matching is introduced as a more direct way to learn the transformation from noise to data, avoiding some complexities of SDEs.

- The Goal: Learn a deterministic ODE (vector field) that transforms a source distribution (e.g., Gaussian) to the target data distribution.
- The Challenge: We don't know the true vector field that connects noise to our specific data distribution.
- Conditional Flow Matching (CFM): The solution.

L20 Flow Matching

Instead of diffusing data to noise and learning to reverse it (Diffusion), just learn a deterministic path (ODE) that transforms noise directly into data.

- **The Setup**: We want to map a source distribution p_0 (Noise/Gaussian) to a target distribution p_1 (Data). We define a Vector Field $v_t(x)$ that pushes particles from p_0 to p_1 over time $t\in[0,1]$. The movement is described by an ODE: $\frac{dx}{dt}=v_t(x)$.
- **The Continuity Equation**: This equation describes how probability mass moves. It says the change in density over time plus the divergence of the flow must be zero (mass is conserved, just moved around).
  - Problem: We don't know the true vector field v_t that creates our complex data distribution.
- **Conditional Flow Matching (CFM)**:
  - Instead of figuring out the complex global field, we look at a single pair of points: one noise sample x_0 and one data sample x_1.
  - We define a simple "conditional" path for this pair. The simplest path is a straight line (Optimal Transport path): $x_t = (1-t)x_0 + tx_1$
  - The **vector field (velocity)** for this straight line is constant and easy to calculate:
$$u_t(x|x_1) = \frac{d}{dt}x_t = x_1 - x_0$$
  - The Theorem (Why this works): If you train a neural network $v_\theta(x,t)$ to match this simple conditional vector field $u_t(x|x_1)$ (averaged over all data), it automatically learns the correct marginal vector field that maps the entire noise distribution to the data distribution.
  - **Loss Function**/Training Objective: $\mathscr{L}_{CFM}(\theta) = E_{t,x_1,x_0}\| v_\theta(x_t,t) - u_t(x_t|x_1)\|^2$
- The Marginal-Conditional Relation: $u_t(x) = E_{q(z|x)}\,[u_t(x|z)]$
  - Meaning: The true vector field at x is the probability-weighted average of all possible target paths passing through x.
  - Implication: Minimizing $E[\|v_\theta(x)-u_t(x|z)\|^2]$ (Conditional Flow Matching) is equivalent to minimizing $E[\|v_\theta(x)-u_t(x)\|^2]$.
  - Proof key: Relies on the Continuity Equation and Bayes' Rule $\nabla\cdot(pu)=\int\nabla\cdot(p(\cdot|z)u(\cdot|z))dz$.

L21 Unifying ODEs and SDEs

You can perform the exact same transformation using either randomness (SDE) or a smooth deterministic path (ODE). This connects Score-based Diffusion with Flow Matching.

- **The Probability Flow ODE**: We know Diffusion is an SDE: $dx = f(x,t)dt + g(t)dw$. (Drift+Noise)
- **Song et al. Theorem**: For every SDE, there exists a deterministic ODE that produces the exact same marginal probability distributions $p_t(x)$ at every time step t.
$$dx=[f(x,t)-\frac{1}{2}g(t)^2\nabla\log p_t(x)]dt$$
- Key Insight: The term $\nabla\log p_t(x)$ is the Score Function.
- Implication: If you train a Score Model (like in standard Diffusion/DDPM), you have learned $\nabla\log p_t(x)$. You can plug this learned score into the equation above. This means a single trained network can be sampled in two ways: *1.* Stochastically (Reverse SDE): Random walk, jagged paths, adds noise at every step. *2.* Deterministically (Probability Flow ODE): Smooth curved paths, no noise added during sampling. (This is essentially what Flow Matching tries to learn directly).

L22 Surgery on Diffusion (Inpainting & Editing)

Core Idea: How do we constrain the generative process to keep parts of an image we like (conditional generation) or fix broken images?

- Inpainting (RePaint Algorithm):
  - Goal: You have an image with a missing region (Mask M). You want to fill M with generated content while keeping the known region (1-M) consistent.
  - The Naive Fail: You can't just generate a generic image and paste the known pixels on top at the end. The boundary will look fake.
  - The Solution: You must enforce consistency during the iterative steps.
  - The Algorithm: At every reverse step $t\rightarrow t-1$:
    - Sample: Predict the whole image x_{t-1} using the diffusion model.
    - Replace: Overwrite the known pixels in x_{t-1} with the noisy version of the original image.
    - Keep: Keep the model's prediction for the unknown (masked) area.
  - This allows the "known" pixels to influence the "unknown" pixels via the Score Function/Attention mechanisms over time.
- The Manifold Hypothesis & Projection:
  - Real images live on a low-dimensional "manifold."
  - Editing via Projection: If you scribble on a face (e.g., draw a red line), you have pushed the image off the manifold of real faces.
  - SDE Edit: If you add noise to this corrupted image (move it toward the Gaussian prior) and then run the reverse diffusion (denoise), the model "projects" the image back onto the manifold.
  - The red line (corruption) is interpreted as "noise" or a feature to be integrated, resulting in a realistic face that might now have a scar or war paint, depending on how far you diffused it.

Gaussians & Probability Foundations

- **Multivariate Gaussian Definition(PDF)**: For $x\in\mathbb{R}^d\sim\mathcal{N}(\mu,\Sigma)$:
$$p(x)=p_t(x|z)=\frac{1}{\sqrt{(2\pi)^d|\Sigma|}}\exp\left(-\frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu)\right)$$
- **Log-Likelihood**:
$$\log p(x) = -\frac{d}{2}\log(2\pi)-\frac{1}{2}\log|\Sigma|-\frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu)$$
$$\log p(y|x) = -\frac{1}{2\sigma^2}\|y-x_0\|^2 + C(const)$$
- **Score Function**:
$$\nabla_x\log p(x)=-\Sigma^{-1}(x-\mu)(\sigma^2 I:-\frac{(x-\mu)^2}{\sigma^2}\text{ isotropic})$$
- Affine Transformation (Reparameterization): If $x\sim N(\mu,\Sigma)$ and $y=Ax+b$: $y\sim N(A\mu+b, A\Sigma A^\top)$
- Scalar case: If $z=\mu+\sigma\epsilon$ where $\epsilon\sim\mathcal{N}(0,I)$, then $z\sim\mathcal{N}(\mu,\sigma^2 I)$.
- **KL Divergence** between Gaussians: For $q=N(\mu_1,\Sigma_1)$and $p=N(\mu_2,\Sigma_2)$:
$$D_{KL}(q\|p) = E_p[logp/q] = \int p(x)log\frac{p(x)}{q(x)}dx=\frac{1}{2}\left[\log\frac{|\Sigma_2|}{|\Sigma_1|}-d+tr(\Sigma_2^{-1}\Sigma_1)+(\mu_2-\mu_1)^\top\Sigma_2^{-1}(\mu_2-\mu_1)\right]$$
- Entropy: $H(p) = -E_p[\log p]$, Cross-Entropy: $H(p,q) = -E_p[\log q]$
  - $H(p,q)=H(p)+D_{KL}(p\|q)$
- **Standard VAE Case**: $q(z|x)=\mathcal{N}(\mu,\sigma^2 I)$, $p(z)=\mathcal{N}(0,I)$:
$$D_{KL}=-\frac{1}{2}\sum^{d}(1+\log(\sigma_j^2)-\mu_j^2-\sigma_j^2)$$
- **Chain rule probability**: $q(x_{t-1}, x_t|x_0) = q(x_t|x_{t-1}, x_0)q(x_{t-1}|x_0)$

VAE & ELBO

- $\log p_\theta(x)\geq ELBO=\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]-D_{KL}(q_\phi(z|x)\|p(z))$
  - Decomposition: $\log p_\theta(x)=ELBO+D_{KL}(q_\phi(z|x)\|p_\theta(z|x))$
  - Tightness: ELBO is tight when $q_\phi(z|x)=p_\theta(z|x)$ (KL term is 0).
  - $ELBO=\mathscr{L}(\theta,\varnothing;x)$
- **Loss Functions**
  - Reconstruction Term: Maximizing $\mathbb{E}[\log p_\theta(x|z)]$ with a Gaussian decoder is equivalent to minimizing MSE: $\|x-\mu_\theta(z)\|^2$.
  - Beta-VAE: $\mathcal{C}=Recon-\beta\cdot D_{KL}$.
    - $\beta>1$: Better disentanglement, worse reconstruction (prior hole).
    - $\beta<1$: Better reconstruction, entangled latents.
- Optimal Surrogate: To maximize ELBO w.r.t $q(fixed\theta)$, the optimal $q^*(z)$ is the true posterior: $q^*(z|x)=p_\theta(z|x)=p_\theta(x|z)p(z)\big/p_\theta(x)$
- **Marginal expectation:** $E_{p(x_t)}[f(x_t)] = \int p(x_t)f(x_t)\,dx_t$

Diffusion Models

- Forward Process (DDPM)
  - Markov Step: $x_t=\sqrt{1-\beta_t}x_{t-1}+\sqrt{\beta_t}\epsilon_t$
  - One-Step Marginal: $x_t=\sqrt{\bar\alpha_t}x_0+\sqrt{1-\bar\alpha_t}\epsilon$, where $\bar\alpha_t=\prod(1-\beta_i)$.
    - $q(x_t|x_0)=\mathcal{N}(x_t;\sqrt{\bar\alpha_t}x_0,(1-\bar\alpha_t)I)$
- Score Matching & Tweedie's Formula
  - Relating the true mean to the noisy observation and the score: $\mathbb{E}[x_0|x_t]=x_t+\sigma_t^2\nabla_{x_t}\log p(x_t)$
- For DDPM, variance is $1-\bar\alpha_t$, so
$$x_0\approx\frac{x_t+(1-\bar\alpha_t)\nabla\log p_t(x_t)}{\sqrt{\bar\alpha_t}})$$
- **Langevin** Dynamics
  - Iterative sampling using the score function:
$$x_{k+1}=x_k+\tau\nabla_x\log p(x_k)+\sqrt{2\tau}z_k,\quad z_k\sim\mathcal{N}(0,I)$$
  - Without noise: Reduces to gradient ascent (finds mode, lacks diversity).

Flow Matching & ODEs

- Flow Definitions: Interpolant: $X_t=\alpha_t X_1+\beta_t X_0$ 公式是 x0-x1 的路径
  - Conditional Velocity Field: $u_t(x|x_1)=\frac{d}{dt}X_t=\dot\alpha_t x_1+\dot\beta_t x_0$

(Substitute x_0 from the interpolant equation to get u_t in terms of x and x_1 only).

- Marginalized Vector Field: $u_t(x)=\mathbb{E}_{q(x_1|x)}[u_t(x|x_1)]$
  - Relation to Score: For Gaussian paths, $u_t(x)$ is a linear combination of x and the score $\nabla_x\log p_t(x)$.
- ODE Solvers, Given $\frac{dx}{dt}=f(x,t)$:
  - Euler's Method (1st Order): $x_{t+h}=x_t+hf(x_t,t)$
  - Reversing an ODE: To reverse time (generate data from noise), solve:
$$\frac{d\tilde{x}}{d\tau}=-f(\tilde{x},T-\tau)$$
  (Velocity field changes sign).
- Runge-Kutta (RK4): Higher accuracy ($O(h^5)$ Euler's $O(h^2)$), larger stable step sizes, ideal for deterministic probability flows.
- **Marginalization**: $p_t(x)=\int p_t(x|z)\,p(z)dz$
- Marginal Score Identity: $\nabla_x\log p_t(x)=E_{z|x}[\nabla_x\log p_t(x|z)]$

Miscellaneous & Math Tricks

- Jensen's Inequality: $f(\mathbb{E}[X])\leq\mathbb{E}[f(X)]$ for convex f (e.g., $-\log$). Used to prove $D_{KL}\geq 0$. It becomes an equality iff X is an constant.
- LOTUS: $\mathbb{E}[g(X)]=\int g(x)p(x)dx$.
- Trace Trick: $x^\top Ax=tr(x^\top Ax)=tr(Axx^\top)$. Used in expectations of quadratic forms.
- InfoNCE Loss:
$$\mathscr{L}=\underbrace{-s(x,x^+)}_{Alignment}+\underbrace{\log\sum\exp(s(x,x^-))}_{Uniformity}$$
- PCA: Maximizes variance $\max_w w^\top\Sigma$ w s.t. $\|w\|^2=1$. Solution is the eigenvector of Σ with largest eigenvalue.
- Bayes rule:
$$p(z|x)=\frac{p_t(x|z)p(z)}{p_t(x)}$$
- Joint Probability: $P(a,b|c,d)=P(a|b,c,d)\,P(b|c,d)$
- Conditional Probability: $P(a,b)=P(b)\,P(a|b)$
- Chain rule for Probability: $P(a,b|c)=P(a|b,c)\,P(b|c)$
- Score function log-derivative trick: $\nabla_x\log p(x)=\nabla_x p(x)/p(x)$
- Trace operator: $\mathbb{E}[tr(AB)]=tr(\mathbb{E}[A]B)$