

ECE194 Fall25 Term Report

Daniel Dai

December 2025

1 Introduction

This quarter, I studied the foundations of diffusion models, focusing in particular on the paper *Pixel Super-Resolved Virtual Staining of Label-Free Tissue Using Diffusion Models*[3]. Now, as I begin learning flow matching, I am preparing to replicate virtual staining results using this more efficient generative framework. This report covers the mathematical theory underlying both models and compares their inputs and outputs to highlight the efficiency gains offered by flow matching.

2 Diffusion Model

2.1 Forward Process

Standard diffusion models like Denoising Diffusion Probabilistic Models (DDPM) gradually add Gaussian noise to a data sample x_0 . Over T timesteps, the original signal is destroyed until it becomes pure isotropic Gaussian noise, $x_T \sim \mathcal{N}(0, \mathbf{I})$.

In our BBDM model, instead of decaying to random noise, the forward process is defined as a stochastic interpolation between two fixed endpoints: the high-resolution ground truth image (x_0) and the low-resolution input image (x_T). The distribution of the intermediate state x_t is given by:

$$x_t \sim \mathcal{N}((1 - m_t)x_0 + m_t x_T, \delta_t \mathbf{I}) \quad (1)$$

where m_t represents the time-dependent interpolation coefficient and δ_t represents the variance schedule.

The mean of the distribution, $\mu_t = (1 - m_t)x_0 + m_t x_T$, acts as a deterministic linear interpolation. As t progresses from 0 to T , m_t shifts from 0 to 1, smoothly transitioning the expectation of the image from the clean target x_0 to the degraded input x_T . And the variance term $\delta_t \mathbf{I}$ functions as the "bridge." The variance of the Brownian Bridge is constrained to be zero at the endpoints ($t = 0$ and $t = T$) and attains its maximum at the midpoint of the process.

2.2 Reverse Process

The goal of the reverse process is to restore the high-resolution ground truth x_0 from the degraded input x_T . To achieve this, we employ a neural network, specifically a U-Net architecture, to iteratively reverse the diffusion steps. At any given timestep t , the network receives the current noisy state x_t and the time index t as inputs. Additionally, the original low-resolution input x_T is provided as a condition to guide the restoration.

Crucially, rather than predicting the clean image directly, the network is trained to predict the noise component $\epsilon_\theta(x_t, t, x_T)$ that is currently present in the image. Once this noise is estimated, we subtract it to approximate the mean of the previous state. In the standard "Vanilla" sampling strategy, we then add a randomly sampled Gaussian noise term to this result.

Mathematically, the update step for the Vanilla sampling strategy is formulated as[3]:

$$x_{t-1} = c_{xt}x_t + c_{yt}x_T + c_{\epsilon t}\epsilon_\theta(x_t, t, x_T) + \sqrt{\tilde{\delta}_t}\epsilon_t \quad (2)$$

where $\epsilon_t \sim \mathcal{N}(0, \mathbf{I})$ represents the added noise, $\tilde{\delta}_t$ is the variance term for the reverse step, and $c_{xt}, c_{yt}, c_{\epsilon t}$ are the time-dependent coefficients derived from the Brownian Bridge schedule.

While the Vanilla strategy adds randomness to help preserve texture, there are situations—especially near the end of the restoration—where reducing variance is more important to avoid introducing artifacts. The Mean sampling strategy handles this by taking a fully deterministic path. It relies on the same predictive model as the Vanilla method but simply removes the added noise term, producing a smoother and more controlled update at each step[3]:

$$x_{t-1} = c_{xt}x_t + c_{yt}x_T + c_{\epsilon t}\epsilon_\theta(x_t, t) \quad (3)$$

By eliminating stochasticity, mean diffusion guides the sample along a smooth, nearly linear trajectory back toward x_0 , resulting in a more stable and less noisy reconstruction.

To accelerate the inference process, the Skip strategy bypasses the iterative chain entirely for certain steps. Instead of computing the incremental state x_{t-1} , this method uses the current noisy state x_t and the predicted noise ϵ_θ to estimate the clean ground truth x_0 directly in a single calculation:

$$\hat{x}_0 = \frac{x_t - m_t x_T - \sqrt{\delta_t} \epsilon_\theta(x_t, t)}{1 - m_t} \quad (4)$$

By estimating \hat{x}_0 directly, the model can traverse the diffusion trajectory much faster, though potentially at the cost of the fine-grained high-frequency details captured by the iterative Vanilla process.

In sum, diffusion models offer strong generative performance, stable training, and the ability to model complex data, but they require many sequential sampling steps and rely on stochastic reverse processes that make generation slow and computationally expensive. These limitations motivate a shift toward flow

matching, which preserves the expressiveness of diffusion models while enabling faster, deterministic, and more efficient sampling.

3 Flow Matching for Generative Modeling

Flow Matching is a generative model that is based on Continuous Normalizing Flows (CNFs) defined by Ordinary Differential Equations (ODEs). The core objective is to learn a time-dependent vector field that "pushes" a simple source distribution p_0 along a trajectory to match a complex target data distribution p_1 .

3.1 Probability Density Paths

We define a *probability density path* as a time-dependent probability density function $p_t(x)$ for $t \in [0, 1]$. At $t = 0$, the distribution p_0 is a simple prior, typically $\mathcal{N}(0, \mathbf{I})$. Whereas at $t = 1$, the distribution p_1 matches the data distribution $q(x)$.

Mathematically, we define such path with a flow map $\phi_t : R^d \rightarrow R^d$, where the density at time t is defined by the *push-forward* equation[1]:

$$p_t = [\phi_t]_* p_0 \quad (5)$$

This operator implies that sampling $x \sim p_0$ and computing $\phi_t(x)$ yields a sample distributed according to p_t .

3.2 Vector Fields and the Continuity Equation

We define this flow $\phi_t : R^d \rightarrow R^d$ using an Ordinary Differential Equation (ODE): specifically, we define a time-dependent vector field $v_t(x)$ that specifies the velocity (speed and direction) of the flow at every point. The trajectory of a sample is thus given by integrating this velocity over time:

$$\frac{d}{dt} \phi_t(x) = v_t(\phi_t(x)) \quad (6)$$

This ODE formulation provides a powerful link between individual sample trajectories and the evolution of the entire probability distribution. And to prove this works, the evolution of the probability density p_t under this vector field is governed by the **Continuity Equation**[2]:

$$\frac{\partial p_t}{\partial t} + \nabla \cdot (p_t v_t) = 0 \quad (7)$$

This partial differential equation dictates that the rate of change of probability density at a point is exactly equal to the divergence of the probability flux at that point. The equation guarantees probability conservation, as no probability mass is created or destroyed when the probability density flows from p_0 to p_1 .

3.3 Training and Inference

Directly solving for the optimal vector field is intractable. Instead, we approximate this field using a neural network $v_\theta(x, t)$.

For the input, the network takes the current intermediate sample x (an interpolation between source and target) and the time step t as inputs. Its output is the predicted velocity vector $\hat{v} \in R^d$, which we will use to proceed to the next step in the generative flow. And we train the neural network v_θ to predict this direction by minimizing the mean squared error between the predicted velocity and this target vector[1]:

$$\mathcal{L}(\theta) = E_{t,x_0,x_1} [||v_\theta(\psi_t(x), t) - (x_1 - x_0)||^2] \quad (8)$$

Once the model is trained, we can initiate the generation process from any intermediate state x_t along the trajectory. By numerically solving the ODE using the predicted velocity $v_\theta(x_t, t)$, the integration effectively pushes the sample along the learned probability path from its current time t until it reaches the final target distribution at $t = 1$.

4 Conclusion

Flow matching offers a remarkably efficient alternative to traditional diffusion models. In practice, an entire flow-matching generative pipeline can be implemented in only a few hundred lines of code to the thousands typically required for diffusion-based systems. Its computational advantages arise from eliminating both the forward and reverse stochastic processes and replacing them with a single deterministic ODE integration, which dramatically reduces the number of sampling steps while maintaining high generative fidelity. However, a notable drawback is that flow matching can be more sensitive to the choice of interpolation path and may require careful tuning to ensure stable training, especially on complex datasets. With these considerations in mind, we are now ready to apply flow matching to virtual staining and evaluate its performance relative to diffusion-based approaches.

References

- [1] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023. <https://arxiv.org/abs/2210.02747>.
- [2] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024. <https://arxiv.org/abs/2412.06264>.

- [3] Yijie Zhang, Luzhe Huang, Nir Pillar, Yuzhu Li, Hanlong Chen, and Aydogan Ozcan. Pixel super-resolved virtual staining of label-free tissue using diffusion models. *Nature Communications*, 16(1):1–15, 2025. <https://www.nature.com/articles/s41467-025-60387-z>.