Notes on Flow Matching and Diffusion Models

compiled by D.Gueorguiev, 9/13/2025

# Introductory Remarks

**Notation**:

Capitalized script letters denote sets or functions with set domain e.g.

Capitalized bold italic letters denote matrix quantities e.g.

Small cap bold italics letters denote vector quantities e.g.

Small cap italics letters denote scalar or string quantity e.g.

Large cap italics letter denote scalar parameters e.g.

Reserved letters:

Small cap italic is reserved for quantities representing probability distributions e.g.

Large cap script is reserved for the normal distribution e.g.

We will be discussing two widely used generative AI algorithms: denoising diffusion models, and flow matching.

These generative models generate objects by iteratively converting noise into data with the help of ordinary or stochastic differential equation (ODEs/SDEs) models. Flow matching and denoising diffusion models are a family of techniques that allow us to construct, train, and simulate such ODEs/SDEs at large scale with Deep Neural Nets (DNNs).

Before delving into the models, we will list the different data types (aka *modalities*) which we will consider and their numerical representations

**Image**: consider images with pixels where describes the height and the width of the image, each with three color channels (RGB). For every pixel and every color channel, it is given an intensity value in . Thus, the image can be represented by an element .

**Video**: video is viewed as a series of images in time. If we have time points (aka *frames*), a video would therefore be represented by an element .

**Molecular structure**: A naïve way would be to represent the structure of a molecule by a matrix where is the number of atoms in the molecule and each describes the location of that atom.

In all of the above example the object we want to generate can be represented as a vector. Therefore, in our further model discussion the object being generated will be referred as vectors .

# Generative Modeling as Sampling

There is a spectrum of images that fit better or worse a predefined concept. We consider this spectrum of images to be represented by probability distribution, which we will denote as the *data distribution* . Using the statistical distribution interpretation, we replace *the goodness* of the fit of an object (image/video/molecule) with *the likelihood* of a fit. Thus, we can express the task of object generation as a sampling from (unknown) distribution . Thus, a generative model is a machine learning mode that allows us to generate samples from .

We need data to train models. We can construct a finite number of examples sampled independently from which serves as a proxy of the true distribution:

Conditional Generation

In many cases we want to generate an object *conditioned* on some data . For example, we might want to generate an image conditioned on “a flower under a cloudy sky”. This can be expressed as a sampling from a conditional distribution:

is the conditional data distribution. The conditional generative modeling involves learning to condition on an arbitrary rather than fixed choice of . The techniques for unconditional generation are readily generalized to the conditional case.

From Noise to Data

Let us assume that we have access to some initial distribution that we can sample from such as the Gaussian

The goal of generative modeling is then to transform samples from into samples from . The initial distribution is some known statistical distribution.

# Flow and Diffusion Models

A *trajectory* , which is solution of ODE is expressed as a function in the form:

With each ODE there is an associated *vector field* which is a function in the form:

Thus, for every time and location

# References

[1] [An Introduction to Flow Matching and Diffusion Models, Peter Holderrieth and Ezra Erives, Notes from MIT Class 6.S184, 2025](https://github.com/dimitarpg13/information_theory_and_statistical_mechanics/blob/main/literature/articles/generative_models/An_Introduction_to_Flow_Matching_and_Diffusion_Models_Holderrieth_MIT_2025.pdf)

[2] MIT Class 6.S184 website: <https://diffusion.csail.mit.edu/>

[3] [Score-Based Generative Modeling through Stochastic Differential Equations, Yang Song et al, Stanford U., 2021](https://github.com/dimitarpg13/information_theory_and_statistical_mechanics/blob/main/literature/articles/generative_models/Score-Based_Generative_Modeling_through_Stochastic_Differential_Equations_Song_2021.pdf)

[4] [Flow Matching for Generative Modeling, Yaron Lipman et al, Meta FAIR, 2023](https://github.com/dimitarpg13/information_theory_and_statistical_mechanics/blob/main/literature/articles/generative_models/Flow_Matching_for_Generative_Modeling_Lipman_Meta_2023.pdf)

[5] [Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, X. Liu et al, 2022](https://github.com/dimitarpg13/information_theory_and_statistical_mechanics/blob/main/literature/articles/generative_models/Flow_Straight_and_Fast-Learning_to_Generate_and_Transfer_Data_with_Rectified_Flow_Liu_2022.pdf)

[6] [Stochastic Interpolants: A Unifying Framework for Flows and Diffusions, Michael S. Albergo et al, 2023](https://github.com/dimitarpg13/information_theory_and_statistical_mechanics/blob/main/literature/articles/generative_models/Stochastic_Interpolants-A_Unifying_Framework_for_Flows_and_Diffusions_Albergo_NYU_2023.pdf)

[7] [Flow Matching Guide and Code, Yaron Lipman et al, 2024](https://github.com/dimitarpg13/information_theory_and_statistical_mechanics/blob/main/literature/articles/generative_models/Flow_Matching_Guide_and_Code_Lipman_2024.pdf)

[8] flow\_matching library at <https://github.com/facebookresearch/flow_matching>