Statistical approach "review"

Samy Braik

April 2025

Two paradigms are mentioned in the following sections. The first one is generative models and the second one nonparametric density estimation. These approaches are particularly relevant in our setting because they make few, if any, assumptions on the shape of the data.
On one hand, generative models are the go-to techniques to learn and sample from an unobserved probability distribution. They more or less learn the true distribution, sometimes implicitly, but they are mostly effective at generating new data. On the second hand, density estimation solely focus on the first goal, learning the distribution. Although I argue that we could sample according to a good density estimation.

1 Generative models

All the methods described in this section follow the same framework. They want to link an unknown distribution with density p to a simpler distribution with density q. Either directly like flow methods or up to a certain degree of precision like diffusion models.

1.1 Normalizing flow

Let $X_0 \in \mathbb{R}^d$ distributed according to q a simple distribution, a Gaussian for example, and p a target distribution.
Let $f : \mathbb{R}^d \to \mathbb{R}^d$, an invertible and differentiable function and set $X_1 := f(X_0)$ such that $X_1 \sim p$.
We can write p as a function of q and f using the change of variable formula

$$p(X_1) = q(f^{-1}(X_1)) \left| \det \frac{\partial f^{-1}}{\partial X_1}(X_1) \right| = q(X_0) \left| \det \frac{\partial f}{\partial X_0}(X_0) \right|^{-1} \tag{1}$$

$$\implies \log p(X_1) = \log q(X_0) - \log \left| \det \frac{\partial f}{\partial X_0}(X_0) \right| \tag{2}$$

Therefore the goal is to learn f_θ, approximation of f, such that $X_1 \simeq f_\theta(X_0)$.
A structure is imposed to f_θ, we define $f_1 \dots f_k$ simple bijective and differentiable transformations, such that

$$f_\theta = f_K \circ f_{K-1} \circ \dots \circ f_2 \circ f_1 \tag{3}$$

where each f_k is chosen such that its Jacobian determinant is easy to compute.

This leads to the following chain of inverse transformations

$$Z_K = X_1, \quad Z_{K-1} = f_K^{-1}(Z_K), \quad Z_{K-2} = f_{K-1}^{-1}(Z_{K-1}), \ \dots, \ Z_0 = f_1^{-1}(Z_1), \tag{4}$$

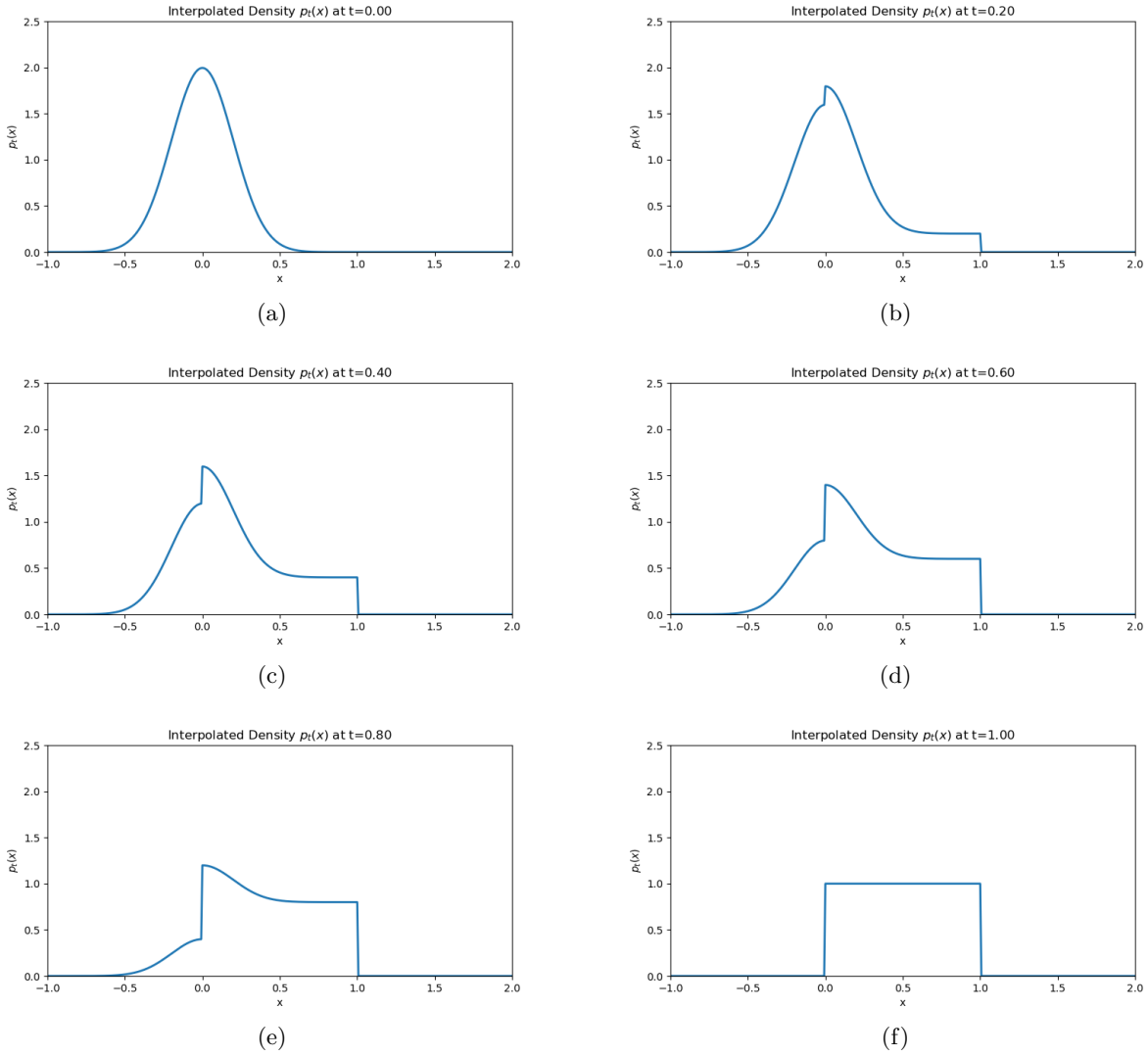
To learn f_θ, we maximize the likelihood of the data, or equivalently minimize the negative log-likelihood.

$$\mathcal{L}_{\mathrm{NF}}(\theta) = -\frac{1}{n} \sum_{i=1}^{n} \left[\log q(Z_0^{(i)}) + \sum_{k=1}^{K} \log \left| \det \left(\frac{\partial f_k^{-1}}{\partial z_k}(Z_k^{(i)}) \right) \right| \right], \tag{5}$$

where $Z_0^{(i)} = f_\theta^{-1}(X_i)$, and each $Z_k^{(i)}$ is computed recursively via the inverse transformations.

2 Notations

We start by defining a probability density path $p : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ meaning that for each time t, p_t is density function i.e. $\int p_t(x)dx = 1$.
A simple example of such a path is a path p interpolating two density p_0 and p_1 with $p_t = tp_1 + (1-t)p_0$



(a) (b) (c) (d) (e) (f)

Figure 1: A probability path interpolating $\mathcal{N}(0, 0.2)$ and $\mathcal{U}([0,1])$

Next we introduce a core object, a time dependant vector field $v : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ which can be used to construct a map $\phi : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$, called a flow, by the following ODE

$$\begin{aligned} \frac{d}{dt}\phi_t(x) &= v_t(\phi_t(x)) \\ \phi_0(x) &= x \end{aligned} \tag{6}$$

The link between the flow and the probability path is given by the change of variables formula

$$p_t(x) = q(\phi_t^{-1}(x)) \det \left[\frac{\partial \phi_t^{-1}}{\partial x}(x) \right] \tag{7}$$

This coincides with the normalizing framework.
The link between the vector field and the probability path is given by the continuity equation

$$\frac{d}{dt}p_t(x) + \operatorname{div}(p_t(x)v_t(x)) = 0 \tag{8}$$

It said that the vector field v_t generates the probability path p_t if the continuity equation holds.

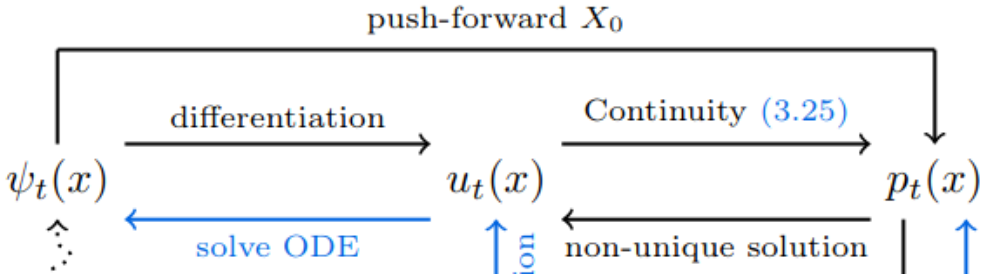


Figure 2: How the notions are linked together, from [6]

3 Objective

Given a target probability path p_t and a corresponding v_t vector field, the naïve flow matching loss is

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t,p_t(x)} \left[\|v_t^\theta(x) - v_t(x)\|^2 \right] \tag{9}$$

But we don't have acces to v_t and p_t. To adress this problem and given a particular data sample x_1, we introduce conditional probability path $p_t(x|x_1)$ such that $p_0(x|x_1) = q(x)$ at time $t = 0$ and by marginalizing over x_1 we can recover the marginal probability path

$$p_t(x) = \int p_t(x|x_1)q(x_1)dx_1 \tag{10}$$

So instead of defining a path between two entire distributions, we take a samples from our distributions and we just define how to go from one to the other.
In the same vein, we can define a conditional vector field, assuming $p_t(x)$ for all t and x

$$v_t(x) = \int v_t(x|x_1) \frac{p_t(x|x_1)q(x_1)}{p_t(x)} dx_1 \tag{11}$$

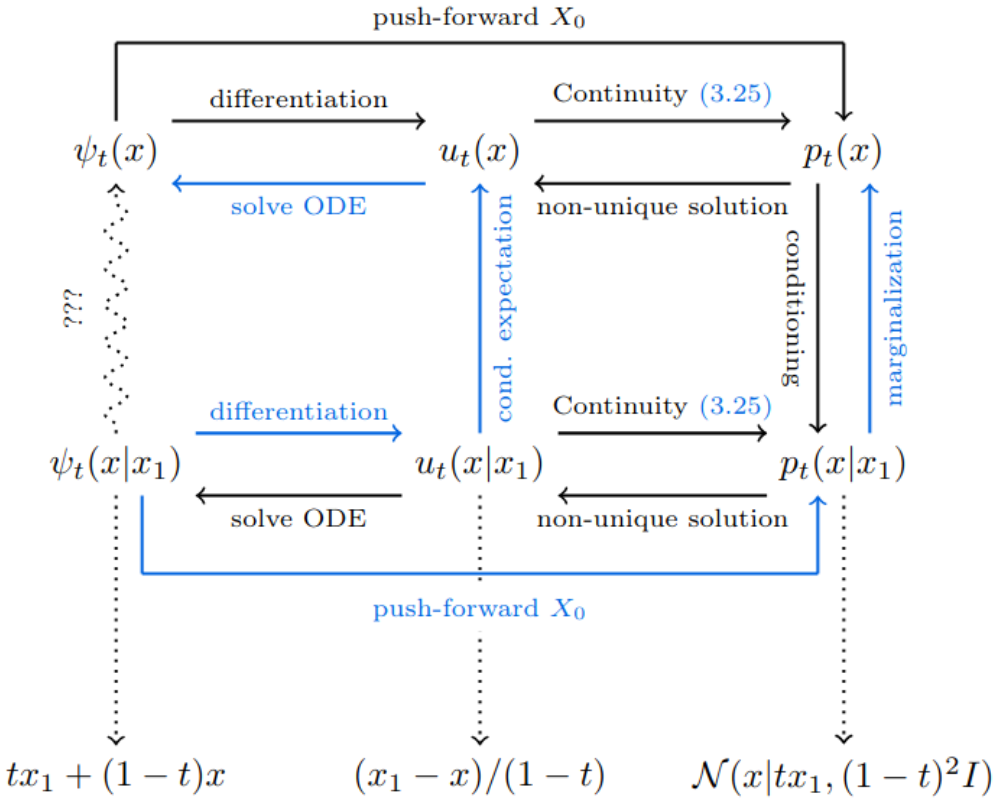


Figure 3: A visual in [6] that illustates well the full framework

Then the author introduces a new loss function called the conditional flow matching loss

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t,q(x_1),p_t(x|x_1)}\left[\|v_t^\theta(x|x_1) - v_t(x|x_1)\|^2\right] \tag{12}$$

with a strong property : $\mathcal{L}_{\text{FM}}(\theta) = \mathcal{L}_{\text{CFM}}(\theta)$ up to a constant independent of θ.
So the focus is now on designing a conditional probability path and vector field and it turns out "the conditional flow matching objective works with any choice of conditional probability path and conditional vector fields". Futhermore, there is an infinite number of vector fields that generate any particular probability path.

4 Example

In the original flow matching paper ([5]), they consider the instance of a Gaussian conditional probability path.

4.1 Diffusion

The general framework of diffusion is divided in two phases. In the forward phase (noising), we start from a random variable distributed according to our target distribution p, gradually add noise until it reaches an easy-to-sample distribution q which is practically always a Gaussian. Then, in the backward phase (denoising) we reverse the process and start from q to get back to p.

We consider $T \in \mathbb{N}^*$ a finite time horizon, a noise schedule $\beta : [0, T] \to \mathbb{R}_+^*$, assumed to be continuous and non-decreasing, B_t a Brownian motion at time t.

Forward and Backward processes

$$d\overrightarrow{X}_t = \frac{-\beta(t)}{2\sigma^2}\overrightarrow{X}_t dt + \sqrt{\beta(t)}dB_t, \quad \overrightarrow{X}_0 \sim p \quad \text{Forward process} \tag{13}$$

$$d\overleftarrow{X}_t = \left(\frac{\beta(T-t)}{2\sigma^2}\overleftarrow{X}_t + \beta(T-t)\nabla \log p_{T-t}\left(\overleftarrow{X}_t\right)\right)dt + \sqrt{\beta(T-t)}dB_t, \quad \overleftarrow{X}_0 \sim p_T \quad \text{Backward process} \tag{14}$$

Since we only noise the random variable until a finite time T, the resulting distribution p_T is not exactly equal to the distribution q, however with a good choice of T and β, we can hope that $p_T \simeq q$. Furthermore, the backward process allows us to retrieve p but the score ∇p_t is unkown at each time t. To adress this problem, denoising score matching is used.

Denoising Score Matching
Let $s : \mathbb{R}^d \to \mathbb{R}^d$. X a random variable with density p and ε an independant random variable with density g, a centered Gaussian density. Then

$$\mathbb{E}[|\nabla \log p_t(X+\varepsilon) - s(X+\varepsilon)|^2] = c + \mathbb{E}[|\nabla \log g(\varepsilon) - s(X+\varepsilon)|^2] \tag{15}$$

$$= c + \mathbb{E}[|(-\varepsilon/\mathrm{Var}(\varepsilon))g(\varepsilon) - s(X+\varepsilon)|^2] \tag{16}$$

with c a constant not related to s.

With a good architecural choice of the neural network s_θ (data dependent) and noise schedule, we can use the loss to learn the score function and generate new samples from the target distribution using the bakward SDE.

5 Nonparametric density estimation

The second approach that could be useful in our situation is nonparametric density estimation. Given an i.i.d. dataset $(X_1, \ldots, X_n)$ drawn from a distribution with density f. The goal, like the name suggests, is to estimate f without assuming a specific parametric form. The two most common methods are kernel density estimation and projection estimator.

5.1 Kernel estimator

Consider a kernel function K which is a symmetric density, H a $d \times d$ symmetric and positive definite matrix, $x \in \mathbb{R}^d$, the kernel estimator is defined by

$$\hat{f}_H(x) := \frac{1}{n|\mathrm{H}|^{1/2}} \sum_{j=1}^{n} K\left(\mathrm{H}^{-1/2}(x - X_j)\right) = \frac{1}{n} K_{\mathrm{H}}\left(x - X_j\right) \tag{17}$$

with $K_{\mathrm{H}}(x) := |\mathrm{H}|^{-1/2} K(\mathrm{H}^{-1/2}x)$.
A widely used kernel is the Gaussian kernel : $K_{\mathrm{H}}(x) = (2\pi)^{-d/2}|\mathrm{H}|^{-1/2} e^{-\frac{1}{2}x^\intercal \mathrm{H}^{-1}x}$
The choice of H is critical since it governs the bias-variance tradeoff and the convergence rate of the estimator. To choose the optimal H few methods could be used like Cross validation or Goldenschlugger-Lepski.

5.2 Projection estimator

This method assumes that $f \in L_2(A)$, $A \subset \mathbb{R}^d$.
Let $(\varphi_j)_{j \leq 1}$ be a Hilbert basis of $L_2(A)$ such as Fourier, Legendre or wavelets basis. The projection estimator is defined by

$$\hat{f}_m(x) = \sum_{\|j\| \leq m} \hat{a}_j \varphi_j(x), \quad \hat{a}_j = \frac{1}{n} \sum_{i=1}^{n} \varphi_j(X_i) \tag{18}$$

Just like the previous case, the choice of m is crucial, and methods like cross validation and penalization help choosing the best model.

References

[1] Heather Battey and Han Liu. Smooth projected density estimation, 2014.

[2] Simon Coste. Flow models ii: Score matching techniques, Mar 2025.

[3] Charlotte Dion-Blanc. Nonparametric density estimation, 2024.

[4] Tor Fjelde, Emile Mathieu, and Vincent Dutordoir. An introduction to flow matching, January 2024.

[5] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023.

[6] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code, 2024.

[7] Stanislas Strasman, Antonio Ocello, Claire Boyer, Sylvain Le Corff, and Vincent Lemaire. An analysis of the noise schedule for score-based generative models, 2025.