Statistical approach “review”

Samy Braik

April 2025

To learn and sample from an unobserved probability distribution, generative models are the go-to techniques. They more or less learn the true distribution but they are mostly effective at generating new data. Density estimation focuses on the first goal. Although, I argue that we could sample according to a good density estimation. I choose to consider methods with minimal to none assumption on the data.

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 written q. Either directly like the 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. The goal is to
Consider $f : \mathbb{R}^d \to \mathbb{R}^d$, called a normalizing flow, an invertible and differentiable function and define $X_1 = f(X_0)$. We are able to determine p, in terms of q,

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

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

Since the data could be highly non-Gaussian in nature, such a transformation f is intractable.
Therefore the goal is to learn f_θ, approximation of f, such that $x_1 \simeq f_\theta^{-1}(x_0)$.
A structure is imposed to f_θ, we define $f_1 \dots f_k$ simpler function, such that

$$f_\theta = f_k \circ f_{k-1} \circ \dots \circ f_2 \circ f_1 \tag{3}$$

There is then,

$$x_0 \sim p_0 = q, \quad f_1(x_0) = x_1 \implies x_1 \sim p_1, f(x_1) = x_2 \dots f(x_{k-1}) = x_k \sim p_k = \hat{p} \simeq p \tag{4}$$

The objective function is the maximum log-likehood of the data

$$\mathcal{L}_{\text{NF}}(\theta) = -\sum_{i=1}^{N} \left[\log q(f_\theta^{-1}(x^i)) + \sum_{k=1}^{K} \log \left| \det \frac{\partial f_k^{-1}}{\partial x_k}(x^i) \right| \right] \tag{5}$$

Normalizing flows requires invertibility of the mappings and an efficient way to compute the determinant of there Jacobian. Therefore, components have to be chosen carefully.

1.2 Flow

A C^r flow is a time-dependent mapping $\phi : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ implementing $\phi(t,x) \to \phi_t(x)$ such that for all $t \in [0,1]$, ϕ_t is a C^r diffeomorphism in x. We define a flow model by applying a flow ϕ_t to the random value X_0

$$X_t = \phi_t(X_0), \quad t \in [0,1], X_0 \sim p \tag{6}$$

Alternatively, we can define a flow using a velocity field $v_t : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ implementing $v : (t,x) \to v_t(x)$ via the following ODE

$$\partial_t \phi_t(x) = v_t(\phi_t(x)) \tag{7}$$

$$\phi_0(x) = x \quad \text{initial condition} \tag{8}$$

We can derive a probability path as the marginal PDF of a flow model 6 at time t by $X_t \sim p_t$. This PDF is obtained by a push-forward formula

$$p_t(x) = p(\phi^{-1}(x))|\det \partial_x \phi^{-1}(x)| \tag{9}$$

Knowing v_t allow us to generate p_t.

1.3 Flow matching

Using the notions defined in the previous section. The Flow Matching framework is as follow: We have a known source distribution q and an unkown target distribution p, we want to retrieve the probability path p_t interpolating from $p_0 = q$ to $p_1 = p$. Therefore we need to learn a velocity field v_t^θ (a neural network) to generate such a path, and by solving the ODE 7 sample according to p (approximation). In order to learn v_t^θ the loss to minimize is

$$\mathcal{L}_{\mathrm{FM}}(\theta) := \mathbb{E}[\|v_t(X_t) - v_t^\theta(X_t)\|^2] = \mathbb{E}[\|v_t^\theta(X_t) - \dot{X}_t\|^2] + c \tag{10}$$

where $c = \mathbb{E}[\|\dot{X}_t\|^2] - \mathbb{E}[\|v_t(X_t)\|^2]$ constant with respect to θ.
There is no constraint on the neural network and the invertibility needed in 9 is due to optimal transport argument.

1.4 Diffusion

The general framework of diffusion is divided in two phases. We start from a random variable distributed according to our target distribution p, add noise until it reaches an easy-to-sample distribution q which is practically always a Gaussian. Then we denoise from q to get back to p.

We consider $T \in \mathbb{N}^*$, 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

$$d\overrightarrow{X}_t = \frac{-\beta(t)}{2\sigma^2}\overrightarrow{X}_t dt + \sqrt{\beta(t)} dB_t, \quad \overrightarrow{X}_0 \sim p \quad \text{Forward process} \tag{11}$$

$$d\overleftarrow{X}_t = \left(\frac{\beta(T-t)}{2\sigma^2}\overleftarrow{X}_t + \beta(T-t)\nabla \log p_{T-t}\left(\overleftarrow{X}_t\right)\right) dt + \sqrt{\beta(T-t)} dB_t, \quad \overleftarrow{X}_0 \sim p_T \quad \text{Backward process} \tag{12}$$

The thing is we only noise the random variable until a finite time T therefore $p_T \neq q$ but with a good choice of T and β, we can hope that $p_T \simeq p$. 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

$$\begin{aligned} \mathbb{E}[|\nabla \log p_t(X+\varepsilon) - s(X+\varepsilon)|^2] &= c + \mathbb{E}[|\nabla \log g(\varepsilon) - s(X+\varepsilon)|^2] && (13) \\ &= c + \mathbb{E}[|(-\varepsilon/\mathrm{Var}(\varepsilon))g(\varepsilon) - s(X+\varepsilon)|^2] && (14) \end{aligned}$$

with c a constant not related to s.

With a good choice of neural network s_θ (data dependent) and noise schedule, we can generate using the backward process.

2 Nonparametric density estimation

Another approach that could be useful in our situation is density estimation. Consider a dataset $(X_1, \dots, X_n)$ all having the same density f. The goal is to estimate f.

2.1 Kernel estimator

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

$$\hat{f}_H(x) := \frac{1}{n|\mathrm{H}|^{1/2}} \sum_{j=1}^{n} K\left(\mathrm{H}^{-1/2}(x - X_j)\right) = \frac{1}{n} K_{\mathrm{H}}\left(x - X_j\right) \quad (15)$$

with $K_{\mathrm{H}}(x) := |\mathrm{H}|^{-1/2} K(\mathrm{H}^{-1/2}x)$.
A very 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. To choose the optimal h few methods could be used like Cross validation or Goldenschlugger-Lepski.

2.2 Projection estimator

To build this estimator, we add another assumption which is $f \in L_2(A), A \subset \mathbb{R}$.
Let $(\phi_j)_{j \leq 1}$ an Hilbert basis of $L_2(A)$, the estimator is defined by

$$\hat{f}_m = \sum_{i=1}^{m} \hat{a}_j \varphi_j, \quad \hat{a}_j = \frac{1}{n}\sum_{i=1}^{n} \varphi_j(X_i) \quad (16)$$

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

Nonparametric estimation can be interesting when the number of observations n is big.

References

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

[2] 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.

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