Multimodal base distributions in conditional flow matching generative models: Supplementary material

Shane Josias
josias@sun.ac.za
Willie Brink
wbrink@sun.ac.za

Department of Mathematical Sciences
Stellenbosch University
Stellenbosch, South Africa

Derivation of the conditional vector field

Theorem 3 from Lipman et al. [3] shows that a conditional vector field defining a Gaussian probability path $p_t(\boldsymbol{z}_t \mid \boldsymbol{z}_1) = \mathcal{N}(\boldsymbol{z}_t \mid \boldsymbol{\mu}_1(t), \sigma^2(t)\boldsymbol{I})$ can be defined as

$$\boldsymbol{u}_t(\boldsymbol{z}_t \mid \boldsymbol{z}_1) = \frac{\sigma'(t)}{\sigma(t)}(\boldsymbol{z}_t - \boldsymbol{\mu}_1(t)) + \boldsymbol{\mu}_1'(t), \tag{1}$$

where $\boldsymbol{\mu}_1(t)$ and $\sigma(t)$ indicate that the mean and covariance of the probability path change over time. Our aim is to construct valid probability paths that lead to the standard base distribution, or a class-specific component in a GMM base distribution. Lipman et al. [3] already construct such a path for the standard base distribution. By defining $\boldsymbol{\mu}_1(t) = t\boldsymbol{z}_1$ and $\sigma(t) = 1 - (1 - \sigma_{\min})t$ we have

$$\boldsymbol{\mu}_1'(t) = \boldsymbol{z}_1 \quad \text{and} \quad \sigma'(t) = \sigma_{\min} - 1, \tag{2}$$

which lead to the following conditional vector field:

$$\boldsymbol{u}_t(\boldsymbol{z}_t \mid \boldsymbol{z}_1) = \frac{\sigma_{\min} - 1}{1 - (1 - \sigma_{\min})t}(\boldsymbol{z}_t - t\boldsymbol{z}_1) + \boldsymbol{z}_1 \tag{3}$$

$$= \frac{\boldsymbol{z}_1 - (1 - \sigma_{\min})\boldsymbol{z}_t}{1 - (1 - \sigma_{\min})t}. \tag{4}$$

To construct a valid probability path that leads to a class-specific component in the GMM base, we leave $\sigma(t)$ unchanged and define $\boldsymbol{\mu}(t)$ as a linear interpolation between the data point $\boldsymbol{z}_1$ and its class-specific mean $\boldsymbol{\mu}_k$ as follows:

$$\boldsymbol{\mu}_1(t) = t\boldsymbol{z}_1 + (1 - t)\boldsymbol{\mu}_k \quad \text{and} \quad \sigma(t) = \sigma_{\min} - 1. \tag{5}$$

In order to apply Theorem 3 from Lipman et al. [3], we first have

$$\boldsymbol{\mu}_1'(t) = \boldsymbol{z}_1 - \boldsymbol{\mu}_k \quad \text{and} \quad \sigma'(t) = \sigma_{\min} - 1, \tag{6}$$

which we substitute into the definition for the conditional vector field:

$$\boldsymbol{u}_t(\boldsymbol{z}_t \mid \boldsymbol{z}_1) = \frac{\sigma_{\min} - 1}{1-(1-\sigma_{\min})t}\left(\boldsymbol{z}_t - t\boldsymbol{z}_1 - (1-t)\boldsymbol{\mu}_k\right) + \boldsymbol{z}_1 - \boldsymbol{\mu}_k \tag{7}$$

$$= \frac{\boldsymbol{z}_1 - \sigma_{\min}\boldsymbol{\mu}_k - (1-\sigma_{\min})\boldsymbol{z}_t}{1-(1-\sigma_{\min})t}. \tag{8}$$

This vector field corresponds to probability paths between a density concentrated around the data $\boldsymbol{z}_1$ and the GMM component corresponding to $\boldsymbol{z}_1$'s assigned class, with mean $\boldsymbol{\mu}_k$.

Implementation details

For the CFM models, we adapt the implementation from Tong et al. [5], with hyperparameter tuning on the batch size and learning rate, for both the standard and GMM bases, and additionally the covariance scale for the GMM base. Tables 1 and 2 show final hyperparameter values used. We refer the reader to the original implementation [5] for descriptions of the various hyperparameters. Log-likelihoods and generated samples are computed using the torchdiffeq [1] framework. All models are trained on a single NVIDIA RTX A6000 GPU. The Adam optimiser [2] is used with default PyTorch [4] values for β_1 and β_2, and its learning rate is warmed up with a linear learning rate scheduler.

Table 1: Hyperparameters for CFM models trained with the standard base.

Parameter	MNIST	FashionMNIST	CIFAR10	SVHN
Channels	128	128	128	128
Channels multiple	(1, 2, 2)	(1, 2, 2)	(1, 2, 2, 2)	(1, 2, 2, 2)
Heads	1	1	1	1
Heads channels	1	1	1	1
Attention resolution	16	16	16	16
Dropout	0.0	0.0	0.0	0.0
Batch size	128	256	256	256
Epochs	150	150	150	150
Learning rate (warmed up)	0.0002	0.0002	0.0002	0.0002

Table 2: Hyperparameters for CFM models trained with the GMM base.

Parameter	MNIST	FashionMNIST	CIFAR10	SVHN
Covariance scale	0.6	0.6	0.8	0.8
Channels	128	128	128	128
Channels multiple	(1, 2, 2)	(1, 2, 2)	(1, 2, 2, 2)	(1, 2, 2, 2)
Heads	1	1	1	1
Heads channels	1	1	1	1
Attention resolution	16	16	16	16
Dropout	0.0	0.0	0.0	0.0
Batch size	256	128	256	256
Epochs	150	150	150	150
Learning rate (warmed up)	0.0002	0.0002	0.0002	0.0002

References

[1] Ricky T.Q. Chen. torchdiffeq: PyTorch implementation of differentiable ODE solvers, 2018. URL https://github.com/rtqichen/torchdiffeq.

[2] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[3] Yaron Lipman, Ricky T.Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. *International Conference on Learning Representations*, 2023.

[4] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 2019.

[5] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024.