Imperial College London

MEng Individual Project

Imperial College London

Department of Mathematics

Department of Computing

Investigating Generalisation in Score-Based Generative Models

Author:
Peter Christofides Paton

Supervisor:
Dr Ömer Deniz Akyıldız

Second marker:
Dr Nikolas Kantas

June 21, 2023

Abstract

Score-based generative models (SGMs) achieve state-of-the-art generalisation on many research and industry standard datasets. As a result, formulating theoretical results which explain this impressive empirical performance is currently a highly active area of research. Specifically, we will be considering de-noising diffusion models, which noise data via a forward process and generate new samples by applying the reverse diffusion. Most of the analysis in the literature ensures convergence by relying on the assumption the forward perturbation process can be run arbitrarily forward in time. While this holds in theory, it is infeasible in practice, particularly in developing general diffusion model libraries.

In this work, we investigate how the parameters of the forward process can be optimally determined to maximise the quality of generated samples, over a fixed time interval. We discuss this in the context of ensuring the convergence of the forward process while mitigating the discretisation error caused by approximating the backward diffusion. Additionally, we investigate how applying regularisation techniques in training the score-network affects generalisation capabilities. We apply our insights to synthetic datasets that lie on disjoint latent supports, which is hypothesised for many real-world datasets. We derive forward convergence results specifically under this setting and compare our results to push-forward generative models (such as generative-adversarial networks and variational auto-encoders), which are known to suffer intrinsic drawbacks in generating samples lying exclusively on the support of such datasets. We experimentally show that SGMs outperform push-forward models under this setting and provide mathematical justifications as to why score-based models do not suffer such drawbacks.

Acknowledgements

Firstly, I would like to thank my supervisor, Deniz, for the time he has dedicated to this project and most importantly for teaching me the research process is never linear. I would also like to thank him for introducing me to the wider world of generative modelling research as our meetings at the Turing Institute and associated conferences were some of my highlights from this academic year.

I would also like to thank Nikolas Kantas for agreeing to be the second marker on this project and for engaging in discussions on particle interacting perturbation processes.

I would like to thank Benjamin Boys from the University of Cambridge for sharing his diffusion Jax library with me and Grigorios Pavliotis for helping me understand results from stochastic calculus beyond the scope of his module at Imperial.

Finally, thank you Tjasa for all your support throughout my degree and to the amazing people I've met during my time at Imperial.

Contents

Chapter 1

Introduction

Generative modelling is a form of unsupervised learning which aims to generate novel samples indistinguishable in distribution from an initial dataset. Generative models have achieved impressive results in variety of applications such as noise removal [1], text prediction [2] and more recently in natural language generation through the use of large language transformer models [3][4]. While generative models come in many forms, in this project we focus primarily on score-based generative models, specifically de-noising diffusion models.

Fundamentally, these models comprise of two related stochastic processes, a forward perturbation process which diffuses the training set into noise and the corresponding time-reversal process which transforms noise back into plausible samples. Accounting to its rich literature, the forward process is conventionally taken to be a stochastic differential equation, specifically the Ornstein-Ulhenbeck process, which we introduce in Section 2. Formulating the backward process requires knowledge of the so-called 'score function' of the data under forward perturbation.[1] Without knowing the true data distribution from which we wish to sample, the true score is intractable. However, we can learn an approximation by considering a model score function and reducing a quantity known as a 'score-matching objective' [5]. This is the standard construction of score-based generative models which we will be considering.

SGMs are renowned for their ability to appropriately generalise beyond the training set, which we demonstrate in Section 3.2, although the literature is still catching up in establishing concrete mathematical justifications for their remarkable generalisation capabilities. One result of particular relevance to this project provides an upper-bound on the error between the true data distribution and generated samples under the time-reversal process. This upper-bound is comprised of three terms which correspond to the error in converging to white noise in the forward process, a discretisation error from approximating the continuous reverse process and the error incurred by approximating the true score function with a learned model.

1.1 Objectives

Much of the theoretical reasoning supporting the generalisation capabilities of score-based models rely on the assumption that the forward perturbation process can be run as long as required for the perturbed samples to sufficiently resemble white noise. While this is a reasonable assumption in theory, it is poorly suited to real applications of diffusion models, particularly in writing general packages which perform score-based modelling on arbitrary input datasets. This assumption often has the undesired effect of restricting the analysis to only considering an Ornstein-Ulhenbeck (OU) perturbation process of a very specific form. In truth however, the parameters of the OU process are free to be determined.

We believe it is more realistic to restrict the time interval of the forward and backward process to a fixed finite interval, which we take to be $[0, 1]$. This has the added benefit of not requiring discretisation over an arbitrarily large time interval. We primarily aim to generalise the bound on the backward convergence error to arbitrary OU processes over this fixed interval. In doing so, we

[1] The score of a distribution with density $p(x)$ is defined as $\nabla_x \log p(x)$.

investigate how varying these free parameters affects the rate of convergence of the forward process and discretisation error in approximating the reverse diffusion.

Additionally, score-based models are susceptible to overfitting. In the context of generative modeling, this corresponds to the generated samples lying exactly on the support of the training set. We investigate how applying standard machine learning regularisation techniques prevents overfitting. Theoretically, this analysis corresponds to minimising the error caused in approximating the true score function.

To the best of our knowledge, there is no discussion in the literature on the capabilities of score-based generative models to learn and generalise on datasets lying on disjoint latent supports. It has been theorised in the literature that push-forward models cannot adequately reproduce samples with disjoint support. We aim to conclude by using our results to motivate explanations as to whether score-based models can succeed in modelling such distributions.

1.2 Contributions

We summarise our contributions below:

1. Formalised the notion of a generalisation metric in score-based generative modelling and provided theoretical and experimental justifications as to why existing metrics in the literature abide by this formalisation. We also developed some novel metrics for specific datasets considered in experiments.

2. Derived an upper-bound on the forward convergence error over a fixed time interval, in terms of the free parameters of the Ornstein-Ulhenbeck process - under some mild assumptions on the moments of the dataset. We demonstrate the relationship between parameters and forward convergence error holds for Gaussian mixture distributions and a proxy distribution with disjoint latent support.

3. Provided theoretical and empirical justifications on the relationship between the free parameters of the OU process and discretisation error over a finite time interval.

4. Combined the above two contributions together to reformulate the upper bound on the error of the backward process to general OU perturbation processes. Demonstrated this relationship holds on a variety of synthetic and research standard datasets.

5. Compared this result to existing theorems in the literature relating to push-forward models and presented justifications as to why SGMs don't suffer the same drawbacks in generating samples on Gaussian mixture distributions.

6. Explored how applying regularisation techniques in the training process of the model score affects the quality of generated samples, specifically we consider early stopping and the capacity of the score-network.

7. Explained theoretically the generalisation capabilities of SGMs on synthetic examples in accordance with the union of manifolds hypothesis. Compared this success to existing results in the literature relating to push-forward models.

1.3 Report Structure

In this report, we introduce necessary preliminaries in stochastic calculus and score-based generative models in Chapter 2. In the related work section we introduce the manifold hypothesis' which motivate the synthetic datasets used in our experiments and introduce an alternative generative modelling architecture called 'push-forward' models, which we compare to SGMs throughout the project. In this chapter, we also present the primary result from the literature which we aim to generalise to arbitrary perturbation processes. In Chapter 4 we formalise generalisation and justify theoretically why our chosen generalisation metrics from the literature abide by this definition. We present our theoretical and experimental results in Chapter 5, these results are evaluated as they are presented but we also summarise our findings in Chapter 6. Final conclusions and avenues for future work are discussed in Chapter 7. Theoretical proofs and details of the various neural network architectures and training processes are documented in the Appendix.

Chapter 2

Background

Firstly, we introduce some relevant preliminaries that will be referred to extensively throughout this report. This discussion is very thorough, therefore readers who are already familiar with stochastic differential equations and de-noising diffusion models can proceed directly to the 'Related Work' chapter (Chapter 3).

2.1 Mathematical Preliminaries

2.1.1 Stochastic Processes

A stochastic process a system that evolves forward in time in an inherently probabilistic manner. More rigorously, it is a collection of random variables which define the distribution of the process over an ordered set. For the purpose of this report, the random variables will be indexed by either continuous time as $\{X_t\}_{t\geq 0}$ or in discrete time as $\{X_{t_i}\}_{i\in N_0}$. In theory, time is taken to be continuous however this is practically discretised to find numerical approximations of the system.

2.1.2 Brownian Motion

Our first stochastic process is *standard* Brownian motion (also referred to in the literature as the *standard* Wiener process). This is a random variable defined over continuous time $t \in [0, T]$. Notationally, any instance of Brownian motion will be denoted throughout this report by W_t. Brownian motion satisfies the following properties [6, Chapter 2],

- $W_0 = 0$ almost surely.
- $\forall t, s \in [0, T]$ and $s < t$ w.l.o.g., $W_t - W_s \sim N(0, t - s)$.
- The increments $\{W_{t_1} - W_{t_0}, W_{t_2} - W_{t_1}, \ldots, W_{t_k} - W_{t_{k-1}}\}$ are mutually independent for $\forall k \in \mathbf{N}$ and $\forall t_i \in [0, T]$ with $0 \leq t_1 < t_2 < \cdots < t_k \leq T$.

From this definition is is quite easy to show, by considering the sum of independent Gaussian variables, $W_t \sim N(0, t) \overset{d}{=} \sqrt{t}N(0, 1)$.

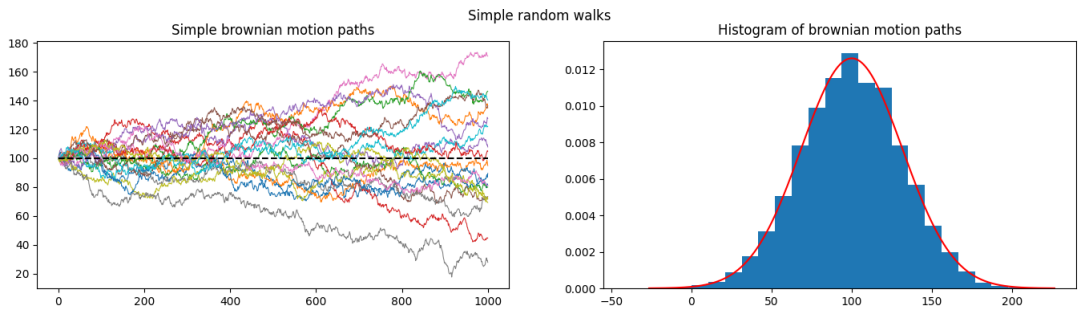

Figure 2.1: Some simulations of a simple system with initial value of 100 which varies over time following the Wiener process, so $X_t = 100 + W_t$. The right hand diagram shows the expected distribution of sample paths (red) against the histogram of $10,000$ realisations at time $T = 1,000$.

Brownian motion is a universal stochastic process with a wide breadth of applications. It is especially prominent in finance as it adopts both the Markov and martingale properties, which are both common assumptions about the dynamics of many financial instruments. Additionally, it can be shown sample paths of Brownian motion are almost surely continuous, although nowhere differentiable[1] [7, Theorem 1.3, p.g. 12].

2.1.3 Stochastic Differential Equations

Stochastic differential equations are differential equations which have an additional stochastic process term. Initially used in molecular process modelling [8], stochastic differential equations have found applications in numerous fields, including the famous Black-Scholes equation for options pricing [9, Chapter 4.5]. More generally, they can be used to model any process for which small complex behaviour can be approximated by random perturbations in the system. In this report we will be considering stochastic differential equations driven by Brownian motion, given by the following general form:

$$dX_t = f(X_t, t)dt + \sigma(X_t, t)dW_t. \tag{2.1}$$

Here, dW_t denotes the Wiener process over a small time dt, $dW_t \sim N(0, dt)$. $f(X_t, t)$ is often referred to as the 'drift' component and $\sigma(X_t, t)$ the 'diffusion' component. The above formulation (2.1) is mathematical shorthand for the following the integral equation,

$$X_{t+dt} - X_t = \int_t^{t+dt} f(X_u, u)du + \int_t^{t+dt} \sigma(X_u, u)dW_u \tag{2.2}$$

where the first integral is a standard Lebesgue/Riemann-Stieltjes integral and the second is a stochastic integral, which are discussed below.

Example: The Ornstein-Ulhenbeck Process

A notable case of (2.1) is the Ornstein-Ulhenbeck process,

$$dX_t = -\beta(X_t - \alpha)dt + \sigma dW_t \tag{2.3}$$

where β measures how quickly the system approaches the asymptotic mean α and we have a constant noise component σ. Additionally, providing $\beta > 0$, Ornstein-Ulhenbeck processes are '*mean reverting*', which means they will converge in expectation to α as $t \to \infty$. This can be seen from the analytic solution to an OU process, which is the unique invariant distribution of the process (this is derived in the Appendix A.2.1).

$$X_t | X_0 = x_0 \sim N\left(x_0 e^{-\beta t} + \alpha(1 - e^{-\beta t}), \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})\right) \xrightarrow{\text{d}} N(\alpha, \frac{\sigma^2}{2\beta}).$$

2.1.4 Stochastic Integration

Stochastic calculus seldom follows the rules and intuition of conventional calculus. In constructing the Riemann integral, we partition the interval $[0, T]$ with $P_N := \bigcup_{i=0}^{N}[t_i, t_{i+1}]$ and define $|P_N| := \max\{\Delta t_i := t_i - t_{i-1} \mid i = 1, \ldots, N\}$. The Riemann integral is then defined on the uniform partition $P := \bigcup_{i=0}^{N}[\frac{iT}{N}, \frac{(i+1)T}{N}]$ as limit of the sum:

$$\int_0^T f(u)du = \lim_{|P_N| \to 0} \sum_{i=1}^{N} f(t_i)(t_i - t_{i-1}). \tag{2.4}$$

We have a similar formulation for the stochastic integral with respect to the Wiener process,

$$\int_0^T f(u)dW_u = \lim_{|P_N| \to 0} \sum_{i=1}^{N} f(t_i)(W_{t_i} - W_{t_{i-1}}). \tag{2.5}$$

[1] Intuitively, the lack of differentiability follows from the Markov property, as the derivative acts as a velocity of the process.

The 'upper Riemann sum' presented in (2.4) has been shown to be equivalent the alternative midpoint formulation:

$$\int_0^T f(u)du = \lim_{|P_N|\to 0} \sum_{i=1}^{N} f\left(\frac{t_i + t_{i-1}}{2}\right)(t_i - t_{i-1}). \tag{2.6}$$

The corresponding midpoint form of the stochastic integral is:

$$\int_0^T f(u) \circ^{\frac{1}{2}} dW_u = \lim_{|P_N|\to 0} \sum_{i=1}^{N} f\left(\frac{t_i + t_{i-1}}{2}\right)(W_{t_i} - W_{t_{i-1}}). \tag{2.7}$$

From the definition of Brownian motion $\Delta W_{t_i} := W_{t_i} - W_{t_{i-1}} \sim N(0, \Delta t_i)$. As Brownian motion has unbounded total variation, the stochastic integral is not independent of where the integrand, f, is evaluated between points t_i and t_{i+1}. Therefore, the two formulations ((2.5) and (2.7)) have different solutions, a mismatch which does not diminish as $|P_N| \to 0$. In the literature, (2.5) is known as the 'Itô integral' and (2.7) the 'Stratonovich integral'. As these intepretations have different properties, the choice of evaluating a stochastic integral in the Itô or Stratonovich sense is entirely a modelling issue.

2.1.5 Discretisation Schemes

In some cases, it is possible to find analytic solutions to stochastic integrals. We can then sample them at a given time t to produce sample realisations of the process, at that point in time. In practice however, it is often impractical or impossible to derive a closed form solution to a stochastic integral. In these cases, we aim to numerically approximate the process instead. Our discretisation scheme follows intuitively from the formulation presented in (2.2). Rewriting this we can see,

$$X_t = X_0 + \int_0^t f(X_u, u)du + \int_0^t \sigma(X_u, u)dW_u, \ 0 \le t \le T. \tag{2.8}$$

which by choosing a uniform partition on the above discretisation (2.5) gives,

$$X_{t_{i+1}} = X_{t_i} + f(X_{t_i}, t_i)\Delta t + \sigma(X_{t_i}, t_i)\Delta(W_t) \text{ for } i = 1, ..., N, \Delta t = T/N \text{ and } t_i = i\Delta t. \tag{2.9}$$

This is the 'Euler-Maruyama' discretisation[2]. Δt is referred to as the 'step-size'. Furthermore, a numerical approximation, $\hat{X}_j^{\Delta t}$, of the stochastic process, X_t, has a strong order of convergence α if there exists a constant $C > 0$ independent of Δt such that

$$\mathbb{E}\left[\left|X_{j\Delta t} - \hat{X}_j^{\Delta t}\right|\right] \le C\Delta t^{\alpha}. \ \forall j = 1, \ \dots \ , N, \text{ and } \Delta t \text{ sufficiently small.} \tag{2.10}$$

The Euler-Maruyama method has strong order of convergence of $\frac{1}{2}$ [7, p.g. 148]. There exists methods with greater order of convergence, such as the Milstein scheme ($\alpha = 1$), however this will not be relevant to this project. For the Ornstein-Ulhenbeck process, the two schemes are actually equivalent as the diffusion term is independent of X_t and therefore the Euler-Maruyama scheme attains an order of convergence of $\alpha = 1$ in approximating OU sample paths.

2.1.6 Time Reversal Of Stochastic Differential Equations

As with many stochastic processes [10], stochastic differential equations have a time-reversal process, which describes the dynamics of the system backwards in time from the current state X_T. For a stochastic differential equation (2.1), $\forall T > 0, \forall t \in [0, T]$, the time-reversal process is defined as [11],

$$dX_{T-t} = [-f(X_{T-t}, T-t) + \sigma(X_{T-t}, T-t)\left[\nabla_x \log(p_{T-t}(x))\right]\sigma(X_{T-t}, T-t)^T]dt + \sigma(X_{T-t}, T-t)dW_t \tag{2.11}$$

It's worth noting the presence of the 'score-function', $\nabla_x \log p(x)$, of the marginal $p(x)$ at time $T - t$ in the time-reverse process. The drift term can be interpreted intuitively in the following

[2]This was derived here using the Itô interpretation (2.5) however there is an analogous result using the Stratonovich integral in a similar manner.

sense: $-f(X_{T-t}, T-t)$ acts a reverse dispersion in the direction of the initial distribution and the score function contributes to a gradient ascent of the log probability density as $t \to T$ in the reverse process. This idea forms the basis of modern score-based generative modelling methods. If we know the score function and stationary distribution of the forward process, we can reverse the forward perturbation to generate samples from the initial data distribution. Of course, these are unrealistic assumptions, as if the score-function at all times t is known we can simply sample from p_0 directly. Nevertheless, an example using the analytic score to generate samples from a Gaussian mixture distribution is presented in Figure 2.2.

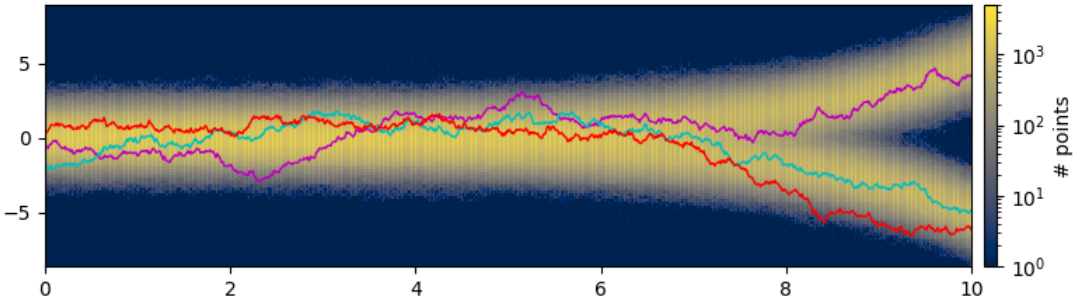

Figure 2.2: This figure shows the time reversal of $10,000$ particles under an Ornstein-Ulhenbeck process with parameters are $\beta = \frac{1}{2}$, $\alpha = 0$, $\sigma = 1$ to generate samples from a one-dimensional Gaussian mixture prior given by: $p_0(x) = \frac{1}{2}\mathcal{N}(-5, 1) + \frac{1}{2}\mathcal{N}(5, 1)$. The coloured background is a histogram of the particles backwards in time, with lighter yellow indicating greater particle density. We can also see the visualisation of 3 sample paths under the backward process overlayed on top of the histogram.

2.2 Score-Based Generative Models

In this section, I will first introduce the theory behind score-based models and describe how the score-function can be used to generate new samples using a Markov chain Monte Carlo (MCMC) method known as Langevin dynamics. This process suffers some notable set-backs, which are addressed by iteratively perturbing the initial samples in discrete time steps. As the discretisation approaches continuous time this perturbation resembles a diffusion process. With this formulation we arrive at state-of-the-art score-based models, which will serve as the foundational model on which we will explore generalisation properties.

2.2.1 Likelihood-Based Models

An intuitive approach to generative modelling would be to explicitly model the underlying data distribution and employ standard sampling techniques to generate new samples from the model density. This task can be formalised as fitting a parameterised density $p_\theta(x)$ to samples $x \in \mathbb{R}^n$ which come from an unknown parent distribution $p_{data}(x)$. This is referred to as 'likelihood estimation' as it aims to estimate the optimal set of parameters, θ^*, which maximise the probability of the observed samples occurring under our proposed model density. As the logarithm function is a monotone, it is often easier to determine when the log-likelihood is maximised instead. Therefore, the model density is often defined in the following form,

$$p(x; \theta) \propto \exp\left(-E(x; \theta)\right). \tag{2.12}$$

Here $E(x; \theta)$ is some function parameterised by $\theta \in \mathbb{R}^m$ and commonly referred to as the '*energy function*'. Formally, this estimation is constructed as in (2.13) and usually entails differentiating the likelihood function $L(\{X_i\}) = \prod_{i=1}^N p_\theta(x_i)$ with respect to θ.

$$\hat{\theta} = \underset{\theta \in \mathbb{R}^m}{\operatorname{argmax}} \prod_{i=1}^{N} p(x_i; \theta) = \underset{\theta \in \mathbb{R}^m}{\operatorname{argmax}} \sum_{i=1}^{N} \log p(x_i; \theta) \tag{2.13}$$

However, as the normalising factor is a function of the parameters θ we aim to minimise, likelihood estimation requires the normalised density[3]. Clearly, $\bar{p}_\theta(x)$ can be normalised for a general energy function,

$$p_\theta(x) = \frac{\bar{p}_\theta(x)}{Z(\theta)}, \text{ where: } Z(\theta) = \int_{\mathbb{R}^n} \bar{p}_\theta(x)dx < \infty \text{ and}$$
$$\bar{p} \text{ denotes the unnormalised density function.}$$

Evidently, this requires the integral to be analytically tractable, or at least feasibly approximated. This is the primary issue with likelihood-based models as the normalising term $Z(\theta)$ is often intractable for a general energy function. Furthermore, numerical approximations have been shown to deteriorate in higher dimensions too [5]. Therefore a likelihood-based approach is quite restrictive on our choice of energy function and therefore overall model. To circumvent this, we can attempt to approximate the likelihood function instead [13, Section 3] - the challenge however lies being in approximating $-\nabla_\theta \log Z(\theta)$.

$$\begin{aligned}
\nabla_\theta \log p(x;\theta) &= \nabla_\theta \log \bar{p}(x;\theta) - \nabla_\theta \log Z(\theta). \\
&= -\nabla_\theta E(x;\theta) - \nabla_\theta \log \int \exp(-E(x;\theta))dx \\
&= -\nabla_\theta E(x;\theta) - \nabla_\theta \left(\int \exp(-E(x;\theta))dx\right)\left(\int \exp(-E(x;\theta))dx\right)^{-1} \\
&= -\nabla_\theta E(x;\theta) - \nabla_\theta \left(\int \exp(-E(x;\theta))dx\right)(Z(\theta))^{-1} \\
&= -\nabla_\theta E(x;\theta) - \left(\int -(\nabla_\theta E(x;\theta))\exp(-E(x;\theta))dx\right)(Z(\theta))^{-1} \\
&= -\nabla_\theta E(x;\theta) - \left(\int -(\nabla_\theta E(x;\theta))\frac{\exp(-E(x;\theta))}{Z(\theta)}dx\right) \\
&= -\nabla_\theta E(x;\theta) - \left(\int -(\nabla_\theta E(x;\theta))p(x;\theta)dx\right) \\
&= -\nabla_\theta E(x;\theta) - \mathbb{E}_{x\sim p_\theta(x)}\left[-\nabla_\theta E(x;\theta)\right]
\end{aligned}$$

Therefore, providing we can sample from our model density, for any θ, we can employ a Monte Carlo estimator of $-\nabla_\theta \log Z(\theta)$,

$$-\nabla_\theta \log Z(\theta) = \mathbb{E}_{x\sim p_\theta(x)}\left[-\nabla_\theta E(x;\theta)\right] \approx \frac{1}{M}\sum_{j=1}^{M}\left[-\nabla_\theta E(x_j;\theta)\right], \text{ where } x_j \sim p_\theta(x).$$

We will denote this estimator as $\log \hat{Z}(\theta)$. This provides us with an algorithmic approach to determining θ^* via gradient ascent on the modelled likelihood function,

$$\begin{aligned}
\theta_k &= \theta_{k-1} + \gamma\nabla_\theta \log L(x^N;\theta_{k-1}) \\
&= \theta_{k-1} + \gamma\nabla_\theta \sum_{i=1}^{N} \log p_{\theta_{k-1}}(x) && x_i \sim p_{data}(x) \\
&= \theta_{k-1} + \gamma\sum_{i=1}^{N} \nabla_\theta \log p_{\theta_{k-1}}(x_i) = \theta_{k-1} + \gamma\sum_{i=1}^{N} \nabla_\theta E(x_i;\theta_{k-1}) - \nabla_\theta \log Z(\theta) \\
&= \theta_{k-1} + \gamma\sum_{i=1}^{N} \nabla_\theta E(x_i;\theta_{k-1}) - \gamma\frac{1}{M}\sum_{j=1}^{M} \nabla_\theta E(x_j;\theta_{k-1}), && x_j \sim p_{\theta_{k-1}}(x).
\end{aligned}$$

Therefore, we can perform an iterative gradient descent on our N samples at x_k to use in estimation of $\log Z(\theta)$ and then performing an update step on θ_k this continues back and forth until we have convergence for θ_k. This process specifies what are known as 'constrastive divergence' methods, for which a general algorithm is presented below (1). Unfortunately, these methods suffer from convergence issues in finding $\hat{\theta}$, as the estimator is biased [14, Section 5].

[3]With an un-normalised density we usually employ MCMC methods such as Metropolis Hastings or Gibbs sampling. These methods too have their limitations, for example the Monte Carlo random walk has been shown to poorly explore the state space, particularly in high dimensions [12].

Algorithm 1 MLE of $\hat{\theta}$ via contrastive divergence

$X_0^M \leftarrow\sim p_0(x)$ ▷ Initialise M samples according to some initial distribution $p_0(x)$
$\theta_0 \leftarrow p(\theta)$ ▷ Initialise θ_0 from some prior
$X^{M*} \leftarrow$ samples from $p_{data}(x)$
while θ_k not converged **do**
  $i \leftarrow 0$
  while $i < M$ **do**
    $X_k^i \leftarrow X_{k-1}^i + \gamma_x \nabla_x \log p(X_k^i; \theta_{k-1}) + \sqrt{2\gamma} W_t$ ▷ Langevin dynamics on our samples X_K
  end while
  $\theta_k \leftarrow \theta_{k-1} + \gamma - \left(\nabla_\theta \sum_{i=1}^N E(X_k^i; \theta_{k-1}) - \frac{1}{M} \sum_{j=1}^M \nabla_\theta E(X^{M*}; \theta_{k-1})\right)$
  ▷ Gradient descent with estimator for $\log Z(\theta)$
end while

2.2.2 Explicit Score Matching

Score-based models avoid the difficulties in approximating intractable normalising constants by instead modelling the gradients of the data distribution. That is, the model S is given by $S(x;\theta) := \nabla_x \log p_\theta(x)$ aims to approximate the true data score function, $\nabla \log p(x)$. In 2005, Hyvarinen introduced the first instance of a score-based model called '*Explicit score matching*' [5, Section 2]. With our model density $p_\theta(x)$, it follows our model score function is given by,

$$S(x,\theta) = \begin{bmatrix} S_1(x,\theta) \\ \dots \\ S_n(x,\theta) \end{bmatrix} = \begin{bmatrix} \frac{\partial \log p_\theta(x)}{\partial x_1} \\ \dots \\ \frac{\partial \log p_\theta(x)}{\partial x_n} \end{bmatrix} = \nabla_x \log p_\theta(x) \approx \nabla_x \log p(x).$$

Importantly, the score function does not depend on the normalising constant $Z(\theta)$,

$$\nabla_x \log p_\theta(x) = \nabla_x \log \frac{\bar{p}_\theta(x)}{Z(\theta)} = \nabla_x \log \bar{p}_\theta(x) - \nabla_x \log Z(\theta) = \nabla_x \log \bar{p}_\theta(x).$$

Considering the model density of an exponential form (2.12), as we no longer require the normalising constant to be calculated (or even approximated) we can consider a much wider set of proposed energy functions $E(x;\theta)$. In this case, it also clearly follows that we have $S(x;\theta) = -\nabla_x E(x;\theta)$. In a manner similar to parameter estimation in likelihood-based models, Hyvarinen suggests we determine the optimal model score function by finding the set of parameters $\hat{\theta}$ which minimises the Fisher divergence[4] between the true score of the data $\nabla_x \log p(x)$ and the model score $S(x,\theta)$, which is given by,

$$J_{ESM}(\theta) := \frac{1}{2}\mathbb{E}_{p(x)}\left[||\nabla_x \log p(x) - S(x,\theta)||_2^2\right] = \frac{1}{2}\int_{\mathbb{R}^n} p(x)||\nabla_x \log p(x) - S(x,\theta)||_2^2 dx.$$

Where $||\cdot||_2$ denotes the Euclidean norm (L^2-norm) in $\mathbb{R}^n$. This is the first instance of a '*score-matching objective*' - a quantity we aim to minimise in modelling the score function. Our optimal score function is therefore parameterised by the $\theta^* \in \mathbb{R}^m$ such that,

$$\theta^* = \underset{\theta^* \in \mathbb{R}^m}{\operatorname{argmin}}\ J_{ESM}(\theta).$$

This is explicit score matching ('*ESM*') as it requires knowing the true score $\nabla_x \log p(x)$ of the parent distribution - the very quantity we're aiming to approximate with $S(x;\theta)$.

2.2.3 Implicit Score Matching

Score-matching objectives are 'equivalent' if they are minimised for the same value of θ. This is sometimes denoted in the literature as $J_1(\theta) \backsim J_2(\theta)$. As we will see throughout this section, finding equivalent score-matching objectives is a very powerful tool in score-based modelling as it enables faster, more accessible and more accurate searches across the parameter space. Fortunately,

[4] The 'Fisher divergence' between two distributions p and q is defined as $\mathbb{E}_{p(x)}[||\nabla_x \log p(x) - \nabla_x \log q(x)||^2]$.

in the case of explicit score matching, Hyvarinen derived an equivalent score matching objective which does not require knowledge of the true score $\nabla_x \log p(x)$ [5, Section 2, Theorem 1].

$$\begin{aligned} J_{ESM}(\theta) &= \mathbb{E}_{p(y)} \frac{1}{2}[||\nabla_y \log(p(y)) - S(y,\theta)||^2] \\ &= \underbrace{\int_{\mathbb{R}^n} p(y) \sum_{j=1}^{N} \frac{\partial S_j(y,\theta)}{\partial y_j} + \frac{1}{2}[S_j(y,\theta)]^2 dy}_{J_{ISM}(\theta)} + \text{constant} \end{aligned} \tag{2.14}$$

Where y_j refers to the j^{th} component of the variable being integrated over, namely y, it is important to distinguish that it does not refer to a sample. This constant term is independent of θ and therefore can be ignored in finding the set of parameters $\hat{\theta}$ which minimises the Fisher divergence, that is $J_{ESM} \backsim J_{ISM}$. Finally, we can use the samples $\{x_i\}$ in a Monte Carlo estimator for the quantity $J_{ISM}(\theta)$,

$$\begin{aligned} \tilde{J}_{ISM}(\theta) &= \frac{1}{T} \sum_{i=1}^{T} \sum_{j=1}^{N} \partial_j S_j(x_i;\theta) + \frac{1}{2}[S_j(x_i;\theta)]^2 \\ &= \frac{1}{T} \sum_{i=1}^{T} \left[-\operatorname{Tr}\left[\nabla_x^2 [E(x_i;\theta)]\right] + \sum_{j=1}^{N} \frac{1}{2}[S_j(x_i;\theta)]^2 \right] \end{aligned}$$

As with any Monte Carlo estimator we know this to be a consistent estimator for $J_{ISM}(\theta)$ by the law of large numbers. We can use our estimator $\tilde{J}_{ISM}(\theta)$ to estimate the set of parameters θ which minimise the Fisher divergence and ultimately gives us an 'optimal' model score function $S(x,\hat{\theta})$. See the Appendix C.1, for an example of score matching on the one-dimensional Gaussian.

Drawbacks Of Implicit Score Matching

In practice, calculating the hessian of the log density is an expensive operation which scales poorly on higher dimensional data. In fact, arguments have been presented as to why from a theoretical standpoint it is unlikely that there exists an accurate and efficient algorithm for computing the trace of an arbitrary function's hessian [15, Section 6]. Methods have been proposed to mitigate this, such as 'sliced-score matching' which approximates the trace of $\nabla_x S(x;\theta)$ by projecting the scores onto random vectors [16].

2.2.4 Unadjusted Langevin Dynamics

Langevin dynamics are an Markov chain Monte Carlo method which generates samples from $p(x)$ by the following iterative process [17],

$$X_{t+1} = X_t + \gamma \nabla_x \log p(x) + \sqrt{2\gamma} Z, \text{ where } Z \sim N(0, I), \gamma \text{ small.}$$

X_0 is drawn from a prior distribution $p_0(x)$. After an initial 'burn-in' period the produced samples will follow the distribution defined by $p(x)$. Intuitively, Langevin dynamics can be interpreted as a noisy gradient ascent of the log density. As Langevin dynamics only accesses the density through the score function, we can approximate this process through our score model $S(x,\hat{\theta})$. By combining our estimator for the score function and Langevin dynamics, we have the first candidate for a generative score-based model, which I will demonstrate in an example below.

Example: Generative Score-Based Modelling Via Langevin Dynamics On A Two Dimensional Gaussian Mixture Model

In this example, I will demonstrate how to generate new samples from an unknown Gaussian mixture using the methods described above. Firstly, we define our un-normalised model density as follows,

$$p(x,\theta) = \frac{1}{2} \exp(-\frac{1}{2}(x - M_1)^T S_1 (x - M_1)) + \frac{1}{2} \exp(-\frac{1}{2}(x - M_2)^T S_2 (x - M_2))$$

where $M_i \in \mathbb{R}^2$ and $S_i \in \mathbb{R}^{2\times 2}$. Therefore our model score function is,

$$S(x,\theta) = \nabla_x \log\left(\sum_{i=1}^{2} \exp -V_i(x)\right), \text{where } V_i(x) = \frac{1}{2}(x-M_i)^T S_i (x-M_i).$$

Instead of finding the optimal parameters analytically, I will instead take a more practical approach and use gradient descent to find the set of parameters which minimises our estimator of the score matching objective: $\tilde{J}_{ISM}(\theta)$. This estimate is of course a random variable, as it is a function of the random input data samples. With this estimation of the optimal set of parameters, $\hat{\theta}$, we can perform sample generation using Langevin dynamics, substituting the true score function $\nabla_x \log p(x)$ with our model score $S(x,\hat{\theta})$.

$$X_{t+1} = X_t + \gamma S(x,\hat{\theta}) + \sqrt{2\gamma}Z, \text{ where } Z \sim N(0,I), \gamma \text{ small.}$$

The phase portrait of the score function and the diffusion of particles forward in time are presented in Figure 2.3 below. The random particles gradually move to be centered around the two means at $(-5,-5)$ and $(5,5)$.

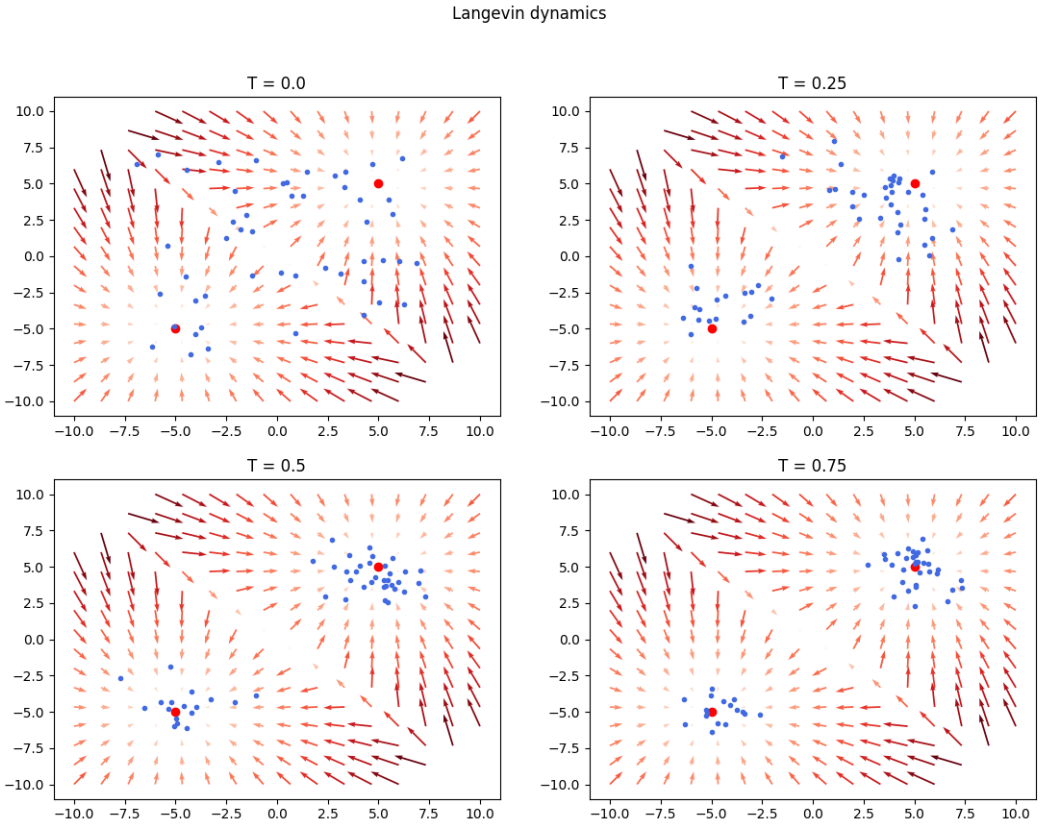

Figure 2.3: On the upper-left diagram we can see the initial random distribution of our particles (illustrated by blue dots). Under Langevin dynamics the particles move across the phase portrait and center themselves around one of the two mixture means (illustrated by red dots). Subsequent iterations after convergence move the points around the two means, with the dispersion of particles roughly following the covariance matrix of each mixture component.

Drawbacks of Unadjusted Langevin Dynamics

As the model score function is found by minimising the Fisher divergence between the model score and the true score, if our model score poorly approximates the true score in areas where the density is low then it's contribution to the $J(\theta)$ is minimal. Furthermore, as in practice we would employ the Monte Carlo estimator, this difference will be greater too in areas where the samples are less

frequent, which is of course exactly areas where the probability density is low. This is an issue because, as in the previous example, little assumptions about the true data distribution can be made in determining our chosen prior $p_0(x)$ as such, particles are initialised somewhat arbitrarily across the state space determined, of course, by the prior chosen. This means particles under Langevin dynamics will often be initialised in areas of the state space for which the true density is low. As a result, sizable inaccuracies in our model score function in these regions will impair the initial trajectory of some particles and can ultimately prevent convergence entirely, in these regions of the state space.

Additionally, even when using the true score function, Langevin dynamics can fail to adequately sample from distributions with multiple modes as they admit regions of zero score. This issues arises as it is only the random perturbations of γZ which can "push" samples across such regions. We demonstrate this flaw in the Appendix C.3 by running the Langevin procedure on an unequally weighted Gaussian mixture distribution.

2.2.5 Annealed Langevin Dynamics

In order to circumvent these issues, Yang Song proposed a method called '*annealed Langevin dynamics*' [18]. Instead of performing Langevin dynamics directly on the data samples we first we perturb the initial input data samples with noise. Conceptually, this noise will scatter the samples around the state space and populate low density regions where the model score poorly approximates the true score. This introduces an important trade-off. Excessive sample perturbation causes a loss of valuable information regarding the initial distribution of samples, which is the most reliable indicator of the parent distribution we aim to sample. Conversely, if we do not noise the system enough then we face poor score estimation in regions of low density as previously discussed.

In order to avoid introducing too much perturbation at once, increasing levels of the noise is iteratively applied to the samples. Usually the noise is Gaussian $N_j \sim \sigma_j N(0, I)$ and so the following noise scales perturb the data: $\sigma_1, \sigma_2, \ \dots \ , \sigma_M$. Under this formulation, the noised samples $x_i^{(j)}$ are distributed as follows,

$$p_{\sigma_j}(x) = \int p(y) N(x; y, \sigma_j^2 I) dy.$$

This motivates training the so-called 'Noise conditional score-based model', $S(x; \theta, j)$, which is trained such that $\forall j \in \{1, \dots, M\}$ $S(x; \theta, j) \approx \nabla_x \log p_{\sigma_j}(x)$. Similarly to minimising our previous score-matching objectives, we fit this model by finding the parameters $\hat{\theta}$ which minimise the following score matching objective,

$$J_M(\theta) = \sum_{j=0}^{M} \lambda(j) \mathbb{E}_{p_{\sigma_j}} \left[||S(x; \theta, j) - \nabla_x \log p_{\sigma_j}(x)||^2 \right]. \tag{2.15}$$

Evidently, J_M is a weighted average of the Fisher divergence across all noise levels (the weight function $\lambda(j)$ is often chosen as $\lambda(j) = \sigma_j^2$). In a similar fashion to approximating $J_{ISM}(\theta)$, each $E_{p_{\sigma_j}}[|| \cdot ||_2^2]$ can be approximated using a Monte Carlo estimator. In which, each sample $x_i^{(j)} \sim p_{\sigma_j}(x)$ can be acquired by adding Gaussian noise $\sigma_j \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ to a random sample drawn from the initial input samples $\sim p(x)$. Once $\hat{\theta}$ is determined new samples can be generated by the following procedure. For σ_M sufficiently large $S(x; \hat{\theta}, M)$ is well defined on the support of the prior. Therefore, we can run Langevin dynamics, substituting $S(x; \hat{\theta}, M)$ for the true score function, until the samples have converged to $p_{\sigma_M}(x)$. By running Langevin dynamics reversed through the noise scales (M down to 0) we see after running j instances of Langevin dynamics the samples are distributed $\sim p_{\sigma_{M-j}}(x)$. Hence the samples will be generated according to $p_0(x)$ after M iterations of Langevin dynamics. The algorithm is presented below (2):

Since it is not wholly relevant to the focus of my project, for a thorough study of how annealed Langevin dynamics can be further improved I direct the reader to this study [19].

2.2.6 De-noising Diffusion Score-Based Generative Models

Instead of a finite limit in the noise added to the system we can instead continuously perturb the data points forward in time using a diffusion process. This is a more general approach as the

Algorithm 2 Annealed Langevin Dynamics

$X_{i,t=0}^{(M)} \sim p_0(x)$ ▷ Initialise samples according to some initial distribution $p_0(x)$
for $j = M, \ldots, 1$:
  for $t = 1, \ldots, T$: ▷ Run Langevin dynamics until convergence to $p_{\sigma_j}(x)$
    $X_{i,t}^{(j)} \leftarrow X_{i,t-1}^{(j)} + \gamma S(X_{i,t-1}^{(j)}; \hat{\theta}, j) + \sqrt{2\gamma} Z$
  $X_{i,t=0}^{(j-1)} \leftarrow X_{i,T}^{(j)}$ ▷ Define initial samples for subsequent Langevin diffusion
return $\{X_{i,t=0}^{(0)}\}$.

perturbation can be run arbitrary forward in time, until the input samples are sufficiently noised across the state space.

For example, one choice of discrete noising that has been shown to work particularly well is setting the noise parameters σ_i to follow a geometric progression [19]. If at each time-step the perturbation is multiplied by a constant λ^2, it follows $\sigma_i = \lambda^i$. Considering the corresponding continuous time formulation,

$$dX_t = e^{t \log(\lambda)} dWt.$$

Which perturbs the data with mean zero and exponentially growing variance, as expected.

After discrete perturbations, samples are generated by applying a sequence of backwards Langevin diffusions. Analogously in continuous time, the reverse stochastic process of the forward time perturbation is applied for sample generation. If the limiting distribution of the perturbation process is known, samples can be generated from the unknown parent distribution by applying the time reversed perturbation process to samples drawn from the limiting distribution. Under this formulation, the limiting distribution of the time reversal process is the parent distribution our initial data was drawn from.

Importantly, in using the reverse process of a stochastic differential equation (2.11), we require knowing the score of the marginal, $\nabla_x \log p_t(x)$, $\forall t \in [0, T]$. As this quantity is unknown we aim to train a '*time dependent score-based model*' (denoted by $S(x, t; \theta)$) as a model of the time dependent score function $\nabla_x \log(p_t(x))$. In a similar manner to previous score estimation schemes, we aim to minimise the following quantity at each time in the perturbation process (which is now the continuous interval $[0, T]$):

$$J_t(\theta) = \mathbb{E}_{p_t(x)}[||\nabla_x \log p_t(x) - S(x, t; \theta)||^2] \approx \mathbb{E}_{x_0 \sim \hat{\mu}_{data}} \mathbb{E}_{p_t(x|x_0)}[||\nabla_x \log p_t(x) - S(x, t; \theta)||^2]$$

Where $\lambda(t)$ is a weight function, by [20] a good candidate is $\lambda(t) \propto 1/\mathbf{E}\left[||\nabla_{x_t} \log p(x_t|x_0)||^2\right]$. It has been shown this score matching objective is equivalent the following '*De-noising score matching objective*' [21]

$$J_t(\theta) = \mathbb{E}_{x_0 \sim \hat{\mu}_{data}} \mathbb{E}_{p_t(x|x_0)}[||\nabla_x \log(p_t(x|x_0)) - S(x, \theta, t)||^2] \tag{2.16}$$

This is a very nice formulation, as the score-matching objective is now independent of the unknown initial data distribution. Furthermore, $p_t(x|x_0)$ is defined wholly by the perturbation process used to noise the data. In many cases, the conditional probability density is analytic so $\nabla_x \log p_t(x|x_0)$ can be easily expressed for further simplification. For example, under an Ornstein-Ulhenbeck perturbation process (2.3) we have,

$$\nabla_x \log p_t(x|x_0) = -\frac{2\beta}{\sigma^2(1 - e^{-2\beta t})} \left(\alpha - x + (x_0 - \alpha)e^{-\beta t}\right)$$

$$\implies J_t(\theta) = \mathbb{E}_{x_0 \sim \hat{\mu}_{data}} \mathbb{E}_{p_t(x|x_0)} \left[\left|\left| S(x, \theta, t) + \frac{2\beta}{\sigma^2(1 - e^{-2\beta t})} \left(\alpha - x + (x_0 - \alpha)e^{-\beta t}\right) \right|\right|^2 \right].$$

As in formulating the annealed score matching objective (2.15), we aim to minimise $J_t(\theta)$ over the entire perturbation process, which is $t \in [0, T]$. Therefore, the complete score matching objective is given by the expectation of J_t over this range,

$$J_T(\theta) = \mathbb{E}_{t \sim [0,T]}[J_t(\theta)]. \tag{2.17}$$

In finding the parameters $\hat{\theta}$ which minimises this score matching objective we obtain our 'time dependent score-based model', $S(x, t; \hat{\theta})$. This is substituted for the true marginal score in the

time reversal of our perturbation process in (2.11),

$$dX_{T-t} = [-f(X_{T-t}, T-t) + \sigma(X_{T-t}, T-t)S(x, T-t; \hat{\theta})\sigma(X_{T-t}, T-t)^T]dt + \sigma(X_{T-t}, T-t)dWt \tag{2.18}$$

This approximate reverse process is applied to samples generated from the known stationary distribution of the forward process to generate new samples from the unknown parent distribution. This approach is well visualised using a diagram presented by Yang Song.

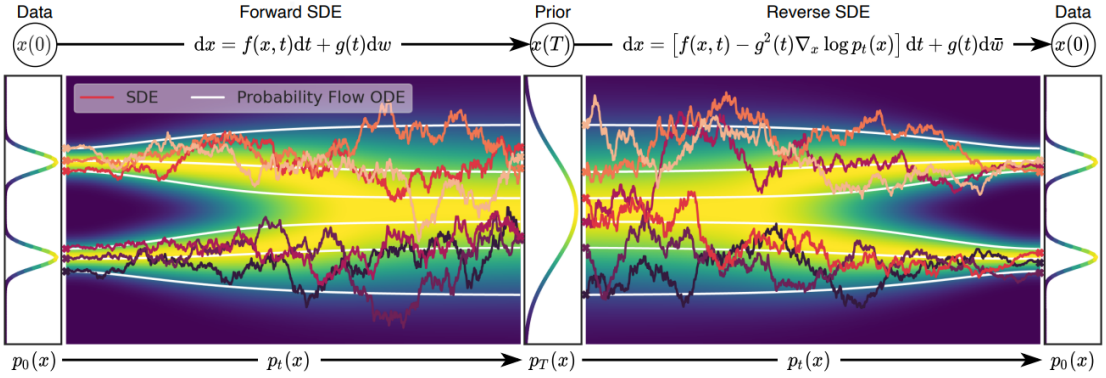

Figure 2.4: Yang Song's visualisation illustrating the forwards perturbation to a known distribution, followed by applying the reverse process to new samples from that distribution to finally generate new data points [20].

2.3 Discretisation and Implementation

2.3.1 Score Model

In practice, the score function is seldom modelled analytically, as presented in the background section, and instead a neural network referred to as the 'score-network' is used instead. Analogously to finding the optimal parameters under an analytic model density, the score-network is parameterised by its weights, θ, and trained by minimising a score-matching objective in the similar fashion. Therefore, the score matching objective serves as the loss function in training the score-network. For a thorough description of the score-network architecture, see the Appendix B.1.

2.3.2 Perturbation Process

In both theory and in experiments, we will only be considering the time homogeneous[5] Ornstein-Ulhenbeck perturbation process (2.3). This is a common choice in score-based modelling as it is mean-reverting and the stationary distribution is Gaussian. In fact, the OU process (with $\alpha = 0$) is the only mean-zero Gaussian second-order stationary Markov process with continuous paths on $\mathbb{R}$ [7, p.g. 43]. In theoretical arguments we will not restrict the choice of parameters, however in experiments we will specify $\alpha = 0$ and $\sigma = \sqrt{2\beta}$, as this maintains the standard Gaussian stationary distribution for an arbitrary $\beta \in \mathbb{R}$. Therefore, the forward and backward processes are specified as follows,

$$\begin{aligned} X_0 \sim p_{\text{data}}, \quad dX_t &= -\beta X_t dt + \sqrt{2\beta} dW_t && \text{(forward process)} \\ Y_0 \sim p_T, \quad dY_t &= [\beta Y_t + 2\beta \nabla_y \log p_{T-t}(Y_t)]\, dt + \sqrt{2\beta} dW_t && \text{(backward process)} \end{aligned}$$

Of course, we do not have access to the true score function and instead substitute the trained score-network, $S(x, t; \hat{\theta})$. In our experiments, the backward process is always discretised with the Euler-Maruyama scheme (2.9) and unless otherwise specified $\Delta t = 0.01$. Finally, as we do not have access to p_T, we instead use the known limiting distribution of the forward process, p_∞. From this we can formulate our discretised reverse process,

$$Z_0 \sim p_\infty, \quad dZ_{j\Delta t} = \left[\beta Z_{j\Delta t} + 2\beta S(Z_{j\Delta t}, j\Delta t; \hat{\theta})\right] \cdot \Delta t + \sqrt{2\beta} dW_{\Delta t}$$

[5] That is, the parameters, α, β, and σ are constant with respect to time, as well as X_t.

Chapter 3

Related Work

'Generalisation' refers to the capability of generative models to produce plausible samples outside of the training set. Investigating how altering the training process of score-based generative models affects generalisation will be the main focus of study in this project. We will start by formalising the structure of the datasets in our experiments and introduce the generalisation property with an example. Furthermore, we will consider a class of alternative generative models, known as 'push-forward' models, as a means of comparison to score-based models throughout the project. We conclude this section with an important result from the literature which bounds the error in the backward diffusion process used to generate samples in score-based modelling.

3.1 The Manifold Hypothesis'

The Manifold Hypothesis

The 'manifold hypothesis' states that data often lies on a lower-dimensional support embedded within higher dimensional space where it is observed [22]. This is a widely accepted assumption in deep learning and motivates the study of dimensionality reduction in data science. Informally, the hypothesis states data can often be wholly and accurately specified in a lower-dimensional representation. Consider, for example, the set of all 28×28 bitmap images of dogs. While this dataset lies in $\mathbb{R}^{28\times 28}$ it would be sensible to reason this dataset does not represent all images expressible in 28×28, such as images of cars, and therefore there exists a lower dimensional representation of just dog images.

The Union of Manifolds Hypothesis

The assumption that data lies on a single underlying support has been theorised to be restrictive as it implies the latent dimension is constant across the entire dataset. A recent paper hypothesises the latent support is not necessarily a single connected component but comprised of multiple separate supports - potentially of differing dimensionality [23]. This is referred to as the 'union of manifolds hypothesis'.

Firstly, I would like to formalise this definition of a disconnected subspace - which will explain why push-forward models exhibit poor generalisation on datasets lying on a union of disconnected supports.

Definition 3.1 (Disconnected data supports)**.** *Let (X, d) be a metric space and $M \subseteq X$ the support of the data. M is 'disconnected' if there exist open sets $U, V \subseteq X$ such that the following hold:*

- $U \cap V = \emptyset$
- $M \subseteq U \cup V$
- $M \cap U \neq \emptyset$ *and* $M \cap V \neq \emptyset$

Intuitively, this definition suggests that M is disconnected, if it can be separated into two pieces using open sets - with the separate pieces being $M \cap U$ and $M \cap V$.

To the best of our knowledge, there is no comprehensive discussion in the literature on the capabilities of score-based generative models to learn and generalise on datasets lying on disjoint latent supports.

3.2 Generalisation In Score-Based Generative Models

To motivate this investigation, we present an example of a dataset which lies on a lower dimensional support, with partial representation, observed in higher dimensional space. Specifically, consider the training dataset given by 8 points uniformly dispersed around the unit circle, $\mathcal{M}_{train} := \{(\cos\frac{\pi}{4}, \sin\frac{\pi}{4})|i = 0, \ \ldots, \ 7\}$, illustrated in Figure 3.1. Despite being a simplistic example, this a very realistic abstraction. In practice, the true support is always under-specified by the training set. Of course, if the true support is fully specified, we can uniformly sample from the training set to produce samples following the parent distribution. Furthermore, the circle can be parameterised by one variable, yet we observe the data points in two dimensional space, which satisfies the manifold hypothesis.

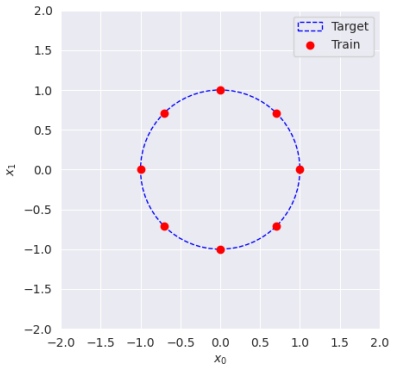

Figure 3.1: The training set and target support of this experiment.

Under perturbation using an Ornstein-Ulhenbeck process ($\beta = 1, \ \alpha = 0, \ \sigma = \sqrt{2}$), the samples converge in the forward process to $\mathcal{N}(\mathbf{0}_2, \boldsymbol{I}_2)$. If the backward process is specified using the score associated with the training set, which can be determined in this case, the support of the generated samples is exactly the support of the training set. This is visualised in Figure 3.2 below.

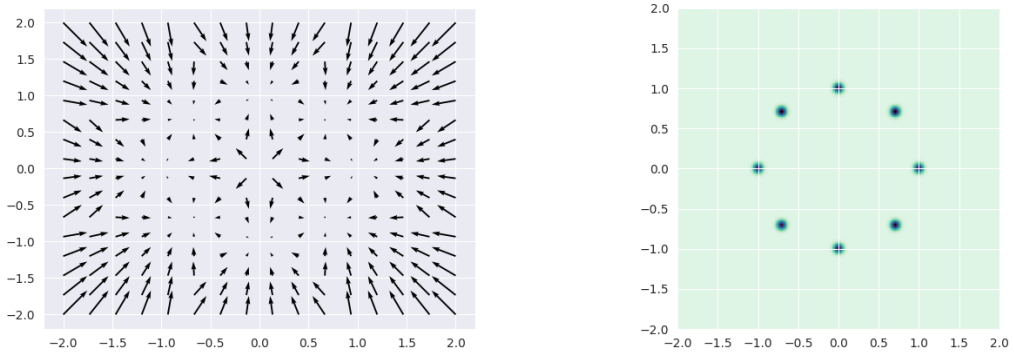

Figure 3.2: On the left-hand side we see the empirical score function evaluated at $t = 0.01$, this shows the direction of particles diffusion towards the end of the backward process. We can see the particles are directed solely towards the training set. The right-hand diagram is a heatmap of the generated samples, which we clearly see are supported exactly on the training set.

Evidently, there is no generalisation beyond the training set. This is because under the empirical score, the discrete distribution of the training set has been taken to be the true parent distribution. This would only be reasonable if the training set is very representative of the parent distribution. The score function of the training set should be replaced the score of the true distribution or, failing that, by a score model trained using an appropriate score matching objective. Figure 3.3 visualises the learned score and generated samples at $5,000$ and $11,000$ training epochs respectively.

Due to the expectation over the training set in the score matching objective (2.16), the learned score will eventually overfit and resemble the score of the training set. This is why the loss function cannot be used as a good measure of generalisation. We see this by comparing the generated samples between the training set score in Figure 3.2 and learned score at $11,000$ epochs in Figure 3.3. However, if the training is stopped $5,000$ epochs the score-based model has remarkably inferred the true support of the dataset, from just 8 training points. This is what is referred

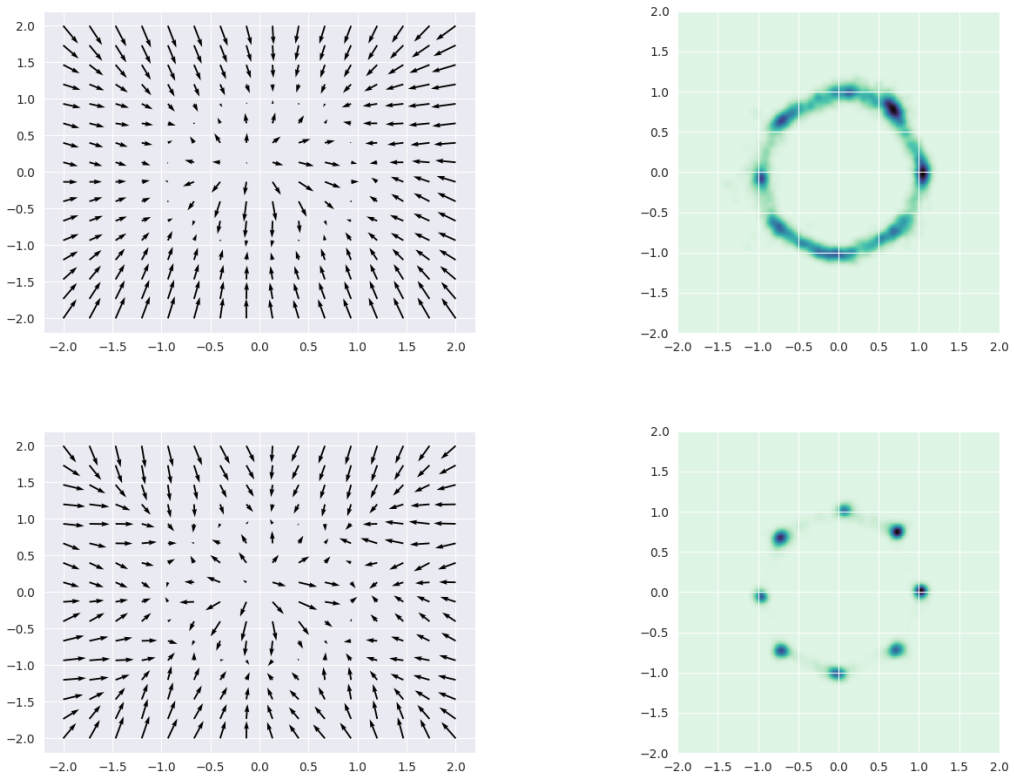

Figure 3.3: On the top row we see the learned score (at $t = 0.01$) and a heatmap of $3,000$ generated samples using the score-network in the reverse SDE at $5,000$ training epochs. The bottom row shows the same setup but with the score network at $11,000$ training epochs.

to as 'generalisation' in score-based models. Early stopping is a classic regularisation technique in machine learning and in this example we have observed its effect on the quality of generated samples.

This example clearly exhibits the generalisation property of score-based generative models and was largely based on the documentation of the `diffusionjax` library [24]. This motivates an investigation of how altering the training process of score-based models affects the quality of generated samples. Specifically, we focus on the parameters of the Ornstein-Ulhenbeck process (2.3) and the architecture of the score network.

3.3 Generalisation Metrics

In order to quantify the affects of varying the training process on the quality of generated samples, we require a set of metrics to measure the quality of generated samples on a target support. Some of these metrics have strong theoretical foundations which we use to derive results on generalisation capabilities. In reality however, data (such as images) is seldom distributed following a tractable probability density. Therefore, in practice such metrics are less applicable. As a result, I have considered more general metrics such as sliced-Wasserstein and also constructed novel metrics for specific datasets of study. The dataset specific metrics are discussed in the following chapter.

3.3.1 Total Variation

The primary metric I will be using, in derivations, to measure distances between distributions is the total variation, which is defined as follows:

Definition 3.2 (Total variation)**.** *Let p and q be probability measures defined on the measurable space $(X, \mathcal{F})$. The total variation is the measurable set for which the difference in measure is*

greatest.

$$\mathrm{TV}(p,q) = \sup_{A\in\mathcal{F}} |p(A) - q(A)| = \frac{1}{2}\int_X |p(x) - q(x)|dx \in (0,1).$$

Using results from measure theory, we can derive at the latter formulation [25, Proposition 4.2, p.g. 48]. Trivially, $\mathrm{TV}(p,p) = 0$ and $\mathrm{TV}(p,q) = 1$ for p, q defined on disjoint supports. In fact, we can make the strong assertion that for p and q defined on $\mathbb{R}^d$, $\mathrm{TV}(p,q) = 1$ if the intersection of the two supports has Lebesgue measure 0.

3.3.2 Kullback–Leibler Divergence

Definition 3.3 (Kullback–Leibler divergence)**.** *Let p and q be probability measures defined on the measurable space $(X,\mathcal{F})$. The Kullback-Leibler divergence (or 'KL-divergence') is given by,*

$$KL(p||q) = \int_X p(x)\log\frac{p(x)}{q(x)}dx.$$

KL-divergence admits the following properties: $KL(p||q) \geq 0$ with $KL(p||q) = 0 \iff p = q$. Additionally, it is not symmetric ($KL(p||q) \neq KL(q||p)$) and therefore is not a metric in the formal sense. KL-divergence is a special case of cross-entropy and is commonly used as a loss function in single-label machine learning classification tasks. It is of particular relevance to this project as it forms an upper-bound on the total variation, given by the following relation:

Lemma 3.1 (Pinsker's Inequality)**.** *[26, Proof 3.18 p.g. 44]*

$$\mathrm{TV}(p,q) \leq \frac{1}{\sqrt{2}}\sqrt{KL(p||q)}.$$

3.3.3 Wasserstein Distance

Wasserstein distance is a metric between two probability measures $p, q \in \mathbb{R}^n$. It is motivated by optimal transport theory as it aims to move probability mass from one distribution to another as efficiently as possible. It is formalised as follows:

Definition 3.4 (Wasserstein distance)**.** *The Wasserstein distance between p and q is given by,*

$$W_d(u,v) = \inf_{\pi\in\Gamma(p,q)}\left[\int_{\mathbb{R}\times\mathbb{R}} ||x-y||_2^d d\pi(x,y)\right]^{1/d}, \text{ so } W_1(p,q) = \inf_{\pi\in\Gamma(p,q)}\left[\int_{\mathbb{R}\times\mathbb{R}} |x-y|d\pi(x,y)\right].$$

Where $\Gamma(p,q)$ is the set of all probability distributions $\pi \in \mathbb{R}^{2n}$ with,

$$\int_{\mathbb{R}^n}\pi(x,y)dy = p(x) \text{ and } \int_{\mathbb{R}^n}\pi(x,y)dx = q(y)$$

So p and q are the marginals of $\pi(x,y)$ and $||\cdot||_2^d$ is the cost function of transporting the probability mass (or density in the continuous case). This definition is quite abstract, so we present a plot of the Wasserstein distance between a standard normal distribution $N(0,1)$ and a normal distribution with a moving mean $\mu \in [-2.5, 2.5]$ in the Appendix C.2.

The d-Wasserstein distance has an analytic form for one dimensional distributions [27, Chapter 2]. Calculating the inverse cumulative density function on a discrete distribution requires sorting the input, therefore in practice this formulation has a time complexity of $\mathcal{O}(N\log N)$ (where N is the number of samples).

$$W_d(p,q) = \left(\int_0^1 ||F_p^{-1}(z) - F_q^{-1}(z)||_2^d dz\right)^{\frac{1}{d}}. \tag{3.1}$$

3.3.4 Sliced Wasserstein Distance

Computing the Wasserstein distance is exponential in the dimensionality of the data [28]. In order to circumvent this computational cost, we use the so-called sliced Wasserstein distance. The samples in higher dimensions are be projected to the real line, after which the Wasserstein distance is calculated between projected samples. This process is repeated for many random projections and an average is taken. This metric is motivated by the relatively inexpensive computation of the analytic form of the Wasserstein distance between one-dimensional distributions [29].

3.4 Push-Forward Models

Push-forward models are a class of generative models that generate new data by transforming samples drawn from an initial distribution through a learned deterministic mapping. Generative-adversarial networks and variational auto-encoders are both prominent examples of push-forward models. In both these models, the push-forward function is a neural network. Notably, as neural networks are compositions between matrix operations and continuous activation functions, this mapping is continuous, usually Lipschitz continuous [30].

3.4.1 Generative Adversarial Networks

Generative-adversarial networks (GANs) are a form of generative model generate new samples through the use of two neural networks - a generator and discriminator. The generator aims to produce samples indistinguishable from the training set to fool the discriminator. Simultaneously, the discriminator aims to correctly label generated and input samples as real or synthetic. The generator takes in latent variables and pass them through the neural network to generate plausible synthetic samples in observable space. The latent variables are almost always from a latent space assumed to have connected support, for example $N(\mathbf{0}, \boldsymbol{I}_n)$ supported on $\mathbb{R}^n$ or $U(0,1)^n$ supported on $[0,1)^n$. The generator and discriminator are trained against each other at the same time. As this adversarial training process continues, the generator learns to produce increasingly realistic samples that plausibly come from the same parent distribution as the real data.

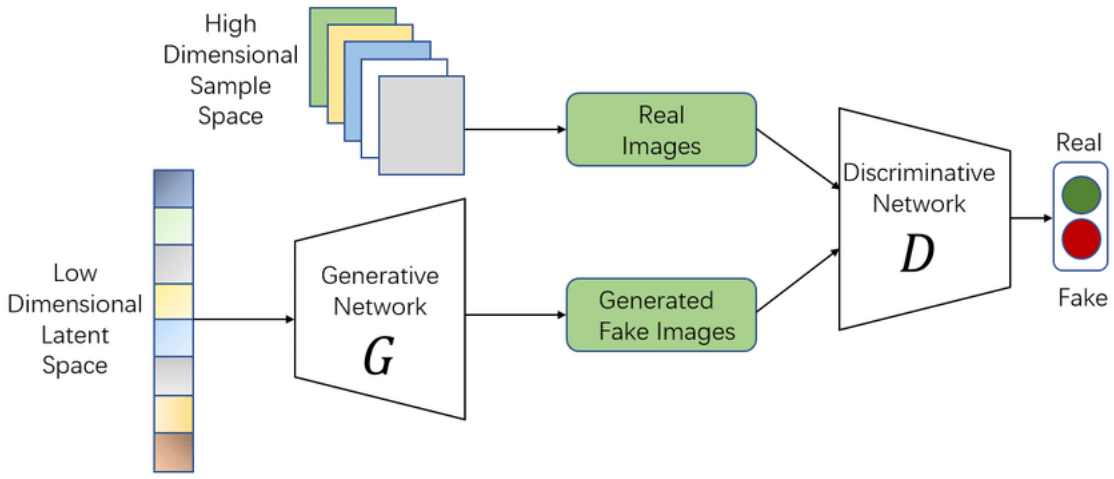

Figure 3.4: This diagram, from Kaggle [31], depicts the standard architecture of generative-adversarial models.

3.4.2 Variational Auto-Encoders

Variational auto-encoders (VAEs) also consist of two neural networks - an encoder and a decoder. The encoder learns a mapping from the samples (in observable space) to a parameterised latent space. The dimension of the latent space is a hyper-parameter of the model and usually motivated by some prior understanding of the data. The encoder maps observed samples onto a mean and standard deviation latent variable. Samples are drawn from this distribution and fed into the

decoder which learns the inverse mapping, from latent space back to observable space. Training is performed simultaneously although not adversarially like with GANs. Throughout the training process, the decoder produces higher quality synthetic samples as it improves its mapping from $N(0,1)$ to observable space. Finally, we can produce new data by transforming independent samples (from $N(0,1)$) with the decoder network.

3.4.3 Drawbacks Of Push-Forward Generative Models

Push-forward models possess a number of drawbacks, some of which are presented below. In this work we aim to reason to what extent score-based models suffer the same limitations.

Push-Forward Sample Generation From Gaussian Mixtures

In generating samples from a one dimensional Gaussian mixture distribution, in order to maintain a low total variation between the transformed samples and data distribution, the size of the push-forward network must scale accordingly. That is, there is an intrinsic relation between the size of the generative network and the distance between the component means. This is a notable flaw as there is no inherent increase in complexity of the distribution with less aligned mixture components. This relation is formalised in the following theorem,

Proposition 3.1 (Corollary 6 [32]). *Let $q \sim \mathcal{N}(0,1)$ and let the target measure $p_0 = \frac{1}{2}N(m_1,\sigma^2) + \frac{1}{2}N(m_2,\sigma^2)$ and g a Lipschitz continuous neural network. Then*

$$\mathrm{TV}(g_{\#}q, p_0) \geq \int_0^{|m_1-m_2|/4\sigma \mathrm{Lip}(g)} \mu(t)dt - \frac{1}{2}\int_{|m_2-m_1|(2\sigma-1)/4\sigma^2}^{|m_2-m_1|(2\sigma+1)/4\sigma^2} \mu(t)dt$$

where $\mu(x) = (2\pi)^{-1/2}\exp(-x^2/2)$ is the standard Gaussian density and $\mathrm{Lip}(g)$ denotes the Lipschitz constant of g.

Failure Of GANs and VAEs To Learn Datasets Which Lie On Disconnected Supports

It has been demonstrated that push-forward models are unable to accurately learn disconnected latent supports [33], that is datasets conforming to the union of manifolds hypothesis. It can be readily shown that 'connected-ness' is preserved under continuous mappings [34, Theorem 43, p.g. 87]. Therefore, a push-forward model will always produce samples lying on a connected support, this is formalised in the following theorem,

Theorem 3.1. *[35, Proposition 1] Let $\mathcal{Z}$ and $\mathcal{X}$ be topological spaces and $G : \mathcal{Z} \to \mathcal{X}$ be continuous. Considering $\mathcal{Z}$ and $\mathcal{X}$ as measurable spaces with their respective Borel σ-algebras, let $\mathbb{P}(\mathcal{Z})$ be a probability measure on $\mathcal{Z}$ such that $supp(\mathbb{P}_{\mathcal{Z}})$ is connected and $\mathbb{P}_{\mathcal{Z}}(supp(\mathbb{P}_{\mathcal{Z}})) = 1$. Then $supp(G(\mathbb{P}_{\mathcal{Z}}))$ is connected.*

As G cannot map connected domains to disconnected domains the support of the probability measure must be connected in $\mathcal{X}$. This implies there exists a region $A \subset X$ connecting the two disconnected supports, such that $\mathbb{P}(A) \neq 0$. Due to this positive density, generated samples will lie in the region between the disconnected supports.

It is worth noting, there are modifications to the training process - such as training a collection of generators for each disconnected region of the support - which help alleviate this shortcoming [35]. However, while this may be applicable on simple datasets, identifying how many latent disconnected supports comprise a dataset is an almost intangible problem and impractical on real world datasets.

3.5 Backward Convergence Of Score-Based Generative Models

Developing mathematical justifications for the remarkable empirical success of score-based models is an active area of research. One result of particular relevance to this project provides an upper-bound on the total variation between the data distribution and the backward process. In order to formulate this result, we first introduce some additional notation.

1. Let p_t denote the forward process under an Ornstein-Ulhenbeck perturbation given by $dX_t = -X_t dt + \sqrt{2}dW_t$, with $p_0 = p_{data}$ and p_∞ the unique stationary distribution, $N(\mathbf{0}_n, \boldsymbol{I}_n)$.
2. Let q_t denote the reverse process $dX_t = (X_t + 2\nabla_x \log p_{T-t}(x))dt + \sqrt{2}dW_t$. This process is initialised at the stationary distribution of the forward process and therefore $q_0 = p_\infty$. The law of this process is denoted by Q_t.
3. Let $Q_t^{p_T}$ denote the law of the discretised score-based generative modelling algorithm which follows the following discretisation of the reverse process, $dX_t = (X_t + 2S(X_{j\Delta t}, T - j\Delta t; \theta)dt + \sqrt{2}dW_t,\ t \in [j\Delta t, (j+1)\Delta t]$, initialised at the end of the forward process, $X_0 \sim p_T$.

Under this formulation we can establish an upper-bound between the total variation of the backward process and data distribution given by,

$$\mathrm{TV}(q_T, p_0) \leq \mathrm{TV}(q_T, Q_t^{p_T}) + \mathrm{TV}(Q_t^{p_T}, Q_T) \leq \mathrm{TV}(p_T, p_\infty) + \mathrm{TV}(Q_t^{p_T}, Q_T), \tag{3.2}$$

where the first inequality follows from the triangle inequality applied to the total variation and second inequality follows from the data processing inequality [36, Section 4]. As samples in the reverse process are initialised at p_∞, the first term corresponds to the error induced by the difference in the forward process p_T and p_∞. The following theorem establishes a relation between the second term and errors associated with the discretisation and score function approximation in the reverse process.

Theorem 3.2 (Discretisation error in backward convergence, Theorem 9, [36])**.** *Under the following assumptions:*

1. *The score function of the forward process, $\nabla_x \log p_t(x)$, is Lipschitz continuous (with Lipschitz constant L).*
2. *The second moment of the limiting distribution of the forward process is finite (denoted as m_2).*
3. *The initial data distribution has finite KL-divergence with respect to the limiting distribution of the forward process.*
4. *Let the step size of our discretisation of the reverse process be $\Delta t := T/N$ which satisfies $\Delta t \lesssim 1/L$[1], where $L \geq 1$.*

$$\mathrm{TV}(Q_T, Q_t^{p_T})^2 \leq KL(Q_T || Q_t^{p_T}) \lesssim (L^2 dh + L^2 m_2^2 h^2)T + \epsilon_{score}^2 T.$$

From which we can conclude,

$$\mathrm{TV}(Q_T, Q_t^{p_T}) \lesssim (L\sqrt{dh} + Lm_2 h + \epsilon_{score})\sqrt{T}.$$

Combining the above inequality (3.2) with Theorem 3.2, we arrive at an upper bound on the total variation between the true data distribution and the reverse process.

Theorem 3.3 (Total variation bound on the backward process, Theorem 2 [36])**.** *Under the above assumptions, the total variation between the backwards process and initial data distribution is given by,*

$$\mathrm{TV}(q_T, p_0) \lesssim \underbrace{\mathrm{TV}(p_T, p_\infty)}_{\text{Convergence in forward process}} + \underbrace{(L\sqrt{dh} + Lm_2 h)\sqrt{T}}_{\text{Discretisation error}} + \underbrace{\epsilon_{score}\sqrt{T}.}_{\text{Score approximation error}}$$

Notably, ϵ_{score} denotes the error in our score network approximating the score of the true parent distribution. If the model score has overfit to the training set then this term, and subsequently the upper bound, will increase. Therefore, this theorem accounts for the generalisation property of SGMs in the bound too.

This result was derived on an Ornstein-Ulhenbeck perturbation process with parameters $\beta = 1, \alpha = 0, \sigma = \sqrt{2}$. This is a standard process to use in diffusion models as the stationary distribution is the standard Gaussian in $\mathbb{R}^n$. Part of this work aims to generalise the above inequality with respect to arbitrary parameters in the forward perturbation process.

[1]The symbol $\lesssim$ denotes being less than or equal to the order of the right-hand side.

Chapter 4

Formalising Generalisation Metrics

As discussed in the preliminary section, we use a variety of generalisation metrics to measure the generalisation quality of generated samples. In this section we formalise the generalisation property and justify our choice of metrics from the literature. Additionally, we introduce the datasets used in our experiments and conclude this section with a discussion on the suitability of different metrics in theory and in practice.

4.1 Desired Generalisation Metric Properties

In order for the generalisation metrics to remain consistent, in this project, good generalisation is indicated by a low score under a generalisation metric while poor generalisation will incur a higher score. What generalisation precisely means in this project is formalised below.

Let A be the set of generated samples by the SGM and $\mathcal{M}$ the target support in observable space. In order for a generalisation metric, $g_{\mathcal{M}} : X \to \mathbb{R}_{\geq 0}$ to be well constructed it must satisfy the following ordering,

1. $g_{\mathcal{M}}(A)$ is high for $A \subseteq X/\mathcal{M}$, that is generalisation measured as 'poor' if the produced samples are off the manifold and elsewhere in the state space. In the context of this project this indicates the SGM is not able to generate appropriate samples which is likely due to the score network poorly approximating the true score.
2. $g_{\mathcal{M}}(A)$ is lower for $A \subset \mathcal{M}$, generalisation gives a better (lower) score for samples which lie on a subset of, or relatively close to, the true support - even if the support of the generated samples lies in the training set (for example $A = \mathcal{M}_{\text{train}}$).
3. $g_{\mathcal{M}}(A)$ is lowest for $A = \mathcal{M}$, generalisation is best (lowest) for samples appropriately dispersed around the true support in observed space.

The distinction between 2. and 3. identifies when the model score has over-fit to the score function associated with the training set. For optimal generalisation under this construction, if there exists a target density then the samples must lie wholly on the support of the target distribution and be relatively dense around the support, that is they fit the probability density too.

We now justify why these chosen measures from the background section abide by this formalisation and therefore adequately measure generalisation in the context of this project.

Total Variation

From the definition of total variation (3.2), we can justify the above ordering holds. Considering the target density q defined on the full support $\mathcal{M}$ and generated samples which follow the density p. If p has positive density on $A \subseteq X/\mathcal{M}$ then as $q(A) = 0$ we get,

$$\begin{aligned}
\text{TV}(p, q) &= \frac{1}{2}\int_X |p(x) - q(x)|dx \\
&= \frac{1}{2}\int_A |p(x) - q(x)|dx + \frac{1}{2}\int_{\mathcal{M}} |p(x) - q(x)|dx \\
&= \frac{1}{2}\int_A |p(x)|dx + \text{TV}(p_{\mathcal{M}}, q_{\mathcal{M}}) = \frac{1}{2}p(A) + \text{TV}(p_{\mathcal{M}}, q_{\mathcal{M}}) \geq \frac{1}{2}p(A).
\end{aligned}$$

Therefore, the measure of the density which lies off the target support determines a reasonable lower bound on the measure of generalisation. In the case we have the generated samples lying exclusively on a subset of the support, such as in the instance of learning the training set, we find,

$$\begin{aligned}\mathrm{TV}(p,q) &= \frac{1}{2}\int_X |p(x)-q(x)|dx \\ &= \frac{1}{2}\int_{\mathcal{M}_{\text{train}}} |p(x)-q(x)|dx + \frac{1}{2}\int_{\mathcal{M}/\mathcal{M}_{\text{train}}} |p(x)-q(x)|dx \\ &= \mathrm{TV}(p_{\mathcal{M}_{\text{train}}}, q_{\mathcal{M}_{\text{train}}}) + \frac{1}{2}q(\mathcal{M}/\mathcal{M}_{\text{train}}) \geq \frac{1}{2}q(\mathcal{M}/\mathcal{M}_{\text{train}}).\end{aligned}$$

Again we have derived a lower bound for the total variation, although this time the bound is on the measure of the true density on the region where the generated samples are not supported. Finally in the case that p and q share support we see that the total variation is just the average difference in density across the the support, which we would expect to be lower as they are each positive quantities for all $x \in \mathcal{M}$.

$$\mathrm{TV}(p,q) = \frac{1}{2}\int_X |p(x)-q(x)|dx = \frac{1}{2}\int_{\mathcal{M}} |p(x)-q(x)|dx.$$

One Dimensional Wasserstein Distance

Considering the analytic form of the one dimensional Wasserstein distance (3.1), we can draw similar conclusions. As p and q are one dimensional distributions we define $p_{min} := \inf\{x \in \mathbb{R} \mid p(x) > 0\}$ and $p_{max} := \sup\{x \in \mathbb{R} \mid p(x) > 0\}$. Substituting $x = F_p(z)$ into the analytic form, we derive an alternative representation,

$$\begin{aligned}W_d(p,q) &= \left(\int_0^1 ||F_q^{-1}(z) - F_p^{-1}(z)||_2^d dz\right)^{\frac{1}{d}} = \left(\int_{p_{min}}^{p_{max}} p(x)||F_q^{-1}(F_p(x)) - F_p^{-1}(F_p(x))||_2^d dx\right)^{\frac{1}{d}} \\ &= \left(\int_{p_{min}}^{p_{max}} p(x)||F_q^{-1}(F_p(x)) - x||_2^d dz\right)^{\frac{1}{d}} = \mathbb{E}_p\left[||F_q^{-1}(F_p(x)) - x||_2^d\right]^{\frac{1}{d}}. \\ \implies W_1(p,q) &= \mathbb{E}_p\left[||F_q^{-1}(F_p(x)) - x||_2\right].\end{aligned}$$

To understand this formulation, we need to interpret the quantity $F_q^{-1}(F_p(x))$. For a given $x \in \text{Support}(P)$, the cumulative density $F_p(x)$ represents how far along the support x lies. $F_q^{-1}(F_p(x))$ therefore identifies the corresponding point in the support of q which lies as far along the cumulative density of q. The one dimensional Wasserstein distance calculates the expectation of the Euclidean distance between each x in $\text{Support}(p)$ and the corresponding point in q.

In the case p has positive density on $A \subseteq X/\mathcal{M}$ we expect $W_1(p,q)$ to be quite large and determined by amount of density attributed to A and the distance between A and $\mathcal{M}$. For p defined on $A \subset \mathcal{M}$, we expect the expectation to be lower as the supports will be close to each other, however there is still a mismatch along some of the support which will be expressed in the value of the expectation. Finally, considering when the supports are equal, it is just the difference in density along the support which will contribute the any differences in Wasserstein distance.

Sliced Wasserstein Distance

Due to the relatively low computational burden we will use the sliced Wasserstein distance, instead of higher dimensional Wasserstein distances, in our experiments. Considering again the 3 cases in our formulation, for p with positive density on $A \subset X/\mathcal{M}$ we would expect the vast majority of random projections from A to map to different parts of the real line to those projected from $\mathcal{M}$. Therefore, following this case in the Wasserstein justification, the sliced Wasserstein distance will be large too. For p defined on $A \subset \mathcal{M}$ we would expect the image under projections to lie in a subset of the image of projections of $\mathcal{M}$ therefore upon evaluating the Wasserstein distance between the projected samples we can apply the same argument as reasoned for the one dimensional Wasserstein distance above. For $A = \mathcal{M}$ we would expect very small discrepancies as the image is the image of the projected support. However the density across that image may differ, causing a non-zero generalisation score. These justifications are supported in Table 4.2.

4.2 Datasets

4.2.1 Gaussian Mixture Distributions

Gaussian mixture distributions are a prominent toy dataset in generative modelling as they possess a simple analytic density and have full support in $\mathbb{R}^n$. Furthermore, Gaussian mixtures can be used to model an extensive set of more complex multi-modal distributions. Gaussian mixtures serve as the base case on which we will develop theoretical results and run experiment on, before generalising to arbitrary datasets.

4.2.2 Circle Dataset

A dataset we will consider in many experiments are samples dispersed around the unit circle, illustrated in Figure 4.1. As previously discussed, this is a simple example of a dataset which lies on a lower dimensional support, with partial representation, observed in higher dimensional space. In many experiments, the training dataset is indicated by the red dots. We can construct an example test dataset by rotating these points $\frac{\pi}{4}$ radians around the circumference. The true data support which we aim to generalise samples along is visualised by the dotted blue line.

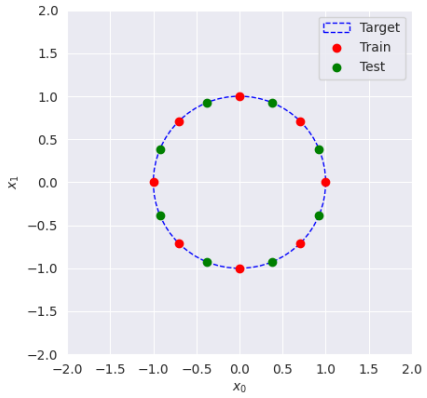

Figure 4.1: The circle dataset.

Generalisation Metrics On The Circle Dataset

Due to the simple analytic form of the circle dataset we can construct generalisation metrics for this specific dataset. In general, constructing metrics which appropriately discriminate between samples off the support and samples on the manifold, while also discriminating between well-generalised samples and the training set, is particularly difficult. In many generative models, this relation is learned implicitly - such with a discriminator network - as opposed to being expressed explicitly.

In each of the below metrics, let $\mathcal{G}$ denote the generated samples and W_1 denote the one dimensional Wasserstein distance. We first map[1] the two dimensional samples to polar coordinates, $\mathcal{G} \to \{\mathcal{G}_r, \mathcal{G}_\theta\}$.

Definition 4.1 (Circle Generalisation Metric C_1)**.** *Given a set of samples, $\mathcal{S}$ from our data support, we define the C_1 metric as follows:*

$$C_1(\mathcal{G}, \mathcal{S}) = W_1(\mathcal{G}_r, \mathcal{S}_S) + W_1(\mathcal{G}_\theta, \mathcal{S}_\theta)$$

For example, if we wish to measure to what extent samples mimic the true support we can use the C_1 metric with $\mathcal{S}$ equal to the training set. If we wish to measure generalisation we can use a witheld test set.

As the target support is an analytically tractable, the true distribution of r and θ along the support can be expressed as distributions, with $r \sim \delta(1)$ and $\theta \sim U[0, 2\pi)$. This motivates our second generalisation metric on the circle dataset.

Definition 4.2 (Circle Generalisation Metric C_2)**.**

$$C_2(\mathcal{G}) = W_1(\mathcal{G}_r, r) + W_1(\mathcal{G}_\theta, \theta).$$

These metrics each abide by our formulation, this can be reasoned by considering the separate components. The Wasserstein distance on the radii measures the average distance the samples lie from the target support while the distance on the angle quantifies the dispersion of generated samples around the support.

[1] Explicitly, this mapping is given by $(x_i, y_i) \to \left(\sqrt{x_i^2 + y_i^2}, \arctan(y_i/x_i)\right)$.

4.2.3 Swiss Roll

The Swiss roll is a common synthetic dataset to experiment on in generative modelling research. It is visualised in the top-left image of Figure 4.2. We will now demonstrate experimentally that the sliced Wasserstein distance is an appropriate generalisation metric for this dataset.

Swiss Roll Generalisation Under The Sliced Wasserstein Metric

In this demonstration, we consider the following four datasets illustrated in Figure 4.2. For this project, I wrote a fast vectorised implementation of the sliced Wasserstein distance, which can be found in the appendix (D.1).

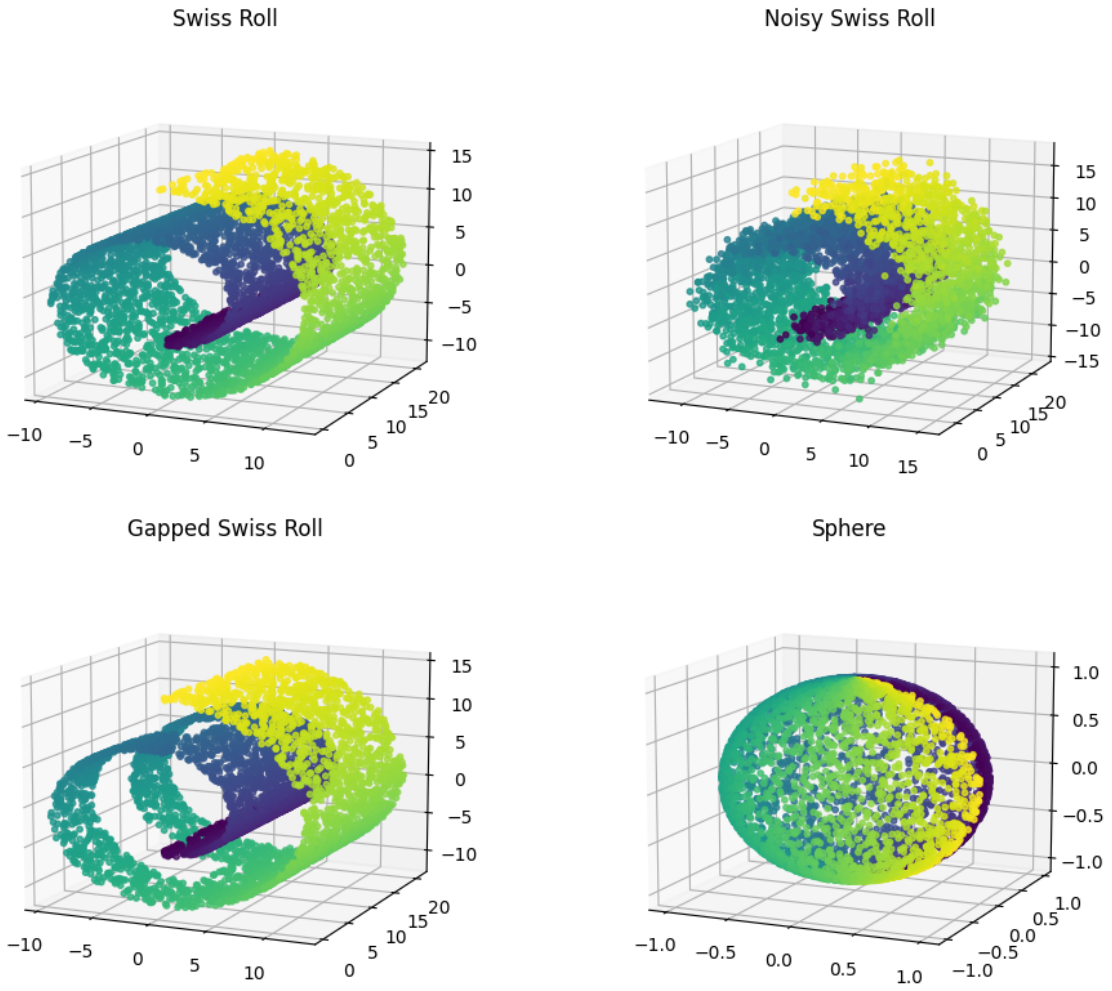

Figure 4.2: The four datasets are: the ground truth Swiss roll dataset (top left),the Swiss roll dataset with Gaussian noise $\sim N(\mathbf{0}_3, 1.5 \cdot \boldsymbol{I}_3)$ (top right), the Swiss roll dataset with a hole in the support (this is therefore a subset of the true support) (bottom left), and a sphere - a very different dataset (bottom right).

The resulting measures of generalisation, under the sliced Wasserstein metric, against the swiss-roll dataset are given below:

Support measured against	Sliced Wasserstein distance
Another Swiss roll generated with a different seed	0.1696
Noised Swiss roll	0.2116
Gapped Swiss roll	0.5388
Sphere	7.1068

These results align with the desired properties as generalisation quantified best for another set of samples which lie exactly on the same support. Generalisation is marginally worse for samples

which lie on a subset (gapped) or very close to the true support (noised). Finally, generalisation is measured dramatically worse for samples lying on a very differently shaped dataset (sphere).

4.2.4 Union Of Circles Dataset

In order to demonstrate the capabilities of score-based models in generating samples supported on disjoint supports, with lower-dimensional representations, we consider the 'union of circles' dataset, Figure 4.3. This dataset is a suitable abstraction of real datasets satisfying the union of manifolds hypothesis (3.1), as the support is disconnected (Definition 3.1) and each component has a one-dimensional representation.

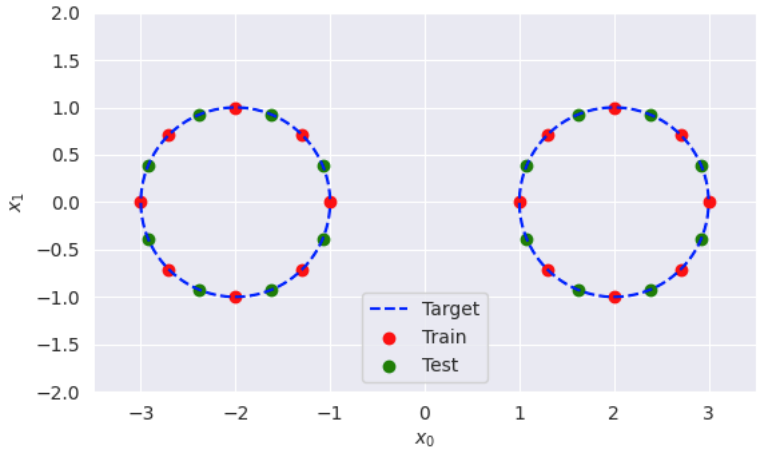

Figure 4.3: The training dataset is indicated by the red dots, the test dataset is rotated $\frac{\pi}{4}$ radians around the circumference of each disconnected support component. The true data support our model will aim to generalise is visualised by the dotted blue line.

4.3 Generalisation Metrics In Theory And Practice

Due to the relatively simple analytic form of total variation, it is well suited for theoretical reasoning about the convergence of the forward and backward diffusion processes. However it is a poor measure of generalisation between samples drawn from each distribution, which makes it poorly suited to experiments. To see this, we consider the Monte-Carlo estimate of the probability density, $p^N(x) := \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}(x) \approx p(x)$. Due to slight numerical differences, it is almost impossible that two estimates, $p^{(N)}(x)$ and $q^{(M)}(x)$ share any support. Therefore, $\mathrm{TV}(p^N, q^M) = 1$. As a result, metrics such as Wasserstein distance, which are calculated based on the distance between samples or cumulative density functions are more appropriate. Therefore, theoretical arguments will be based on the total variation and these results will be verified experimentally with other generalisation metrics, such as the Wasserstein distance. This is a suitable substitution as I have shown all of the above metrics follow the ordering specified at the beginning of this section.

Chapter 5

Results

In this chapter, we present the derivations and supporting experiments for the results in the project. Firstly, we derive the rate of convergence of the forward process, under some minimal assumptions, for arbitrary data distributions. We then discuss the impact of perturbation parameters on the discretisation error and demonstrate a trade off in selecting β. We conclude our bound on the error in the backward process with an investigation into regularisation techniques in training the score-network. Finally we combine these results to provide theoretical justifications into the capabilities of score-based models to learn datasets with disjoint latent supports.

5.1 Convergence in the Forward Process

The first part of this project aims to investigate what effect the parameters have on the convergence of the forward process. As the total variation between the forward process and stationary distribution contributes to the upper-bound on the total variation of the backward process in Theorem 3.3, we aim to minimise this quantity with respect to the parameters α, β and σ. Evidently, from the solution to the Ornstein-Ulhenbeck process, these parameters determine the unique stationary distribution. However, they also affect the rate of convergence, which we aim to quantify in this section.

5.1.1 Forward Convergence On Gaussian Mixture Models

We initially consider the forward convergence on one dimensional Gaussian mixture distributions. Due to this focus on mixture distributions, we first establish the following lemma relating to the total variation,

Lemma 5.1. *Let p, q be probability measures on $(X, \mathcal{F})$, where p is a mixture of measures, $\lambda \in (0, \frac{1}{2}), p(x) = \lambda p^{(1)}(x) + (1-\lambda)p^{(2)}(x)$. Then,* $\mathrm{TV}(p, q) \leq \lambda \mathrm{TV}(p^{(1)}, q) + (1-\lambda)\mathrm{TV}(p^{(2)}, q)$.

Proof. This follows from the triangle inequality and is proved in the Appendix A.1. □

This result naturally generalises to arbitrarily many mixture components, however we restrict our analysis to just two components. Arguments pertaining to greater numbers of mixture components follow analogously.

One Dimensional Gaussian Mixture Models

In order to derive the total variation in the forward process, we first recall the total variation between one dimensional Gaussian distributions,

Lemma 5.2 (Theorem 1.3, [37])**.** *The following establishes an upper bound on the total variation between two one-dimensional Gaussian distributions.*

$$\mathrm{TV}(N(\mu_1, \sigma_1^2), N(\mu_2, \sigma_2^2)) \leq \frac{|\sigma_1^2 - \sigma_2^2|}{2\sigma_1^2} + \frac{|\mu_1 - \mu_2|}{2\sigma_1}.$$

Corollary 5.1. *Notably, the above bound is not symmetric, however this can be easily corrected,*

$$\mathrm{TV}(N(\mu_1,\sigma_1^2),N(\mu_2,\sigma_2^2)) \leq \frac{|\sigma_1^2-\sigma_2^2|}{2\max(\sigma_1^2,\sigma_2^2)}+\frac{|\mu_1-\mu_2|}{2\max(\sigma_1,\sigma_2)}.$$

It can be shown that Gaussian distributions remain Gaussian under the Ornstein-Ulhenbeck process, this is proved in the Appendix A.1. Therefore, we have each component of the mixture distribution, $p_t^{(i)}$ is Gaussian. From this we can apply the above results to bound the total variation in the forward process,

Theorem 5.1. *Let our initial data X_0 follow a one-dimensional Gaussian distribution. Under an Ornstein-Ulhenbeck perturbation process (2.3) with a limiting invariant distribution q. Applying Corollary 5.1 we can derive the following bound of the total variation of our forward process.*

$$\mathrm{TV}(p_t,q) \leq e^{-2\beta t}\frac{|(\mathbb{V}[X_0]-\frac{\sigma^2}{2\beta})|}{2\max\left\{\frac{\sigma^2}{2\beta}+(\mathbb{V}[X_0]-\frac{\sigma^2}{2\beta})e^{-2\beta t},\frac{\sigma^2}{2\beta}\right\}}+e^{-\beta t}\frac{|\mathbb{E}[X_0]-\alpha|}{2\max\left\{\sqrt{\frac{\sigma^2}{2\beta}+(\mathbb{V}[X_0]-\frac{\sigma^2}{2\beta})e^{-2\beta t}},\frac{\sigma}{\sqrt{2\beta}}\right\}}$$

Which tends to 0 as $t\to\infty$.

Proof. The proof of this theorem follows from deriving the marginal distribution p_t explicitly (see Appendix A.1) and bounding the total variation between Gaussian distributions (Corollary 5.1). □

Corollary 5.2. *From Corollary 5.1 it immediately follows for $\mathbb{V}[X_0]<\frac{\sigma^2}{2\beta}$ we have:*

$$\mathrm{TV}(p_t,q) \leq \beta e^{-2\beta t}\frac{|(\mathbb{V}[X_0]-\frac{\sigma^2}{2\beta})|}{\sigma^2}+\sqrt{\beta}e^{-\beta t}\frac{|\mathbb{E}[X_0]-\alpha|}{\sqrt{2}}.$$

Less rigorously, considering the denominator in the Theorem 5.1, this bound approximately holds for βt sufficiently large.

With this result, the rate of convergence of an arbitrary one dimensional Gaussian mixture parent distribution can be determined.

Corollary 5.3. *Let the initial distribution be a one dimensional Gaussian mixture given by $p_0(x)=\lambda p_1(x)+(1-\lambda)p_2(x)$ where $p_i\sim\mathcal{N}(\mu_i,\sigma_i^2),\lambda\in(0,\frac{1}{2}]$. Considering each mixture component separately and applying the lemma on the total variation of mixture distributions (5.1) we find,*

$$\begin{aligned}\mathrm{TV}(p_t,q) \leq e^{-2\beta t}&\left[\frac{\lambda|(\sigma_1^2-\frac{\sigma^2}{2\beta})|}{2\max\left\{\frac{\sigma^2}{2\beta}+(\sigma_1^2-\frac{\sigma^2}{2\beta})e^{-2\beta t},\frac{\sigma^2}{2\beta}\right\}}+\frac{(1-\lambda)|(\sigma_2^2-\frac{\sigma^2}{2\beta})|}{2\max\left\{\frac{\sigma^2}{2\beta}+(\sigma_2^2-\frac{\sigma^2}{2\beta})e^{-2\beta t},\frac{\sigma^2}{2\beta}\right\}}\right]+\\ e^{-\beta t}&\left[\frac{\lambda|\mu_1-\alpha|}{2\max\left\{\sqrt{\frac{\sigma^2}{2\beta}+(\sigma_1^2-\frac{\sigma^2}{2\beta})e^{-2\beta t}},\frac{\sigma}{\sqrt{2\beta}}\right\}}+\frac{(1-\lambda)|\mu_2-\alpha|}{2\max\left\{\sqrt{\frac{\sigma^2}{2\beta}+(\sigma_2^2-\frac{\sigma^2}{2\beta})e^{-2\beta t}},\frac{\sigma}{\sqrt{2\beta}}\right\}}\right].\end{aligned}$$

This implies, under an Ornstein-Ulhenbeck perturbation process, the convergence of the initial mixture to the stationary distribution is exponentially fast, with respect to the total variation. The exponential rate of convergence specified by β.

Experiment: Forward Convergence Of One Dimensional Gaussian Mixture Models

In this experiment, we consider the initial distribution to be a Gaussian mixture, $p_0\sim\frac{1}{2}\mathcal{N}(\mu,1)+\frac{1}{2}\mathcal{N}(-\mu,1)$. Following the above results, we aim to demonstrate empirically how β should scale as μ increases. In order to maintain the stationary distribution as β varies, we specify an Ornstein-Ulhenbeck process with $\alpha=0,\ \sigma=\sqrt{2\beta}$, which has stationary distribution $\mathcal{N}(0,\frac{2\beta}{\sigma^2})=\mathcal{N}(0,1)$. From the above results, we expect β to scale logarithmically with the approximate diameter of the data from the stationary mean $(\mu-\alpha)$. We considered exponentially increasing values of $\mu\in\{1e0,1e0.5,1e1,1e1.5,1e2,1e2.5,1e3,1e3.5,1e4,1e4.5,1e5\}$ and β uniformly over the interval $[1,20]$. In Figure 5.1 we record the smallest β for which the Wasserstein distance between the end of the forward process p_1 and the stationary distribution $p_\infty\sim N(0,1)$ is less than 0.05.

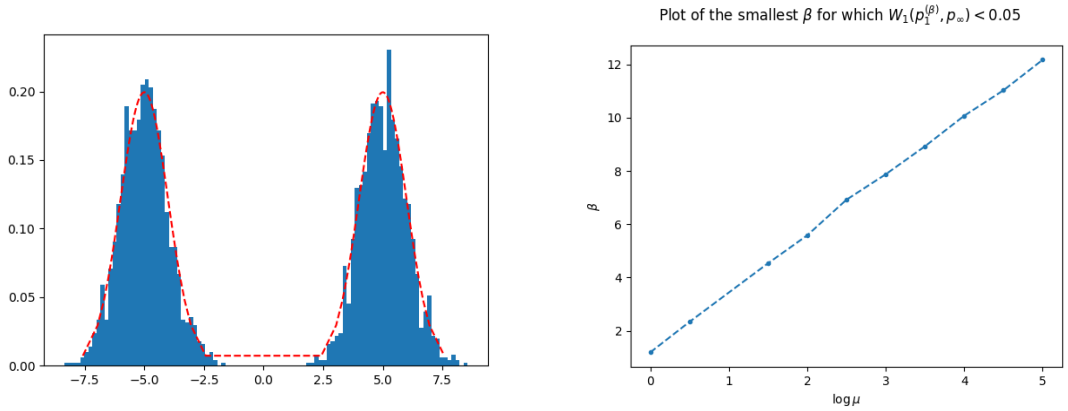

Figure 5.1: On the left-hand side we see an example of the initial data distribution for $\mu = 10$ with the probability density in red. The right-hand side demonstrates the relationship between the diameter of the data, $\mu - \alpha$ and the corresponding parameter β in the perturbation process.

From this diagram we can actually fit a linear model to the data, to derive a relationship on how β should scale with the dispersion of the mixture component means μ from the stationary mean α. Fitting this linear model, we find the gradient is ≈ 2.19283. Therefore,

$$\beta = 2.19283 \log_{10}(\mu - \alpha) \implies \beta = \frac{2.19283}{\ln(10)} \ln(\mu - \alpha) \implies \beta \approx 0.95 \ln(\mu - \alpha).$$

In the case of one-dimensional Gaussian distributions, β should scale logarithmically with the distance between mixture component means and stationary mean α.

Higher dimensional Gaussian mixture models

We now aim to generalise this result to higher dimensional Gaussian mixture models. To approach this, we consider the KL-divergence between the marginal distribution p_t and stationary distribution p_∞ and aim to apply Pinsker's inequality (3.1).

Lemma 5.3 (KL-Divergence between Multivariate Gaussians [37])**.**

$$\mathrm{KL}(\mathcal{N}(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1}) || \mathcal{N}(\boldsymbol{\mu_2}, \boldsymbol{\Sigma_2})) \leq \frac{1}{2} trace(\boldsymbol{\Sigma_1}^{-1}\boldsymbol{\Sigma_2} - \boldsymbol{I}_n) + (\boldsymbol{\mu_1} - \boldsymbol{\mu_2})^T \boldsymbol{\Sigma_1}^{-1}(\boldsymbol{\mu_1} - \boldsymbol{\mu_2}) - \log\det(\boldsymbol{\Sigma_2}\boldsymbol{\Sigma_1}^{-1}).$$

Corollary 5.4 (Proposition 2.1, [37])**.** *Upper-bound on total variation between multivariate Gaussian distributions. Applying Pinsker's inequality to* (5.3) *we find,*

$$\mathrm{TV}(\mathcal{N}(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1}), \mathcal{N}(\boldsymbol{\mu_2}, \boldsymbol{\Sigma_2})) \leq \frac{1}{2}\sqrt{\mathrm{Tr}(\boldsymbol{\Sigma_1}^{-1}\boldsymbol{\Sigma_2} - \boldsymbol{I}_n) + (\boldsymbol{\mu_1} - \boldsymbol{\mu_2})^T \boldsymbol{\Sigma_1}^{-1}(\boldsymbol{\mu_1} - \boldsymbol{\mu_2}) - \log\det(\boldsymbol{\Sigma_2}\boldsymbol{\Sigma_1}^{-1})}.$$

Consider the initial data distribution to be a multivariate Gaussian mixture given by $p_0(x) = \lambda\mathcal{N}(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1}) + (1-\lambda)\mathcal{N}(\boldsymbol{\mu_2}, \boldsymbol{\Sigma_2})$, with $\lambda \in (0, \frac{1}{2}]$ as before. Following a similar justification as in the one-dimensional case, we find the stochastic process of X_0 under OU perturbation is a Gaussian process and therefore has Gaussian distributed marginals (see Appendix A.1). By considering the Gaussian form of the marginal and applying the total variation mixture Lemma 5.1 in conjunction with the total variation between Gaussian distributions (Corollary 5.4), we arrive at the following upper-bound on the total variation of the forward process.

Proposition 5.1. *For an Ornstein-Ulhenbeck perturbation process, with invariant distribution* $q = \mathcal{N}\left(\boldsymbol{\alpha}, \frac{\sigma^2}{2\beta}\mathbf{I}_n\right)$*, applied to a multivariate Gaussian mixture model,* $p_0 \sim \lambda\mathcal{N}(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1}) + (1-\lambda)\mathcal{N}(\boldsymbol{\mu_2}, \boldsymbol{\Sigma_2})$ *we find:*

$$\mathrm{TV}(q, p_t) \leq e^{-\beta t}\Big[\frac{\lambda}{2}\left(\mathrm{Tr}(\boldsymbol{\Sigma}_1 - \boldsymbol{I}_n) + ||\boldsymbol{\mu}_1 - \boldsymbol{\alpha}||_2^2 \frac{\sigma^2}{2\beta}\right)^{\frac{1}{2}} + \frac{(1-\lambda)}{2}\left(\mathrm{Tr}(\boldsymbol{\Sigma}_2 - \boldsymbol{I}_n) + ||\boldsymbol{\mu}_2 - \boldsymbol{\alpha}||_2^2 \frac{\sigma^2}{2\beta}\right)^{\frac{1}{2}}\Big].$$

under some weak conditions on the magnitude of β*.*

Proof. See Appendix A.3.1. □

Therefore, we find our result of exponentially fast convergence in the forward process, with respect to the total variation, extends to multivariate Gaussian mixture initial distributions too.

5.1.2 Forward Convergence On Arbitrary Data Distributions

Finally, we wish to generalise the above results of the convergence in the forward process, with respect to the total variation, to arbitrary initial data distributions p_0. Specifically, we aim to find an expression for the exponential rate of convergence in terms of the free parameters of the perturbation process: α, β and σ. In order to do this, we introduce some results from stochastic calculus.

Definition 5.1 (Logarithmic-Sobolev inequality [38])**.** *A probability measure p defined on $\mathbb{R}^n$ satisfies the logarithmic-Sobolev inequality ('LSI') with constant C if for all smooth functions $f : \mathbb{R}^n \to \mathbb{R}$ with $\mathbb{E}_p[f^2] = \int_{\mathbb{R}^n} f(x)^2 p(x)dx < \infty$ then:*

$$\mathbb{E}_p[f^2 \log f^2] - \mathbb{E}_p[f^2] \log \mathbb{E}_p[f^2] \leq \frac{2}{C} \mathbb{E}_p \left[||\nabla_x f||_2^2 \right].$$

Lemma 5.4 (Exponential convergence in the KL-divergence for measures satisfying the logarithmic-Sobolev inequality [39])**.** *If the probability measure p satisfies the LSI 5.1 with constant $C > 0$. Then under under the following Ornstein-Ulhenbeck perturbation $dX_t = -\beta(X_t - \alpha) + \sigma dW_t$,*

$$\mathrm{KL}(p_t||p_\infty) \leq e^{-2\beta C T} \mathrm{KL}(p_0||p_\infty).$$

Proposition 5.2. *The invariant distribution of an Ornstein-Ulhenbeck process, p_∞, satisfies the LSI with constant $\frac{2\beta}{\sigma^2}$.*

Proof. See Appendix A.4.1. □

Theorem 5.2 (Forward process converges exponentially fast and with respect to total variation and the rate of convergence is a function of β and σ.)**.** *Assuming p_0 has a finite second moment,*

$$\mathrm{TV}(p_t, p_\infty) \leq \frac{1}{\sqrt{2}} \sqrt{\mathrm{KL}(p_t||p_\infty)} \leq \frac{1}{\sqrt{2}} e^{-2\beta t} \sqrt{\mathrm{KL}(p_0||p_\infty)}.$$

Under this formulation we have only proved exponentially fast convergence in total variation providing that $\mathrm{KL}(p_0||p_\infty) < \infty$.

Proof. This follows directly from applying Pinsker's inequality 3.1 in conjunction with Lemma 5.4 and Proposition 5.2. □

Notably, the requirement of p_0 having a finite second moment is necessary to ensure our upper-bound converges to zero. For example, if we consider the case in which p_0 does not have a finite second moment, we can derive:

$$\begin{aligned}
\mathrm{KL}(p_0||p_\infty) &= \int_{\mathbb{R}^n} p_0 \log \frac{p_0}{p_\infty} dx = \int_{\mathbb{R}^n} p_0 \log p_0 dx - \int_{\mathbb{R}^n} p_0 \log p_\infty dx \\
&= \int_{\mathbb{R}^n} p_0 \log p_0 dx + \frac{\beta}{\sigma^2} \int_{\mathbb{R}^n} p_0 ||\boldsymbol{x} - \boldsymbol{\alpha}||_2^2 dx \\
&= \int_{\mathbb{R}^n} p_0 \log p_0 dx + \frac{\beta}{\sigma^2} \mathbb{E}_{p_0} \left[||\boldsymbol{x} - \boldsymbol{\alpha}||_2^2 \right] = \infty.
\end{aligned}$$

Therefore, the upper-bound remains unbounded for all t.

Experiment: Forward Convergence Of Data Lying On A Disjoint Support

In this experiment, we consider a data support satisfying the union of manifolds hypothesis (3.1), specifically the union of circles dataset. This dataset is perturbed under an Ornstein-Ulhenbeck process with $\alpha = 0, \sigma = \sqrt{2\beta}$, which is discretised with $dt = 0.01$ over the interval $[0, 1]$. We aim to find a suitable value of β as we vary the diameter of the dataset - which is the furthest point in the support from the long-term mean α. In order to increase the diameter, we will vary the centre of each circle to be $(\pm\mu, 0)$. Evidently, with circles of radius 1 the diameter of the dataset is therefore $\mu + 1$. Similarly to the one dimensional Gaussian mixture case, in this experiment, the origin of each support component is varied exponentially. $\mu \in \{1e0, 1e0.5, 1e1, 1e1.5, 1e2, 1e2.5, 1e3, 1e3.5, 1e4, 1e4.5, 1e5\}$ and β uniformly over the interval $[1, 20]$. To carry out this experiment, I generated $5,000$ samples around each component of the support and $5,000$ samples from the stationary distribution $p_\infty \sim N(\mathbf{0}_2, \boldsymbol{I}_2)$. I subsequently calculated the sliced-Wasserstein distance (with 100 projections) between the perturbed samples and those drawn from the stationary measure. In Figure 5.2 we record the smallest β for which the sliced-Wasserstein distance is less than 0.05.

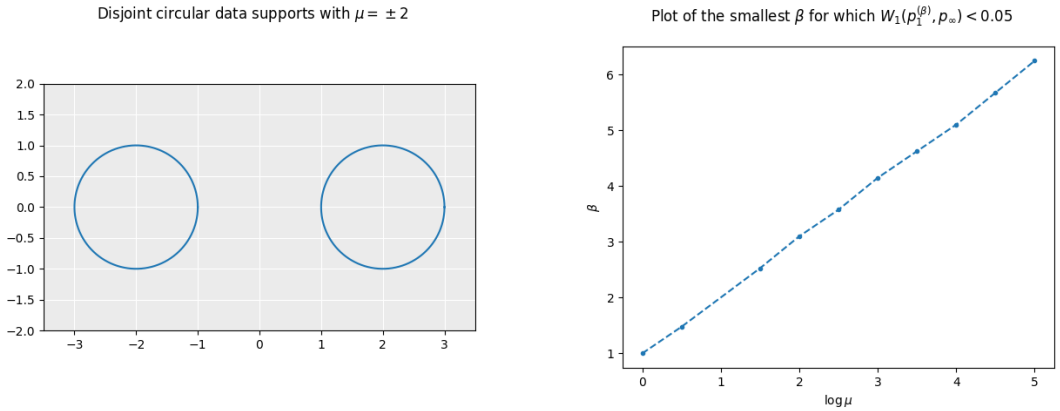

Figure 5.2: On the left-hand side we see and example initial data distribution with disjoint support, for $\mu = \pm 2$. On the right-hand side, we observe the relationship between the log diameter of the data and the minimal β for which the sliced Wasserstein distance after perturbation is < 0.05.

If we fit a linear model to this data, we find the exponential convergence holds again, although this time with a different constant governing the logarithmic rate.

$$\beta \approx 0.45 \ln(\mu - \alpha).$$

This experiment is a nice proxy for real datasets on bounded disjoint supports. This shows we can diminish the first term bound of the total variation in the backwards process (3.3) by identifying an appropriately large β. It shows experimentally exponential convergence holds in the forward process for data distributions with disconnected support, indeed with this dataset explicitly, β only needs to scale by less than half the logarithm of the diameter of the data.

Experiment: Pathological Case Of Data With Undefined Moments

Finally, we demonstrate forward convergence is still attainable even in the case the moments of the initial distribution are not finite, although our proof does not cover these cases. As an example, we consider the Cauchy distribution which is defined on $\mathbb{R}$ as follows,

$$p_{x_0,\gamma}(x) = \frac{1}{\pi\gamma\left(1 + \left(\frac{x - x_0}{\gamma}\right)^2\right)}, \quad F_{x_0,\gamma} = \frac{1}{2} + \frac{1}{\pi}\arctan\left(\frac{x - x_0}{\gamma}\right).$$

Evidently, the cumulative density function is bijective and so $F_{x_0,\gamma}$ is invertible, with inverse $F^{-1}_{x_0,\gamma}(x) = x_0 + \gamma \tan\left[\pi\left(x - \frac{1}{2}\right)\right]$. So, we can easily generate samples from a Cauchy distribution via inversion sampling. The Cauchy distribution is a classic pathological case for many results, as the first moment is undefined and the second moment is infinite. Despite this, we will see that we

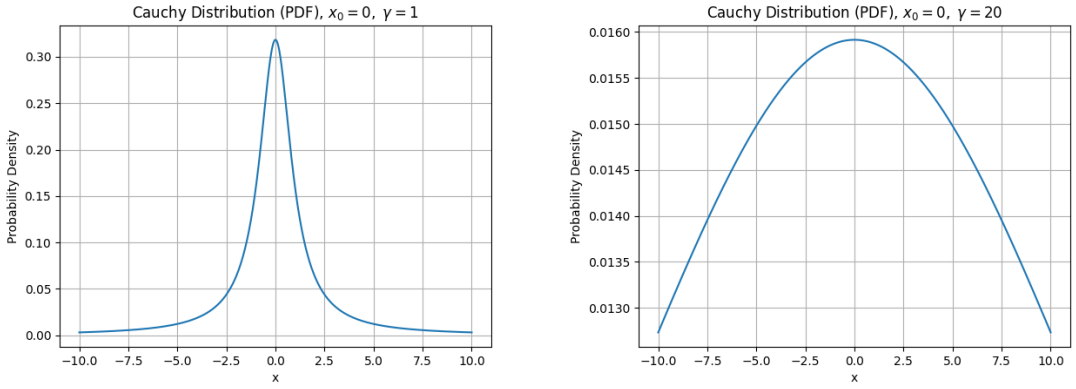

Figure 5.3: The probability density of two Cauchy distributions with $x_0 = 0$, $\gamma = 1$ on the left and $\gamma = 20$ on the right.

attain convergence under the Wasserstein metric to the stationary distribution, $p_\infty \sim N(0, 1)$. In this experiment I have perturbed samples drawn from a Cauchy distribution ($x_0 = 0,\ \gamma = 5$) with the Ornstein-Ulhenbeck process, $\alpha = 0,\ \beta = 10,\ \sigma = \sqrt{2\beta} = 2\sqrt{5}$. We observe the Wasserstein distance and distribution of samples at p_1 in Figure 5.4 below.

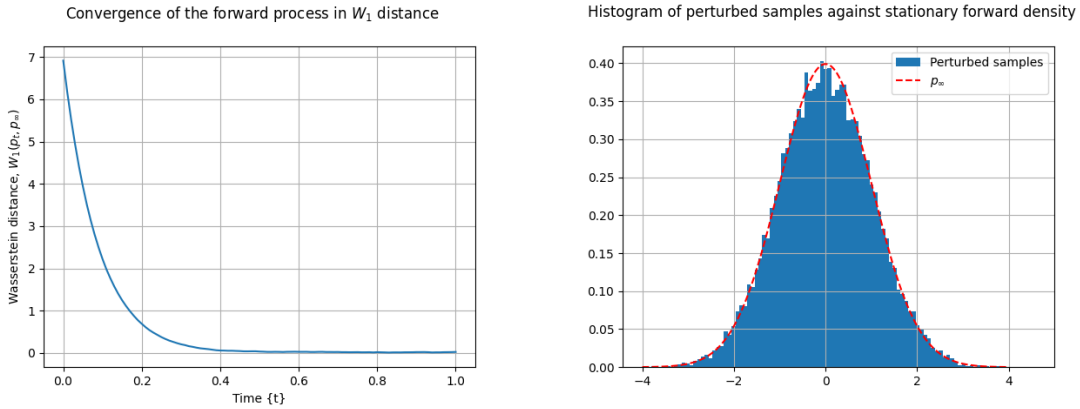

Figure 5.4: On the left-hand side we see the Wasserstein distance between the perturbed samples and stationary distribution as the process is run forward in time. On the right-hand side we see a histogram of the perturbed samples at $T = 1$, with the density of the stationary distribution p_∞ overlaid.

To justify why the samples still converge, we make an approximation and consider e^{-5} to be sufficiently small to be ignore the relevance of the initial point x_0 in the forward measure at time $T = 1$. Under this perturbation we find all samples in $[-e^5, e^5]$ converge. Considering the cumulative density function this represents 98% of all samples generated from the initial distribution.

$$\frac{1}{\pi}\left(\arctan(e^5/5) - \arctan(-e^5/5)\right) = 0.979.$$

5.2 Parameter Influence On The Discretisation Error

The above conclusions on minimising total variation in the forward process suggests an optimal β can be arbitrarily large, as this rapidly removes the most information of the initial distribution and minimises $\mathrm{TV}(p_1, p_\infty)$. In this section, we will discuss errors caused by the discretisation in relation to β. This discussion will be motivated with an example of the discretising the forward convergence of a simple Gaussian mixture distribution under Ornstein-Ulhenbeck perturbation.

Experiment: Strong Convergence Of The Forward Process With Respect To β

In this experiment, we aim to demonstrate how the quantity of interest in the strong convergence of a discretisation scheme (2.10), which is $\mathbb{E}[|X_1 - \hat{X}_1^{\Delta t}|]$, increases as β increases. This is tractable in the case of an Ornstein-Ulhenbeck perturbation as we have an analytic form of the process, given by

$$X_t = e^{-\beta t} X_0 + \sqrt{2\beta} \int_0^t e^{-\beta(t-s)} dW_s.$$

Furthermore, our Euler-Maruyama discretisation (2.9) is given by,

$$\hat{X}_{j\Delta t}^{\Delta t} = \hat{X}_{(j-1)\Delta t}^{\Delta t} - \beta \hat{X}_{(j-1)\Delta t}^{\Delta t} \cdot \Delta t + \sqrt{2\beta}\Delta(W_t).$$

Importantly, in order to compare the path-wise differences in each scheme, each path under each formulation must be driven by the same underlying Brownian motion. Therefore, in this experiment, I considered a range of values Δt. For each Δt, I varied β uniformly over the range $[11, 99]$. I generated $10,000$ sample paths from a one dimensional Gaussian mixture distribution with $\mu = \pm 5$, $\sigma = 1$. Finally, I took an expectation over the generated paths of the absolute difference between the two processes. The results are presented in Figure 5.5.

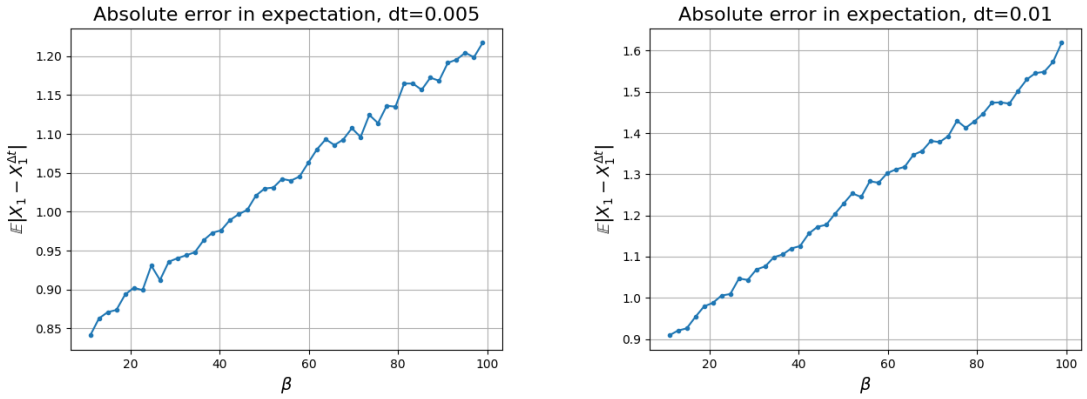

Figure 5.5: On the left-hand side ($\Delta t = 0.005$) we see how the discretisation error at $t = 1$ linearly increases as β increases. On the right-hand side ($\Delta t = 0.01$) we see this relationship is maintained, however as expected the magnitude of the error is greater for larger Δt.

In this case, we observe a linear relationship between β and the discretisation error at $t = 1$. This implies there is a trade-off in β between the bound on forward convergence and the discretisation error of the backward process too. That is to say, an arbitrarily large β will effect the convergence of the backward process.

5.2.1 Reformulating The Discretisation Error For A General OU Process

Motivated by this observation, we aim to formally derive the influence of β on the discretisation error in the bound of the total variation of the backward process. In order to construct this result we introduce similar notation as in Theorem 3.3.

1. Let p_t denote the forward process under an Ornstein-Ulhenbeck perturbation given by $dX_t = -\beta X t dt + \sqrt{2\beta} dW_t$, with $p_0 = p_{data}$ and p_∞ the unique stationary distribution, $N(\mathbf{0}_n, \boldsymbol{I}_n)$.

2. Let q_t denote the reverse process $dX_t = (\beta X_t + 2\beta \nabla_x \log p_{T-t}(x))dt + \sqrt{2\beta} dW_t$. This process is initialised at the stationary distribution of the forward process and therefore $q_0 = p_\infty$. The law of this process is denoted by Q_t.

3. Let $Q_t^{p_T}$ denote the law of the discretised score-based generative modelling algorithm which follows the following discretisation of the reverse process, $dX_t = (\beta X_t + 2\beta S(X_{j\Delta t}, T - j\Delta t; \theta))dt + \sqrt{2\beta} dW_t$, $t \in [j\Delta t, (j+1)\Delta t]$, initialised at the end of the forward process, $X_0 \sim p_T$.

We present the following conjecture regarding the general discretisation and score approximation error.

Conjecture 5.1 (Discretisation error of the backward process for a general Ornstein-Ulhenbeck perturbation process)**.** *Under the same assumptions outlined in Theorem 3.2, we believe, the total variation between the reverse process and score-based generative modelling algorithm is bounded above by the following quantity,*

$$\mathrm{TV}(Q_T, Q_t^{p_T})^2 \leq \mathrm{KL}(Q_T || Q_t^{p_T}) \lesssim (L^2 dh\beta + L^2 m_2^2 h^2 \beta^2 + \epsilon_{score}^2)T.$$

From which we can conclude,

$$\mathrm{TV}(Q_T, Q_t^{p_T}) \lesssim (L\sqrt{dh}\sqrt{\beta} + \beta L m_2 h + \epsilon_{score})\sqrt{T} \lesssim \beta(L\sqrt{dh} + Lm_2 h)\sqrt{T} + \epsilon_{score}\sqrt{T}.$$

Proof. The proof is sketched in the Appendix A.5 and largely based on the discretisation of denoising diffusion probabilistic models presented by Chen Et al. [36, Theorem 9]. □

We remark the linearity of β in the discretisation error coincides with the results of the previous experiment in quantifying the discretisation error in the forward process, under a more general OU forward perturbation process.

5.3 Convergence In The Backward Process

Based on the results derived on the affect of β on the rate of convergence (Theorem 5.2) and discretisation error (Conjecture 5.1), and the inequality presented in (3.2), we derive the following result on the total variation between the backward process and data distribution.

Conjecture 5.2 (Total variation bound on the backward process under a general OU process)**.** *Under the standard assumptions asserted in Theorem 3.2, the total variation between the backwards process and initial data distribution, perturbed in forward time by the an Ornstein-Ulhenbeck process specified by $dX_t = -\beta X_t dt + \sqrt{2\beta} dW_t$, is given by,*

$$\mathrm{TV}(q_T, p_0) \lesssim \underbrace{e^{-\beta T}\sqrt{\mathrm{KL}(p_0||p_\infty)}}_{\textit{Convergence in forward process}} + \underbrace{\beta(L\sqrt{dh} + Lm_2 h)\sqrt{T}}_{\textit{Discretisation error}} + \underbrace{\epsilon_{score}\sqrt{T}.}_{\textit{Score approximation error}}$$

We note the requirement of finite KL-divergence between the data and stationary distributions follows from the finite second moment assumption of p_0, asserted in Theorem 3.2. This result establishes what we set out to achieve at the beginning of the previous section, in that we cannot set β arbitrarily large and it should in fact be appropriately determined to ensure a balance in both the attaining forward convergence and minimising the discretisation error. We will now explore experimentally what affect varying β has on the quality of generated samples on a variety of datasets.

5.3.1 Experiments On The Affects Of Varying Perturbation Parameters On The Quality Of Generated Samples

We now aim to demonstrate that Conjecture 5.2 holds experimentally on a range of datasets. In each of the below experiments, we trained a score-network for $5,000$ epochs on 500 samples from the parent distribution. A large number of training samples were provided to minimise the score approximation error, this ensures we are just evaluating errors due to varying β on the first two terms in Conjecture 5.2. The exact specifications of the training and sample generation processes can be found in the Appendix B.1 and implementation section 2.3.

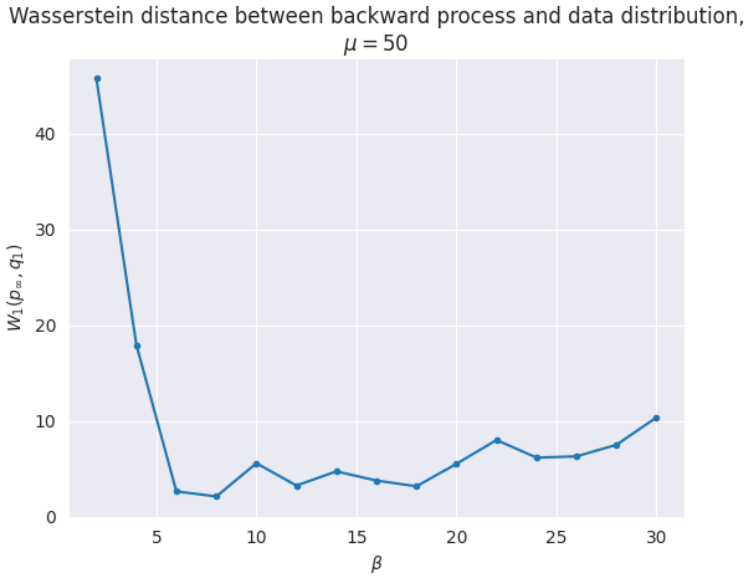

Figure 5.6: Here we see the quality of the generated samples under the discretised backward process $q_1^{\Delta t}$, measured by the Wasserstein distance from the true Gaussian mixture distribution, as β is varied over the range $\{2, \ldots, 30\}$.

Experiment: Perturbation Process Parameter Influence In Score-Based Generative Modelling of A One Dimensional Gaussian Mixture

We start considering one dimensional Gaussian mixture model specified by $p_0 \sim \frac{1}{2}\mathcal{N}(-50, 1) + \frac{1}{2}\mathcal{N}(50, 1)$. We aim to verify experimentally that the affect of varying β exhibits the same relationship as expected by Conjecture 5.2. Training our score-network as specified above, we present the plot of the Wasserstein distance between the discretised backward process and true data distribution p_0 in Figure 5.6.

In Figure 5.6, we see the initial exponential decay in the Wasserstein distance as β grows to an adequate size to ensure convergence in the forward process. This is followed by a linear increase caused by discretisation error as β further increases.

Experiment: Perturbation Process Parameter Influence In Score-Based Generative Modelling On the Circle Dataset

We will now consider the circle dataset illustrated in Figure 4.1, although with many more training points. As previously alluded to, this dataset is a reasonable abstraction of more complicated datasets which conform to the union of manifolds hypothesis. In order to clearly illustrate the forward convergence as β grows, I used a circle of radius 10 and altered the true generalisation metric accordingly. In Figure 5.7 we present the relationship between β and generated samples, measured under the true circle generalisation metric (4.2). Here we also observe the relationship specified by Conjecture 5.2. In this case, we can use some functions from the diffusion Jax library [24] to nicely visualise the learned score function and generated samples, these are presented in Figure 5.8.

Experiment: Perturbation Process Parameter Influence In Score-Based Generative Modelling On The Swiss Roll Dataset

Finally, we aim to demonstrate this result holds on the Swiss roll dataset introduced in Section 4.2.3. Due to the higher dimensionality of this dataset I trained the score-network on $1,500$ training points, as opposed to 500 in the previous two examples. This is, again, to ensure the score-network well approximates the true score and therefore any degradation in sample quality is caused primarily by the first two terms in Conjecture 5.2.

In order to measure generalisation, I used the sliced Wasserstein distance with 100 projections. I measured this distance between $3,500$ generated samples and $3,500$ withheld samples generated

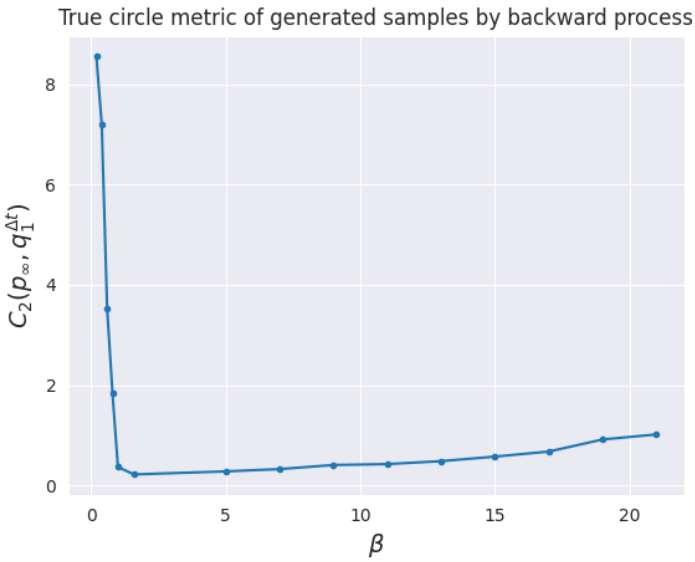

Figure 5.7: Here we see the quality of the generated samples under the discretised backward process $q_1^{\Delta t}$, measured by the true circle generalisation metric, C_2, as β is varied over the range $[0.2,\ 23]$.

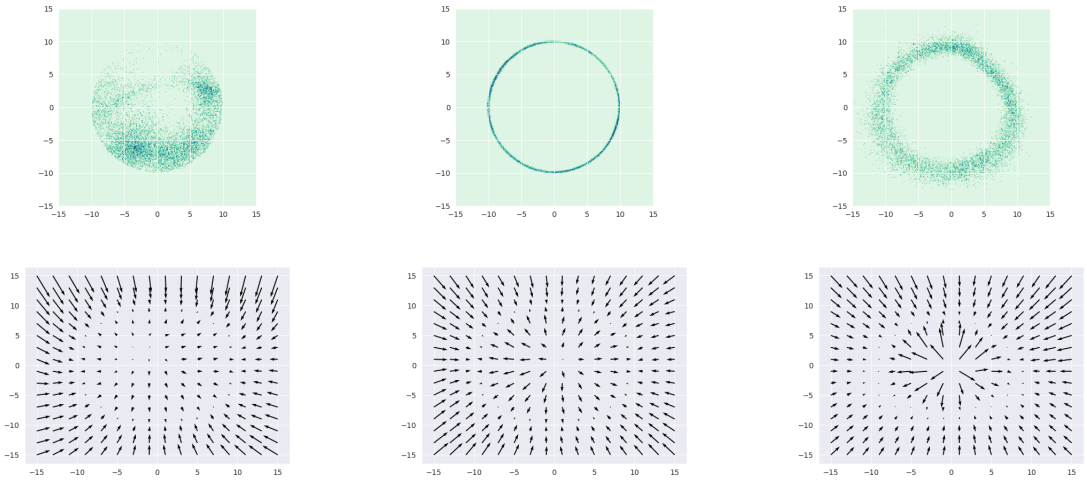

Figure 5.8: In the top row, we present a heatmap of the generated sample at $\beta = 0.6,\ 1.4$ and 23 respectively. The bottom row shows the corresponding learned score functions evaluated at $t = 0.01$.

on the support separately from the training set. The results of this experiment are presented in Figure 5.9. Again, we observe the initial exponential decrease in sliced Wasserstein distance, owing to the forward convergence term. Then, we observe a mild linear increase as the discretisation error gradually increases for greater values of β. While the increase in sliced Wasserstein distance for greater values of β looks mild in comparison to the those observed at lower values of β, we present the severity of same degradation visually in Figure 5.10.

5.3.2 Comparison To Push-Forward Models In Generating Samples From Gaussian Mixture Distributions

In the related work section, we recalled a result which shows an intrinsic relation between the distance of mixture component means and the Lipschitz constant of the push-forward network (Proposition 3.1). We now reason whether score-based models suffer the similar limitations in generating samples from Gaussian mixture distributions.

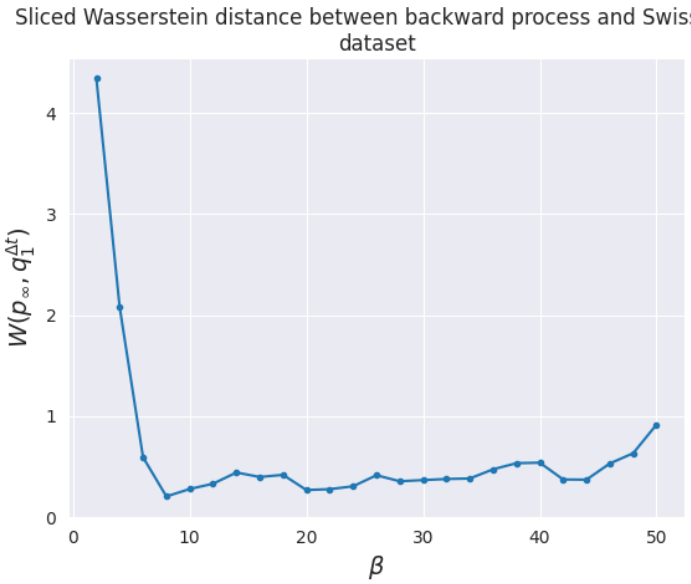

Figure 5.9: Here we see the quality of the generated samples under the discretised backward process $q_1^{\Delta t}$, measured by the sliced Wasserstein distance from the withheld Swiss roll test set, as β is varied over the range $\{2, \ldots, 50\}$.

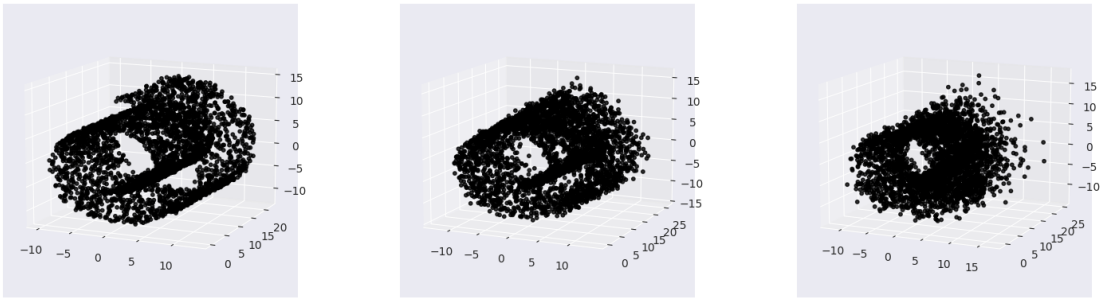

Figure 5.10: This figure visualises the samples generated by the discretised reverse process, $q_1^{\Delta t}$ under different values of β. The left-hand diagram corresponds to the optimal $\beta = 8$ from the experiment, for which we observe the generated samples well approximate the true support. We see the quality of generated samples degrade in the middle and right-hand images, which are associated with $\beta = 30$ and $\beta = 50$ respectively.

As demonstrated in the forward convergence section, in order to ensure forward convergence with Gaussian mixture data distributions, β should scale logarithmically with the distance of mixture component means from the stationary mean α (see Figure 5.1). In our formulation of convergence in the backward process, we see scaling β logarithmically incurs a logarithmic cost in the discretisation error. An argument could be made that the discretisation error cost could be mitigated by decreasing the step size Δt. However, following the training procedure, this requires training our score-network at more points in the interval $[0, 1]$. Learning more evaluation times of the score function would demand a larger neural network, which is in effect similar to increasing the Lipschitz constant of the push-forward network in Proposition 3.1. Intuitively, we would only need to scale the network size logarithmically with the distance of mixture component means from the stationary mean α.

5.4 Regularisation Techniques for Accurate Score Approximation

We now focus on how to minimise the error in approximating the true score function of the parent distribution. This corresponds to the final term, ϵ_{error}, in Theorem 3.3 and our generalised

Conjecture 5.2. Importantly, as we aim to model the score function of the parent distribution, model score functions which overfit to the training set will suffer a greater error. This is exactly what we observed in the motivating example of the generalisation property of score-based generative models in Section 3.2. In this example we demonstrated we can prevent over-fitting in training the score-network by invoking early stopping - a classical machine learning regularisation technique. In this section, we aim to investigate further how applying regularisation techniques in training the score-network affects the parent score function approximation and consequently the quality of generated samples.

5.4.1 Number Of Samples

Firstly, we reason how increasing the number of samples in the training set (denoted by μ_{data}) makes our score network less susceptible to over-fitting and produces a more accurate model of the true score. This might seem obvious intuitively, however it also follows theoretically from the definition of the de-noising score matching objective (2.17) - which serves as the loss function of our score-network. We replicate the definition below,

$$\begin{aligned} J(\theta) &= \mathbb{E}_{t\in[0,t]}\left[\mathbb{E}_{p(x_t)}[||\nabla_x \log p_t(x) - S(x,t;\theta)||^2]\right] \\ &= \mathbb{E}_{t\in[0,t]}\left[\mathbb{E}_{p(x_0)}\mathbb{E}_{p(x_t|x_0)}[||\nabla_x \log p(x_t|x_0) - S(x,t;\theta)||^2]\right] && \text{(Law of total expectation)} \\ &\approx \mathbb{E}_{t\in[0,t]}\left[\mathbb{E}_{\mu_{data}}\mathbb{E}_{p(x_t|x_0)}[||\nabla_x \log p(x_t|x_0) - S(x,t;\theta)||^2]\right] && (\mu_{data} \sim p_0) \end{aligned}$$

In the estimate of the second expectation, $\mathbb{E}_{p_0}$, we assume the training set is comprised of independently drawn samples from the parent distribution, this estimate is obviously better with more samples. Therefore, with a more extensive training set, our model score function is trained on a better estimate of the score matching objective associated with the true data score. This supports the well-founded intuition that a training set which is more representative of the parent distribution leads to a score-network closer which approximates the true data score. Therefore, in reality, instead of immediately applying complicated regularisation techniques, we should always first think if it is feasible to acquire more data.

5.4.2 Network Architecture Size

We previously demonstrated our model score is susceptible to over-fitting to the score function of the training set. Unfortunately, it is very difficult to detect when this is happening during training, so far our justifications have come from generating samples throughout the training process and using metrics to identify when generalisation is getting worse. This is not practically feasible as sample generation in score-based modelling requires running the discretised backward diffusion - an inherently sequential, and therefore slow, procedure. This encourages other regularisation techniques which do not require sample generation throughout training.

More generally in machine learning, over-fitting occurs when the neural network learns the training set too well, to the point it starts to memorise patterns in the training set over more general observations. This is closely related to the capacity of the network - which refers to the number of free parameters or the complexity of the model. Larger neural networks with more neurons and layers are regarded as having greater capacity. While these networks are able to infer more complex relationships in the data, they're also more susceptible to over-fitting. This motivates the experiments in this section, which are in investigating how changes to the capacity of the score-network prevent over-fitting and affect the model's generalisation capabilities. We now demonstrate this relation experimentally.

As in the previous experiments, the exact specifications of the training and sample generation processes can be found in the Appendix B.1 and implementation section 2.3, although we will be varying the network architecture and number of training epochs throughout this experiment. Furthermore, β is appropriately chosen to ensure forward convergence and an adequate discretisation error. The neural network architectures considered are as follows: {3L16N, 3L64N, 3L256N, 5L16N, 5L64N} whereby the first digit corresponds to the number of hidden layers and the second the number of nodes per layer in the multi-layer perception network. The score-network is trained in intervals of $1,000$ training epochs. Samples are generated and generalisation is measured after each interval.

Experiment: The Affects Of Varying The Score-Network Capacity In Generalising The Circle Dataset

We consider the unit circle dataset with 8 training points, visualised in Figure 4.1. This is similar to the setup of the initial demonstration of the generalisation property in Section 3.2. In order to measure generalisation, we consider two circle generalisation metrics. Firstly, the C_1 metric applied to the training set, which identifies when the training set is being overly reproduced in the generated samples. Secondly, we consider the C_2 metric to measure generalisation on the true support. We divide these metrics into three components. The first component measures the distribution of the radii of generated samples, which corresponds to the distance from the target support. The second component measures the distribution of the angle of generated samples, which quantifies the dispersion around the target support. The third component measures the sum of the previous two and serves an aggregate. The measure of generalisation with respect to the training set and true distribution are presented in Figures 5.11 and 5.12 respectively.

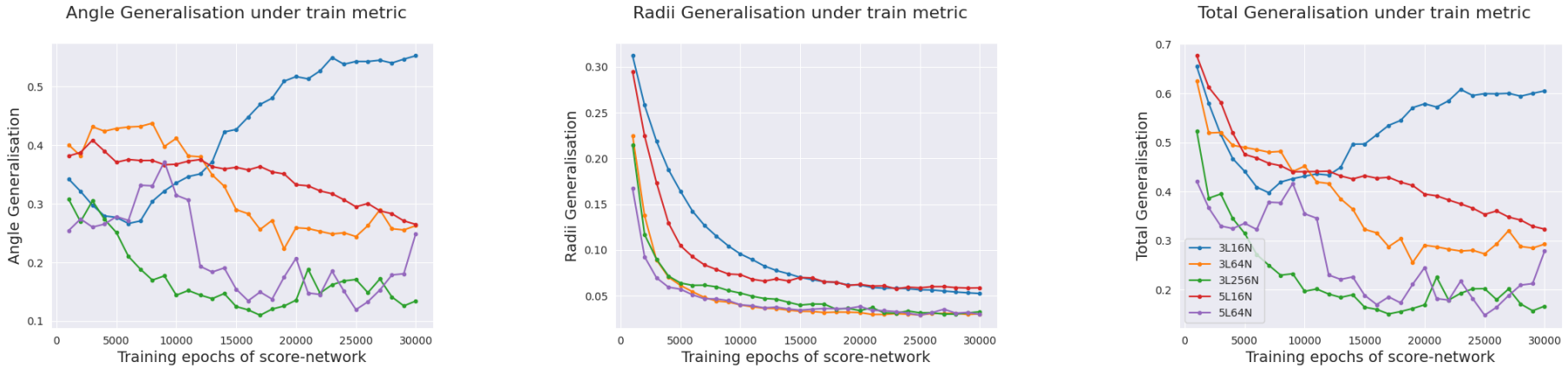

Figure 5.11: In this figure we present the measure of how well the generated samples resemble the training set, as the score-network is trained up to $30,000$ epochs.

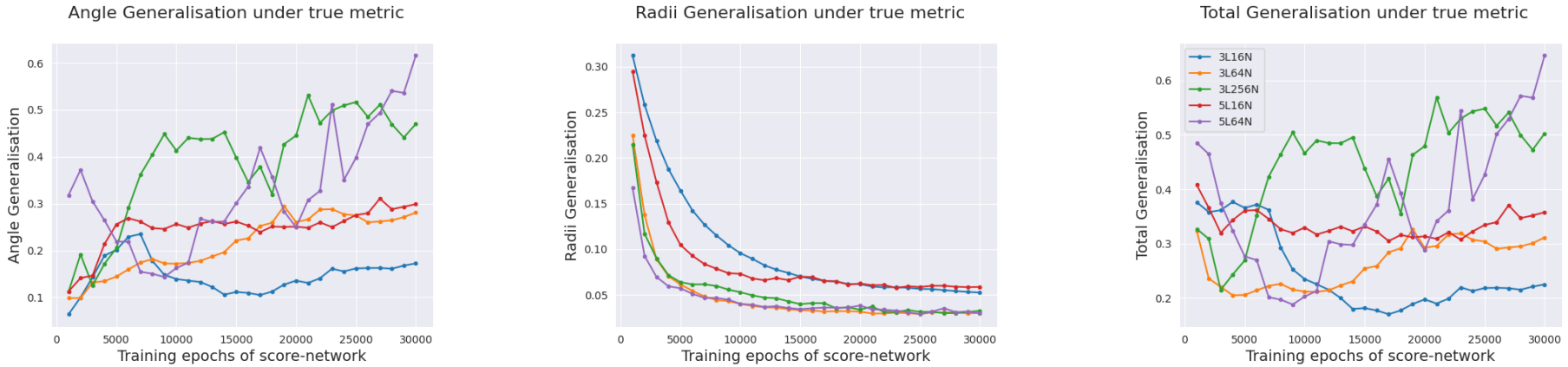

Figure 5.12: In this figure we present the measure of how well the generated samples fit the true support of the data distribution, as the score-network is trained up to $30,000$ epochs.

We first discuss how susceptible the different architectures are to reproducing the training set, throughout the training process. This analysis corresponds to Figure 5.11. In this figure, we observe smaller networks exhibit poorer reproduction of the training set. This is indicated by the greater measure of angle generalisation against the training set, which indicates the produced samples vary in angle from the training set. Conversely, as the training process continues, larger networks with greater capacity produce samples closely aligned to the training set, this is indicated by considering the total generalisation with respect to the training process, illustrated in sub-figure 3 of Figure 5.11. These results quantify the relation between training and overfitting that was presented initially in Section 3.2.

Considering now the measure of generalisation against the true support, we observe larger networks reach an optimal approximation of the true score early in the training process. As the training continues, larger score-networks closer approximate the score of the training set. This is seen by the worsening of angle generalisation for larger networks as the generated samples are no longer well dispersed around the true support, which is illustrated in sub-figure 1 of Figure 5.12. This quantifies why early stopping is an effective regularisation technique. We observe smaller models do not have the capacity to overfit to the training score and therefore the quality of generated samples does not worsen as the training process continues. After an initial training period, the radii generalisation remains low for models of all sizes. This is true of larger networks as in replicating the training set they generate samples lying exactly on the true data support. Smaller networks produce samples with slightly worse radii generalisation, this is because the generated samples are noiser.

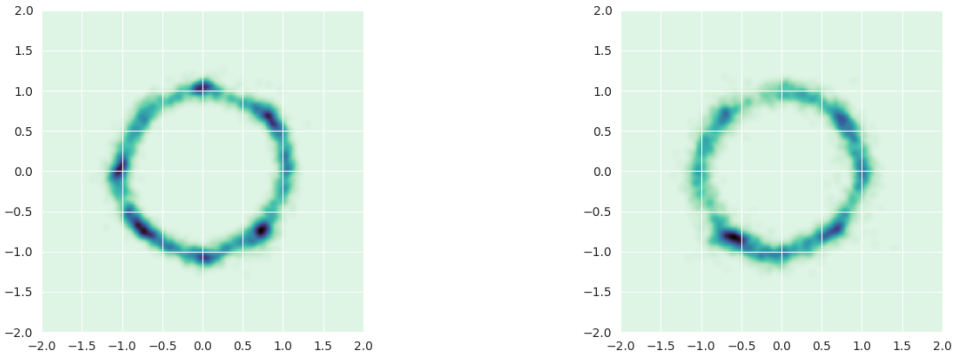

Figure 5.13: This figure illustrates the generated samples measured best under the true generalisation metric. These correspond to $9,000$ training epochs on a network architecture of $5L64N$ on the left-hand side and $17,000$ training epochs on the smallest network, $3L16N$, on the right-hand side. The training set is more identifiable in the greater sized network, with more generated samples concentrated around those points.

These results suggest smaller networks are less susceptible to overfitting to the training score and exhibit better generalisation throughout the training process. As it requires sample generation to determine when to perform early stopping, it is practically much easier to extensively train a smaller network, without concern of significant overfitting.

5.5 Generalisation Capabilities Of Score-Based Generative Models Under The Union Of Manifolds Hypothesis

While we have previously demonstrated the ability for SGMs to competently generalise to the underlying support of a dataset, there is currently no discussion in the literature as to whether this applies to data conforming to the union of manifolds hypothesis, presented in Section 3.1. In this section, we aim to use the theory and results derived in this report, to justify novel explanations of how accurately score-based generative models generalise on data distributions lying on disjoint supports. We support our argument with experiments comparing score-based models to push-forward models in generating samples from such datasets.

5.5.1 Theoretical Justifications For SGMs Learning Datasets With Disconnected Supports

The main argument in this section is presented in the following conjecture.

Conjecture 5.3 (Score-Based Generative Modelling on Datasets under the Union Of Manifolds Hypothesis)**.** *Under an Ornstein-Ulhenbeck forward perturbation process specified by* $dX_t = \beta X_t dt + \sqrt{2\beta} dW_t$, $t \in [0,1]$ *and assuming our standard assumptions from Conjecture 5.2 we can bound the total variation between the data distribution with disjoint latent support and the backward process in the following sense:*

$$d_{TV}(p_0, q_1) \lesssim \mathcal{O}(\epsilon) + \epsilon_{score}.$$

Proof. This argument comes from our previous analysis that for distributions with a finite second moment, we can find a β arbitrarily large to reduce the forward convergence error to $\mathcal{O}(\epsilon)$. For such a chosen β we can appropriately scale the step-size, as discussed in Section 5.2, to ensure a discretisation error of the same order. □

Of course, we can experimentally reduce this bound further by applying regularisation techniques to minimise the score approximation error, as discussed in Section 5.4.

We note this conjecture is consistent with general results in literature which state SGMs are able to learn datasets from distributions with bounded support [40, Assumption 1]. Although, as bounded supports imply bounded moments, the assumption the dataset has a finite second moment, which we use, is weaker.

5.5.2 Experiments of SGMs Learning Datasets With Disconnected Supports

Following this conjecture, we now aim to demonstrate experimentally that score-based generative models can capably generate samples lying on a union of disconnected supports and compare these results to push-forward models in a similar set up.

Experiment: Score-Based Generative Modelling On A Simple Union Of Disconnected Supports

In this experiment, we use a score-based generative model to generate samples from the union of circles dataset presented in Section 4.2.4. As a reminder, this dataset is comprised of two separate, disconnected, unit circles centered around $(-2, 0)$ and $(2, 0)$ in $\mathbb{R}^2$. We used the standard score-network architecture documented in the Appendix B.1 and implementation previously discussed in Section 2.3 with an appropriately chosen β to ensure convergence and adequate discretisation error. We present the results in Figure 5.14.

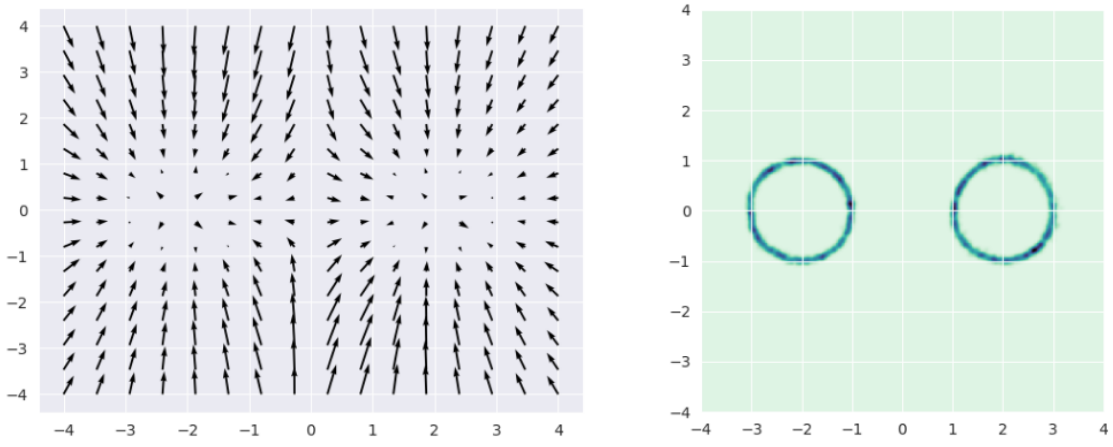

Figure 5.14: On the left-hand side is the learned score-network, at 3,000 training epochs, evaluated at $t = 0.01$. The right-hand side corresponds to a heatmap of 2,500 generated samples using the score-network in the discretised reverse process.

As this dataset is a union of measures with finite support, it has finite moments. Therefore, this experimental result supports our claim in Conjecture 5.3 as it appears the SGM algorithm has appropriately learned the true data score and can reproduce samples lying around the true disjoint support.

Demonstration Of VAEs and GANs failing to learn a simple lying on union of disconnected supports

In the related work section, we recalled Theorem 3.1 which provided theoretical justifications as to why push-forward models cannot appropriately generate samples lying exclusively on the disjoint support components. For means of comparison we demonstrate this experimentally too.

Below we present the results of training a variational auto-encoder and generative-adversarial network to learn the same union of circles dataset. Details about the network architectures and training methods of the push-foward models can be found in the Appendix, GAN: B.2, VAE: B.3. Importantly to ensure consistency, the generator and decoder networks are of identical size. This ensures they have the same capacity to generate samples. The score network is marginally bigger, although of a similar order of learn-able parameters, however this is excusable as each push-forward model requires an additional network to aid the training process.

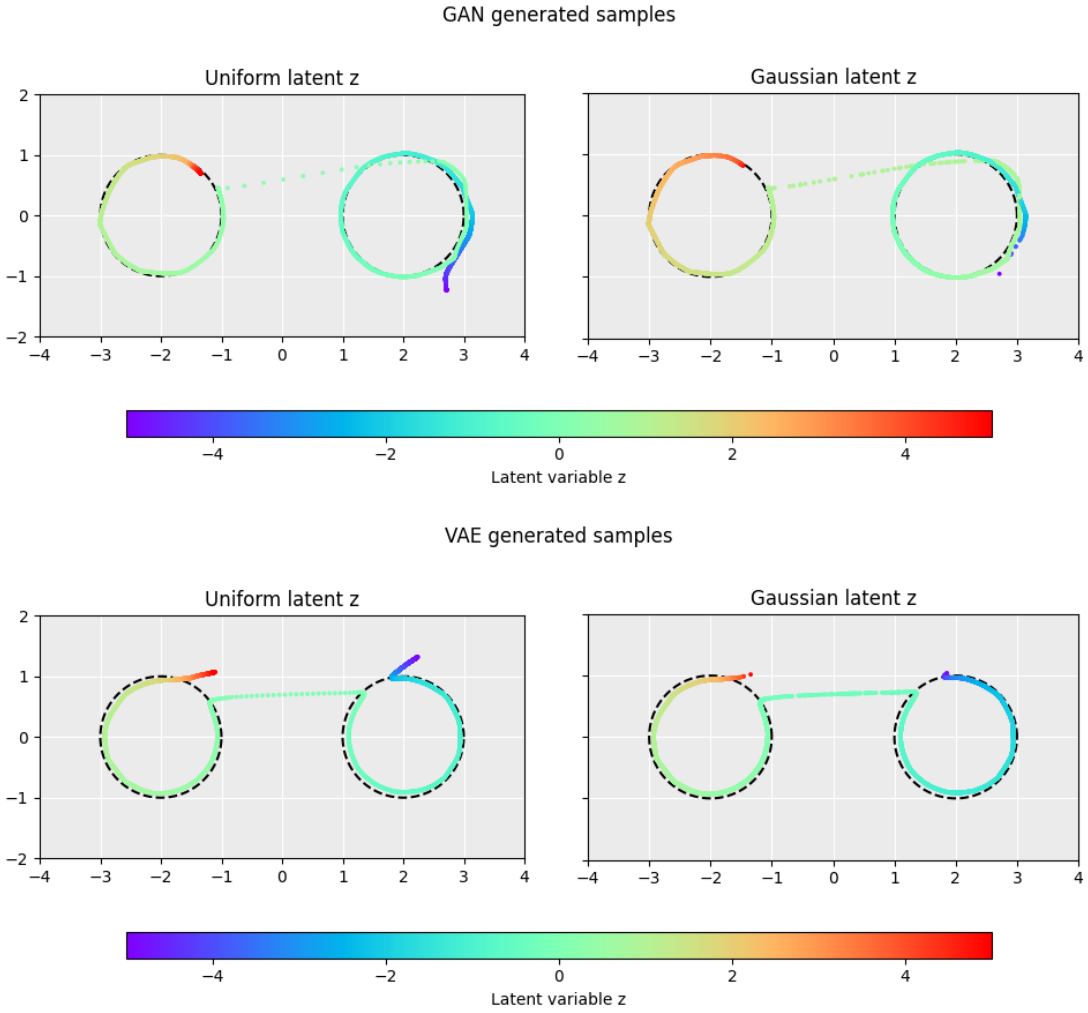

Figure 5.15: On the top row we see samples produced by the generator network after 960 training epochs. On the second row we observe the samples produced by the decoder network after 460 training epochs. On the left-hand side we see the distribution of transformed samples from $z \sim U[-5, 5]$ on the right-hand side we see the distribution of transformed samples from $z \sim N(0, 1)$. In all images the black dotted lines represent the true target support.

In Figure 5.15, we clearly observe a region between the two components which is supported in the push-forward measure by both models. This aligns with Theorem 3.1 as it demonstrates that although the probability density is relatively small in the region between the two manifolds, it is non-zero. This flaw is more detrimental in the Gaussian latent variable case as transformed samples which lie off the support are those from the latent space with greatest probability density.

Comparison Between Push-Forward Models And Score-Based Generative Models In Generating Samples Lying On A Disjoint Support

The experimental results indicate that score-based generative models succeed where push-forward models fail in generating samples with disjoint latent support. Furthermore, providing we have an accurate score-approximation, under Conjecture 5.3 (and associated assumptions) we can produce samples arbitrarily close to the data distribution with respect to the total variation. Push-forward models could potentially also achieve this by assigning arbitrarily small, but importantly non-negative, density to the region connecting the two disconnected supports. Indeed, due to the Brownian increments in our reverse discretisation process the entire state space is in the support of the generated samples under the backward process. However from our experiments it appears score-based models better assign density on the support.

We also remark that it would be natural to theorise the generalisation property of score-based models, particularly in the circle example from Section 3.2, is due to linear interpolation and not actually learning the latent support. However, the lack of any explicit learned support between the two circles, under the SGM algorithm, would dispute this.

Experiment: Score-Based Generative Modelling On More Complicated Disjoint Supports

Finally we present the results of sample generation on a couple more example datasets, to demonstrate these results hold more generally too. The union of manifolds hypothesis states the latent supports can be of varying dimensionality, therefore we first consider a disjoint support with varying latent dimension.

We consider the target support illustrated in Figure 5.16, notably as the support is bounded we have the required finite moment assumption. Using our standard score-network architecture and implementation we see the score-based model again generates appropriate samples without evidently interpolating between the disjoint support components, which is seen in Figure 5.17.

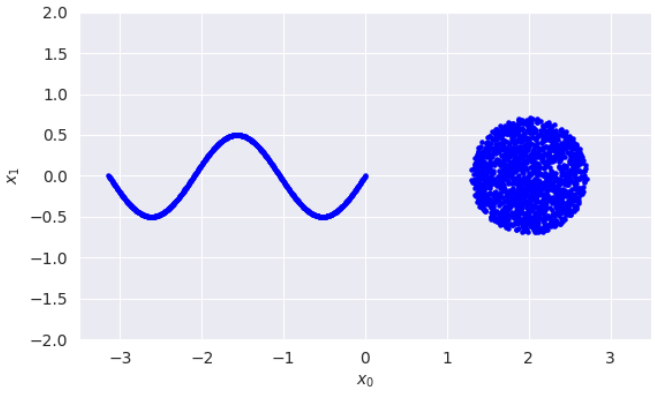

Figure 5.16: The left-hand support is sinusoidal and evidently has a one dimensional representation. The right-hand support is a filled circle, which is two dimensional.

Finally, we consider a higher dimensional example, a gapped Swiss roll dataset and a unit disc embedded in three dimensional space. The training set and generated samples are illustrated in Figure 5.18.

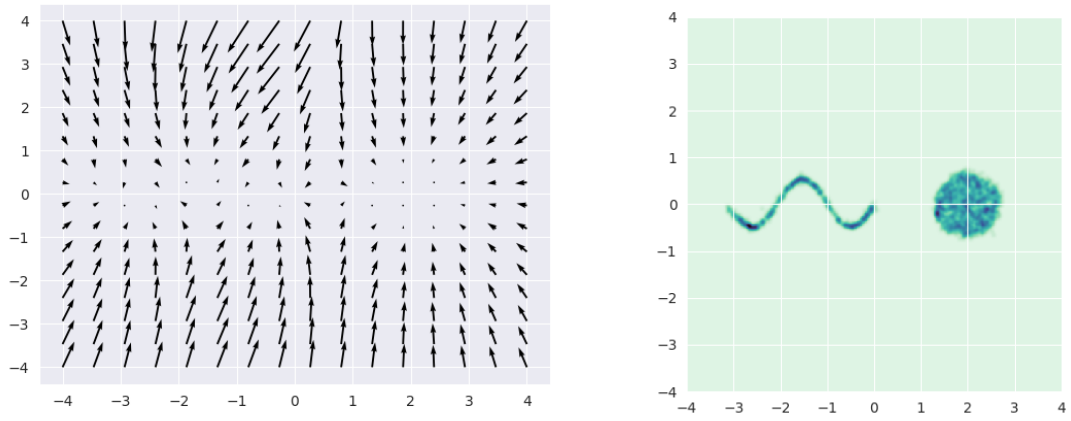

Figure 5.17: On the left-hand side is the learned score-network, at $5,000$ training epochs, evaluated at $t = 0.01$. The right-hand side corresponds to a heatmap of generated samples using the score-network in the discretised reverse process.

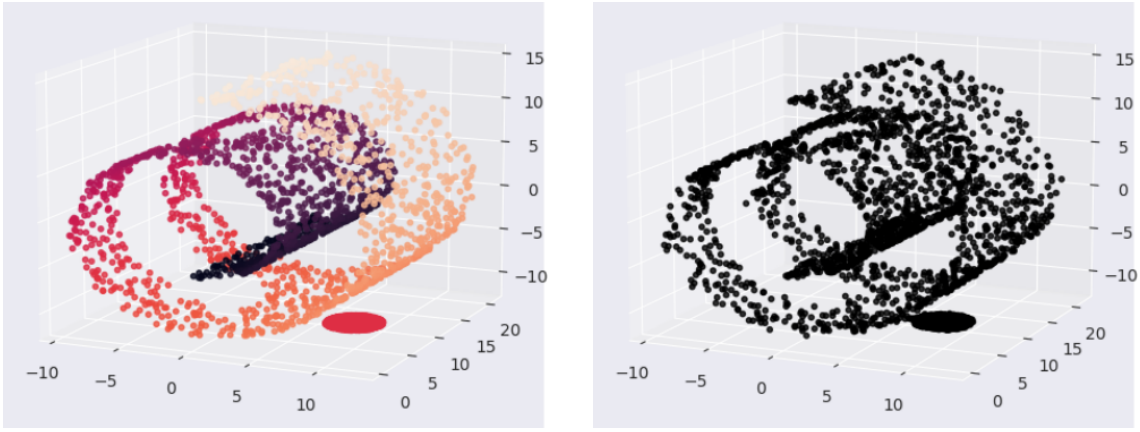

Figure 5.18: On the left-hand side is the training set embedded in three dimensions. The right-hand side illustrates the generated samples using score-network at $5,000$ training epochs, in the discretised reverse process.

Chapter 6

Evaluation

The implications of our results are discussed as they are demonstrated and derived in the previous section. In this section we discuss our findings in relation to the wider literature.

Much of the theoretical work in this project aims to generalise Theorem 3.2 in terms of arbitrary time-homogeneous Ornstein-Ulhenbeck processes, over a fixed time interval.

The exponential convergence of the forward perturbation process in terms of β is by no means a novel result. Indeed, Ornstein-Ulhenbeck processes are one of the most well studied stochastic differential equations. However, we note the paper which formulates Theorem 3.2 does not refer to these results and instead uses a far more general argument to justify the exponential convergence. Therefore, presenting this result in the context of score-based generative models is a meaningful contribution in of itself - as we saw in our experiments, understanding the relationship between β and the diameter of the dataset is essential in minimising the error of the backward process and in ensuring high quality samples are generated.

We also consider the Cauchy distribution to demonstrate experimentally that our assumption the data distribution has a finite second moment is too strong and we can still obtain convergence in the forward process in cases without this assumption. We believe this is the best justification we can feasibly obtain as to the best of our knowledge there is no result of in the literature which specifies the necessary conditions for exponentially fast convergence under Ornstein-Ulhenbeck processes.

We present the relationship between the rate of convergence β and the discretisation error in approximating the continuous backward process. While this result was not presented to the same level of rigour as in Theorem 3.2 from the literature, we believe our sketch of the proof is convincing and the supporting experiments adequately justify the relation in Conjecture 5.1.

Combining this result with the data-processing inequality presented in (3.2) provides new insight of a trade-off in determining appropriate perturbation parameters which guarantee a low bound in the total variation between the backward process and target support. We demonstrate this result holds on a variety of proxy datasets conforming to the manifold hypothesis and on a popular synthetic dataset from the literature.

Following these derivations we shifted our analysis to a more practical consideration, in minimising the error associated with approximating the true score function of the law of the forward process. Our reasoning indicates there is an intrinsic relation between the complexity of the target support and capacity of the score-network. If the network is appropriately sized we mitigate the score-approximation error induced by overfitting to the score of the training set and consequently produce high quality samples. Unfortunately, while we demonstrate this relation exists we provide no insight into how the optimal size should be determined. We also remark that quantifying the complexity of a dataset in terms of the number of learn-able parameters in a neural network is a very vaguely constructed problem statement and requires formalising in its own right before being researched further.

We combine the above results to motivate a number of theoretical comparisons to another class of state-of-the-art generative models, namely push-forward models. We firstly consider generating samples from one dimensional Gaussian mixture distributions. Under Conjecture 5.2 we argue the free parameters of the perturbation process ensure, under good score function approximation and suitable discretisation increments, we can make the bound of total variation in the backward process arbitrarily small. This contrasts the corresponding result for push-forward models, as

the Lipschitz constant of the push-forward network must scale accordingly with the dispersion of mixture component means. We argue SGMs perform better due to the logarithmic increase in discretisation error induced by dispersing mixture component means further from the stationary mean α.

Finally, we apply our results to argue novel justifications as to why score-based generative models can adequately learn distributions lying on disjoint latent supports. We demonstrate these capabilities on a variety of synthetic examples in accordance with the union of manifolds hypothesis. Push-forward models cannot generate samples without interpolating between the disjoint supports. Despite samples generated by the SGM algorithm having full support in $\mathbb{R}^n$, by considering our upper-bound on the total variation between the target support and generated samples we hypothesise we can make the density off the target disjoint support arbitrarily small. As a result, we argue SGMs outperform push-forward models in this setting.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In conclusion, we have generalised Theorem 3.3 to arbitrary Ornstein-Ulhenbeck processes in the practical restriction of a finite time perturbation processes. Considering general initial data distributions, a fixed time interval is preferable to running the forward perturbation arbitrarily forward in time and discretising this variable, dataset dependent, range. In this context, the trade-off in specifying the perturbation parameters, quantified in terms of the forward convergence and discretisation error of the backward process, helps justify the broad empirical capabilities of score-based generative models. As we have considered synthetic datasets which are suitable abstractions of many real-world datasets, our results help explain why SGMs are able achieve state-of-the-art performance over other generalisation model architectures.

7.2 Future Work

While we are confident in the relationship presented in Conjecture 5.1, and consequently Conjecture 5.2, our proof is not as rigorous as the one presented for Theorem 3.3. Therefore, we have refrained from presenting it as a theorem in the same way. With additional time, it would be possible to formalise this result to the same standard and make stronger assertions about our findings. Additionally, should we wish to develop this research to the standard of publication we would need to consider more conventional research datasets, such as MNIST of CIFAR-10.

Additionally, we restricted our analysis wholly to time-homogeneous Ornstein-Ulhenbeck processes, which means the parameters are constant with respect to time. It has been shown empirically that varying these parameters throughout the perturbation process improves the quality of generated samples along the target support [41]. Generalising Theorem 3.3 and Conjectures 5.1 and 5.2 to theoretically quantify this improved performance is therefore a natural extension to the work in this project.

Early on in the project, we investigated particle interacting perturbation processes. Briefly, these processes introduce an attractive interaction term between particles in the forward process. The intuition is that under the reverse process this reverse attraction would repel particles apart and could speed up backward convergence or better disperse particles around the target support. In considering these systems we derived the exponential convergence of the forward process in terms of the drift β and largest eigenvalue of the interaction matrix. As the convergence was also exponentially fast, we can derive similar convergence results as presented in this report for these systems too.

7.3 Ethical Considerations

Due to the theoretical nature of this project and synthetic experimental datasets, the ethical considerations in this project are minimal. However, we will discuss potential ethical issues in the wider field of generative modeling - which is contextually relevant to this project.

Score-based generative models are often used in image generation, specifically faces. As the score-function is trained on the input training samples its important this data is sourced ethically.

More generally, under GDPR[1] we require personal data to be anonymised. Furthermore, as faces are specially classed as 'biometric data' there are further complications in ensuring compliance. In the wider scope of generative modelling, many professionals are worried how advances in generative modeling threatens their job security.

In relation to this work specifically, as we have purely focused on generalisation capabilities, these issues are not directly relevant to this project. However, it is important to bear these implications in mind, especially if we aim to further our results in future on more complicated datasets, such as images.

[1]Following Brexit, the UK wrote an identical form of GDPR into national law.

Bibliography

[1] J.A. Sethian. Noise removal from images. URL https://math.berkeley.edu/~sethian/2006/Applications/ImageProcessing/noiseremoval.html. Last accessed: 11 Janurary 2023.

[2] Suvrat Bhooshan Prasad Kawthekar, Raunaq Rewari. Evaluating generative models for text generation. URL https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/reports/2737434.pdf. Last accessed: 21 December 2022.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[6] Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013. Last accessed: 20 January 2023.

[7] Grigorios A Pavliotis. *Stochastic processes and applications: diffusion processes, the Fokker-Planck and Langevin equations*, volume 60. Springer, 2014.

[8] Rabi N Bhattacharya and Edward C Waymire. *Stochastic processes with applications*. SIAM, 2009.

[9] Steven E Shreve et al. *Stochastic calculus for finance II: Continuous-time models*, volume 11. Springer, 2004.

[10] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982. Last accessed: 20 January 2023.

[11] Ulrich G Haussmann and Etienne Pardoux. Time reversal of diffusions. *The Annals of Probability*, pages 1188–1205, 1986.

[12] David J.C. MacKay. Information theory, inference, and learning algorithms. URL https://www.inference.org.uk/itprnn/book.pdf. Page 384, Lasted accessed: 22 December 2022.

[13] Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021.

[14] Yee Whye Teh, Max Welling, Simon Osindero, and Geoffrey E Hinton. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, 4(Dec):1235–1260, 2003.

[15] James Martens, Ilya Sutskever, and Kevin Swersky. Estimating the hessian by back-propagating curvature. *arXiv preprint arXiv:1206.6464*, 2012.

[16] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. pages 574–584, 2020.

[17] Alain Durmus and Eric Moulines. High-dimensional bayesian inference via the unadjusted langevin algorithm. 2019.

[18] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[19] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

[20] Yang Song. Score-based generative modeling through stochastic differential equations. URL https://arxiv.org/pdf/2011.13456.pdf. Last accessed: 10 January 2023.

[21] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[22] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8): 1798–1828, 2013.

[23] Bradley CA Brown, Anthony L Caterini, Brendan Leigh Ross, Jesse C Cresswell, and Gabriel Loaiza-Ganem. The union of manifolds hypothesis, 2022.

[24] Benjamin Boys. Diffusion jax python package. URL https://pypi.org/project/diffusionjax/. Last accessed: 9th June 2023.

[25] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[26] Imre Csiszár and János Körner. *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press, 2011.

[27] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. Generalized sliced wasserstein distances. *Advances in neural information processing systems*, 32, 2019.

[28] Hoang Thanh-Tung and Truyen Tran. Toward a generalization metric for deep generative models. *arXiv preprint arXiv:2011.00754*, 2020. Last accessed: 20 January 2023.

[29] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51:22–45, 2015.

[30] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

[31] Kaggle. Generative adversarial networks (gans) tutorial. URL https://www.kaggle.com/code/emreustundag/generative-adversarial-networks-gans-tutorial. Last accessed: 9th June 2023.

[32] Antoine Salmona, Valentin de Bortoli, Julie Delon, and Agnès Desolneux. Can push-forward generative models fit multimodal distributions?, 2022.

[33] Mahyar Khayatkhoei, Maneesh K Singh, and Ahmed Elgammal. Disconnected manifold learning for generative adversarial networks. *Advances in Neural Information Processing Systems*, 31, 2018.

[34] Charles Chapman Pugh and CC Pugh. *Real mathematical analysis*, volume 2011. Springer, 2002.

[35] Bradley CA Brown, Anthony L Caterini, Brendan Leigh Ross, Jesse C Cresswell, and Gabriel Loaiza-Ganem. Verifying the union of manifolds hypothesis for image data. 2023.

[36] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru R. Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions, 2023.

[37] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total variation distance between high-dimensional gaussians with the same mean, 2022.

[38] Leonard Gross. Logarithmic sobolev inequalities. *American Journal of Mathematics*, 97(4): 1061–1083, 1975.

[39] Santosh Vempala and Andre Wibisono. Rapid convergence of the unadjusted langevin algorithm: Isoperimetry suffices. *Advances in neural information processing systems*, 32, 2019.

[40] Valentin De Bortoli. Convergence of denoising diffusion models under the manifold hypothesis. *arXiv preprint arXiv:2208.05314*, 2022.

[41] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022.

[42] Dominique Bakry and Michel Émery. Diffusions hypercontractives. pages 177–206, 2006.

Appendix A

Proofs

A.1 Total Variation On Mixture Distributions

Proof of lemma (5.1).

$$\begin{aligned} d_{TV}(p,q) &= \frac{1}{2}\int_X |p(x)-q(x)|dx \\ &= \frac{1}{2}\int_X |\lambda p_1(x)+(1-\lambda)p_2(x)-q(x)(\lambda+(1-\lambda))|dx \\ &\leq \frac{\lambda}{2}\int_X |p_1(x)-q(x)|dx + \frac{1-\lambda}{2}\int_X |p_2(x)-q(x)|dx \\ &= \lambda d_{TV}(p_1,q)+(1-\lambda)d_{TV}(p_2,q) \end{aligned}$$

□

A.2 Ornstein-Ulhenbeck process

A.2.1 Solution Of The Ornstein-Ulhenbeck Process

Proof. Let the driving Brownian motion, $\mathbf{W_t}$, be the standard Brownian motion in $\mathbb{R}^n$, $\boldsymbol{\alpha} \in \mathbb{R}^n$, $\beta > 0$, then:

$$d\mathbf{X_t} = -\beta(\mathbf{X_t}-\boldsymbol{\alpha})dt + \sigma d\mathbf{W_t}$$

$$\text{Let } \mathbf{Y_t} = \mathbf{X_t}-\boldsymbol{\alpha} \overset{(1)}{\implies} d\mathbf{Y_t} = -\beta(\mathbf{Y_t})dt + \sigma d\mathbf{W_t}$$

Let $\mathbf{Z_t} = \varphi(\mathbf{Y_t},t) = \left(\mathbf{Y_t}^{(1)}e^{\beta t}, \mathbf{Y_t}^{(2)}e^{\beta t}, \ldots, \mathbf{Y_t}^{(n)}e^{\beta t}\right) = \mathbf{Y_t}e^{\beta t}$ with $\mathbf{Y}_t^{(i)}$ denoting the i^{th} component at time t.

$$\begin{aligned} &\overset{(2)}{\implies} d\mathbf{Z_t} = \sigma e^{\beta t} d\mathbf{W_t} \\ &\implies \mathbf{Z_t} = \mathbf{Z_0} + \int_0^t \sigma e^{\beta s} d\mathbf{W_s} \\ &\implies \mathbf{Y_t}e^{\beta t} = \mathbf{Y_0}e^{\beta 0} + \int_0^t \sigma e^{\beta s} d\mathbf{W_s} \\ &\implies \mathbf{Y_t} = \mathbf{Y_0}e^{-\beta t} + \int_0^t \sigma e^{\beta(s-t)} d\mathbf{W_s} \\ &\implies \mathbf{X_t} = \boldsymbol{\alpha} + (\mathbf{X_0}-\boldsymbol{\alpha})e^{-\beta t} + \int_0^t \sigma e^{\beta(s-t)} d\mathbf{W_s}. \end{aligned}$$

Here $(1), (2)$ follow from applying Itô's lemma. We note that the Itô stochastic integral of a deterministic integrand is Gaussian distributed, so we can further derive the first two moments of

the integral and the law of the process.

$$\begin{aligned}
\mathbb{E}\left[\mathbf{Y_t}\right] &= \mathbb{E}\left[\int_0^t \sigma e^{\beta(s-t)} d\mathbf{W_s}\right] = \mathbf{0}_n \\
\mathbb{E}\left[\mathbf{Y_t} \otimes \mathbf{Y_t}\right]_{ij} &= \mathbb{E}\left[\left(\int_0^t \sigma e^{\beta(s-t)} d\mathbf{W_s}\right) \otimes \left(\int_0^t \sigma e^{\beta(s-t)} d\mathbf{W_s}\right)\right]_{ij} \\
&\overset{(3)}{=} \delta_{ij} \int_0^t \left(\sigma e^{\beta(s-t)}\right)^2 ds \\
&= \delta_{ij}\sigma^2 e^{-2\beta t} \int_0^t e^{2\beta s} ds \\
&= \delta_{ij}\sigma^2 e^{-2\beta t} \frac{1}{2\beta}(e^{2\beta t} - 1) = \delta_{ij}\frac{\sigma^2}{2\beta}(1 - e^{-2\beta t}) \\
&\implies \delta_{ij} \int_0^t \sigma e^{\beta(s-t)} d\mathbf{W_s} \sim N(\mathbf{0}_n, \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})\mathbf{I}_n).
\end{aligned}$$

Where $\delta_{ij} := \{0 \text{ if } i \neq j,\ 1 \text{ if } i = j\}$ and (3) follows from Itô's isometry. □

A.2.2 Solution To The Ornstein-Ulhenbeck Process With An Initial Gaussian Distribution

Corollary A.1 (Marginal distribution $p_t(x)$ when X_0 is Gaussian)**.** *If X_0 is Gaussian, then under a OU perturbation the marginals p_t are also Gaussian and can be derived analytically. This follows as the sum of two independent multivariate normal distributions is given by $\mathcal{N}(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1}) + \mathcal{N}(\boldsymbol{\mu_2}, \boldsymbol{\Sigma_2}) = \mathcal{N}(\boldsymbol{\mu_1} + \boldsymbol{\mu_2}, \boldsymbol{\Sigma_1} + \boldsymbol{\Sigma_2})$. Let $\mathbf{X_0} \sim \mathcal{N}(\boldsymbol{\mu_0}, \boldsymbol{\Sigma_0})$.*

$$\begin{aligned}
\mathbf{X_t} &= \boldsymbol{\alpha} + (\mathbf{X_0} - \boldsymbol{\alpha})e^{-\beta t} + \int_0^t \sigma e^{\beta(s-t)} d\mathbf{W_s} \\
&\sim \boldsymbol{\alpha} + (\mathcal{N}(\boldsymbol{\mu_0}, \boldsymbol{\Sigma_0}) - \boldsymbol{\alpha})e^{-\beta t} + \mathcal{N}(\mathbf{0}, \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})\mathbf{I}_n) \\
&\overset{d}{=} \mathcal{N}\left(\boldsymbol{\alpha} + (\boldsymbol{\mu_0} - \boldsymbol{\alpha})e^{-\beta t}, \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})\mathbf{I}_n + \boldsymbol{\Sigma_0} e^{-2\beta t}\right) \to \mathcal{N}\left(\boldsymbol{\alpha}, \frac{\sigma^2}{2\beta}\mathbf{I}_n\right) \text{ as } t \to \infty.
\end{aligned}$$

Furthermore, by considering the forward and backward Kolmogorov equations associated with this stochastic differential equation it follows that this is the unique stationary distribution of the OU process [7, Chapter 4].

A.3 Total variation forward process with an initial Gaussian distribution and stationary process, under OU perturbation

Proof. Total variation in the forward process between the invariant distribution and perturbed process for Multivariate Gaussian initial distributions. Let $p_0 = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ then from the above derivation we have $p_t = \mathcal{N}\left(\boldsymbol{\alpha} + (\boldsymbol{\mu} - \boldsymbol{\alpha})e^{-\beta t}, \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})\mathbf{I}_n + \boldsymbol{\Sigma} e^{-2\beta t}\right)$. Using (5.3) we find:

$$\begin{aligned}
KL(q, p_t) &= KL\left(\mathcal{N}\left(\alpha, \frac{\sigma^2}{2\beta}\boldsymbol{I}_n\right) \ || \ \mathcal{N}\left(\boldsymbol{\alpha} + (\boldsymbol{\mu} - \boldsymbol{\alpha})e^{-\beta t}, \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})\mathbf{I}_n + \boldsymbol{\Sigma} e^{-2\beta t}\right)\right) \\
&\leq \frac{1}{2} trace(\boldsymbol{\Sigma_1}^{-1}\boldsymbol{\Sigma_2} - \boldsymbol{I}_n) + (\boldsymbol{\mu_1} - \boldsymbol{\mu_2})^T \boldsymbol{\Sigma_1}^{-1}(\boldsymbol{\mu_1} - \boldsymbol{\mu_2}) - \log\det(\boldsymbol{\Sigma_2}\boldsymbol{\Sigma_1}^{-1}).
\end{aligned}$$

In our case we have:

$$\boldsymbol{\Sigma_1}^{-1}\boldsymbol{\Sigma_2} - \boldsymbol{I}_n = \left(\frac{2\beta}{\sigma^2}\boldsymbol{I}_n\right)\left(\frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})\boldsymbol{I}_n + \boldsymbol{\Sigma} e^{-2\beta t}\right) - \boldsymbol{I}_n = e^{-2\beta t}(\boldsymbol{\Sigma} - \boldsymbol{I}_n).$$

$$(\boldsymbol{\mu_1} - \boldsymbol{\mu_2})^T \boldsymbol{\Sigma_1}^{-1}(\boldsymbol{\mu_1} - \boldsymbol{\mu_2}) = (\boldsymbol{\mu} - \boldsymbol{\alpha})^T e^{-\beta t}\frac{\sigma^2}{2\beta}\boldsymbol{I}_n(\boldsymbol{\mu} - \boldsymbol{\alpha})e^{-\beta t} = ||\boldsymbol{\mu} - \boldsymbol{\alpha}||_2^2 \frac{\sigma^2}{2\beta}e^{-2\beta t}.$$

In considering the final term, $\det(\mathbf{\Sigma_2}{\mathbf{\Sigma_1}}^{-1})$, we find:

$$\begin{aligned}
\det(\mathbf{\Sigma_2}{\mathbf{\Sigma_1}}^{-1}) &= \det(\frac{2\beta}{\sigma^2}\boldsymbol{I}_n\left(\frac{\sigma^2}{2\beta}(1-e^{-2\beta t})\boldsymbol{I}_n + \mathbf{\Sigma}e^{-2\beta t}\right)) \\
&= \det(\boldsymbol{I}_n)\det\left(\frac{2\beta}{\sigma^2}\boldsymbol{I}_n\left(\frac{\sigma^2}{2\beta}(1-e^{-2\beta t})\boldsymbol{I}_n + \mathbf{\Sigma}e^{-2\beta t}\right)\right) && (\det(AB)=\det(A)\det(B)) \\
&= \det\left(\frac{2\beta}{\sigma^2}\boldsymbol{I}_n\left(\frac{\sigma^2}{2\beta}(1-e^{-2\beta t})\boldsymbol{I}_n + \frac{2\beta}{\sigma^2}\mathbf{\Sigma}e^{-2\beta t}\right)\right) && (\det(\boldsymbol{I}_n)=1) \\
&= \det((1-e^{-2\beta t})\boldsymbol{I}_n + \frac{2\beta}{\sigma^2}\mathbf{\Sigma}e^{-2\beta t}) \\
&= \det(\boldsymbol{I}_n + e^{-2\beta t}(\frac{2\beta}{\sigma^2}\mathbf{\Sigma} - \boldsymbol{I}_n)) \\
&= 1 + \sum\det\left(e^{-2\beta t}(\frac{2\beta}{\sigma^2}\mathbf{\Sigma} - \boldsymbol{I}_n)_I\right) && (\det(I_n+A)=1+\sum\det(A_I))
\end{aligned}$$

Where A_I denotes the principle sub-matrices of A, this is equivalently the sum of the principle minors of A. As $\mathbf{\Sigma}$ is the initial co-variance matrix it is positive semi-definite, therefore for β sufficiently large each term in the sum is positive. Hence, $\det(\mathbf{\Sigma_2}{\mathbf{\Sigma_1}}^{-1}) \geq 1 \implies -\log\det(\mathbf{\Sigma_2}{\mathbf{\Sigma_1}}^{-1}) \leq 0$ and arrive at the following:

$$\begin{aligned}
KL(q||p_t) &\leq \frac{1}{2}trace({\mathbf{\Sigma_1}}^{-1}\mathbf{\Sigma_2} - \boldsymbol{I}_n) + (\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T{\mathbf{\Sigma_1}}^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2}) - \log\det(\mathbf{\Sigma_2}{\mathbf{\Sigma_1}}^{-1}) \\
&\leq \frac{1}{2}trace({\mathbf{\Sigma_1}}^{-1}\mathbf{\Sigma_2} - \boldsymbol{I}_n) + (\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T{\mathbf{\Sigma_1}}^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2}) \\
&= \frac{1}{2}e^{-2\beta t}\left(trace(\mathbf{\Sigma} - \boldsymbol{I}_n) + ||\boldsymbol{\mu}-\boldsymbol{\alpha}||_2^2\frac{\sigma^2}{2\beta}\right).
\end{aligned}$$

Now by applying Pinkser's inequality (**??**) with (5.4) we establish:

$$\begin{aligned}
d_{TV}(q,p_t) &\leq \frac{1}{2}\left(trace({\mathbf{\Sigma_1}}^{-1}\mathbf{\Sigma_2} - \boldsymbol{I}_n) + (\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T{\mathbf{\Sigma_1}}^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})\right)^{\frac{1}{2}} \\
&= \frac{1}{2}e^{-\beta t}\left(trace(\mathbf{\Sigma} - \boldsymbol{I}_n) + ||\boldsymbol{\mu}-\boldsymbol{\alpha}||_2^2\frac{\sigma^2}{2\beta}\right)^{\frac{1}{2}}.
\end{aligned}$$

Which leads us to the result of exponentially fast convergence of the forward process with respect to the total variation. □

A.3.1 Total Variation In The Forward Process Of A Multi-Dimensional Gaussian Mixture Distribution

Proof. Considering now a multivariate Gaussian mixture with two components, we can define the initial distribution as follows, $p_0 = p_0^{(1)} + p_0^{(2)} = \lambda\mathcal{N}(\boldsymbol{\mu_1},\mathbf{\Sigma_1}) + (1-\lambda)\mathcal{N}(\boldsymbol{\mu_2},\mathbf{\Sigma_2})$ and $q = \mathcal{N}\left(\boldsymbol{\alpha}, \frac{\sigma^2}{2\beta}\mathbf{I}_n\right)$ the invariant stationary distribution of the OU process. Then we have, $p_t = \lambda p_t^{(1)} + (1-\lambda)p_t^{(2)} = \lambda\mathcal{N}\left(\boldsymbol{\alpha} + (\boldsymbol{\mu_1}-\boldsymbol{\alpha})e^{-\beta t}, \frac{\sigma^2}{2\beta}(1-e^{-2\beta t})\mathbf{I}_n + \mathbf{\Sigma_1}e^{-2\beta t}\right) + (1-\lambda)\mathcal{N}\left(\boldsymbol{\alpha} + (\boldsymbol{\mu_2}-\boldsymbol{\alpha})e^{-\beta t}, \frac{\sigma^2}{2\beta}(1-e^{-2\beta t})\mathbf{I}_n + \mathbf{\Sigma_2}e^{-2\beta t}\right)$. Considering now $d_{TV}(q,p_t)$ and applying lemma (5.1) and (A.3), we get:

$$\begin{aligned}
&d_{TV}(q,p_t) \leq \lambda d_{TV}(q,p_t^{(1)}) + (1-\lambda)d_{TV}(q,p_t^{(2)}) \\
&\leq e^{-\beta t}\left[\frac{\lambda}{2}\left(trace(\mathbf{\Sigma}_1 - \boldsymbol{I}_n) + ||\boldsymbol{\mu}_1-\boldsymbol{\alpha}||_2^2\frac{\sigma^2}{2\beta}\right)^{\frac{1}{2}} + (1-\lambda)\left(trace(\mathbf{\Sigma}_2 - \boldsymbol{I}_n) + ||\boldsymbol{\mu}_2-\boldsymbol{\alpha}||_2^2\frac{\sigma^2}{2\beta}\right)^{\frac{1}{2}}\right].
\end{aligned}$$

Which we note tends to 0 as $t \to \infty$. □

A.4 Forward Convergence For Arbitrary Data Distributions

Definition A.1 (C-strongly convex)**.** *A function $f : \mathbb{R}^n \to \mathbb{R}$ is C-strongly convex if the smallest eigenvalue of $\nabla_x^2 f(x)$ (the hessian of f) $\geq C$ $\forall x \in \mathbb{R}_n$.*

Lemma A.1 (Relation between strong convexity and the logarthmic-Sobolev inequality [42]). *For a probability measure p defined on $\mathbb{R}^n$, if $f(x) := -\log p(x)$ is C-strongly convex then p satisfies the log-Sobolev inequality with constant C.*

A.4.1 Logarithmic-Sobolev Inequality For The Invariant Distribution Of The Ornstein-Ulhenbeck Process

Proof. From Appendix (A.2.1) we have under the Ornstein-Ulhenbeck perturbation process $X_\infty \sim \mathcal{N}(\boldsymbol{\alpha}, \frac{\sigma^2}{2\beta}\boldsymbol{I})_n) \implies p_\infty(x) = \sqrt{\frac{\beta}{\pi\sigma}}\exp(-\frac{\beta}{\sigma^2}||\boldsymbol{x}-\boldsymbol{\alpha}||_2^2)$. I will derive the LSI constant for this process by showing this density is strongly convex.

$$
\begin{aligned}
p_\infty(x) &= \sqrt{\frac{\beta}{\pi\sigma}}\exp(-\frac{\beta}{\sigma^2}||\boldsymbol{x}-\boldsymbol{\alpha}||_2^2) \\
f(x) := -\log p(x) &= \frac{1}{2}\log\frac{\pi\sigma^2}{\beta} + \frac{\beta}{\sigma^2}||\boldsymbol{x}-\boldsymbol{\alpha}||_2^2 \\
\implies \nabla f(x) &= \frac{2\beta}{\sigma^2}\left(\boldsymbol{x}^{(1)}-\boldsymbol{\alpha}^{(1)},\ \dots\ ,\ \boldsymbol{x}^{(n)}-\boldsymbol{\alpha}^{(n)}\right) \implies \nabla^2 f(x) = \frac{2\beta}{\sigma^2}\boldsymbol{I}_n.
\end{aligned}
$$

Therefore, we have $p_\infty(x)$ is $\frac{2\beta}{\sigma^2}$-strongly convex. By applying Lemma A.1 it follows that p_∞ satisfies the logarithmic-Sobolev inequality with constant $\frac{2\beta}{\sigma^2}$. □

A.5 Discretisation Error Of The Backward Process For A General Ornstein-Ulhenbeck Perturbation Process

Under the same assumptions outlined in Theorem (3.2), the total variation between the reverse process and score-based generative modelling algorithm is bounded above by the following quantity,

$$d_{TV}(Q_T, Q_t^{p_T})^2 \leq KL(Q_T||Q_t^{p_T}) \lesssim (L^2dh\beta + L^2m_2^2h^2\beta^2 + \epsilon_{\text{score}}^2)T.$$

This proof is a sketch, which is largely based on the discretisation of de-noising diffusion probabilistic models presented by Chen Et al. [36, Theorem 9]. In this proof, $S(X, t; \hat{\theta})$ denotes our training score network, which is assumed to be L^2-accurate.

Proof. In a similar manner, we start by proving

$$\sum_{k=0}^{N-1}\mathbb{E}_{Q_T}\int_{j\Delta t}^{(j+1)\Delta t}||S(X_{j\Delta t}, T-j\Delta t; \hat{\theta}) - \nabla_x \log p_{T-t}(X_t)||^2 \lesssim (L^2dh\beta + L^2m_2^2h^2\beta^2 + \epsilon_{\text{score}}^2)T.$$

Once this is established, we can apply the 'approximation argument' [36, p.g. 13] in a similar manner to derive the result.

For $t \in [jh, (j+1)h]$,

$$
\begin{aligned}
&\mathbb{E}_{Q_T}[||S(X_{j\Delta t}, T-j\Delta t; \hat{\theta}) - \nabla_x \log p_{T-t}(X_t)||^2] \lesssim \\
&\quad \mathbb{E}_{Q_T}[||S(X_{j\Delta t}, T-j\Delta t; \hat{\theta}) - \nabla_x \log p_{T-kh}(X_{kh})||^2] + && \text{(score function error)} \\
&\quad \mathbb{E}_{Q_T}[||\nabla_x \log p_{T-kh}(X_{kh}) - \nabla_x \log p_{T-t}(X_t)||^2] + && \text{(discretisation error in score)} \\
&\quad \mathbb{E}_{Q_T}[||\nabla_x \log p_{T-t}(X_t) - \nabla_x \log p_{T-t}(X_t)||^2] + && \text{(discretisation error in } X_t\text{)} \\
&\quad \lesssim \epsilon_{\text{score}}^2 + \mathbb{E}_{Q_T}\left[||\log\frac{p_{T-kh}(X_{kh})}{p_{T-t}(X_{kh})}\right] + L^2\mathbb{E}_{Q_T}[||X_{kh}-X_t||^2].
\end{aligned}
$$

The middle term corresponds to the change in score in the forward process. In a similar manner to Chen we define the mapping $S : \mathbb{R}^n \to \mathbb{R}^n,\ S(x) := \exp(-\beta(t-kh))x$, then $p_{T-kh} = S_\# p_{T-t} * \mathcal{N}(0, 1-\exp(-2\beta(t-kh)))$. We then apply Lemma 16 from Chen [36, Lemma 16], with $\alpha = \exp(\beta(t-kh)) = 1+\beta\mathcal{O}(h)$ and $\sigma^2 = 1-\exp(-2\beta(t-kh)) = \beta\mathcal{O}(h)$,

$$
\begin{aligned}
||\nabla_x \log\frac{p_{T-kh}(X_{kh})}{p_{T-t}(X_{kh})}|| &\lesssim L^2\beta dh + L^2h^2\beta^2||X_{kh}||^2 + (1+L^2)\beta^2h^2||\nabla_x p_{T-t}(X_{kh})||^2 \\
&\lesssim L^2\beta dh + L^2h^2\beta^2||X_{kh}||^2 + L^2\beta^2h^2||\nabla_x p_{T-t}(X_{kh})||^2 \qquad \text{(as } L \geq 1\text{)}
\end{aligned}
$$

Finally, as Q_T is the time reversal process of the forward process given by marginals p_T, we can apply the moment bounds from Lemma 10 [36, Lemma 10] and moment bound from Lemma 11 [36, Lemma 11], and so,

$$\mathbb{E}_{Q_T}[||S(X_{j\Delta t}, T - j\Delta t; \hat{\theta}) - \nabla_x \log p_{T-t}(X_t)||^2] \lesssim \epsilon^2_{\text{score}} + L^2\beta dh + L^2h^2\beta^2.$$

As previously referred to, this result can be used in the approximation argument presented by Chen Et al. [36] to arrive at the final result. □

Appendix B

Neural Network Architectures and Training

In this section we describe the various architectures and training procedures of the neural networks used in this project.

B.1 Score-Network

Our primary score-network architecture is a multi-layer perception model with 3 hidden layers with 256 neurons in each layer. Following the implementation section 2.3, as our perturbation process is run over the fixed time interval $[0, 1]$ with step-size $\Delta t = 0.01$ our score network is trained by using the de-noising score matching objective (2.16) at each time $j\Delta t,\ j = 0,\ \ldots,\ 100$. The network is trained for $5,000$ epochs as experimentally this proved to be a nice balance between learning and avoiding over-fitting, however this quantity is very dataset dependent. The network was trained using the Adam optimiser and Relu activation functions between the layers.

All of the above specifications are maintained unless explicitly otherwise stated in an experiment.

B.2 Generative-Adversarial Network

The generator network learns a mapping from one-dimensional latent space to two dimensional observable space. We used increasing numbers of neurons in each layer, given by $\{16, 32, 64, 128, 256\}$ neurons respectively. We used the LeakyRelu activation between layers. For the discriminator network we learned a mapping from observable space to $[0, 1]$. For this network we used decreasing numbers of neurons in each layer as follows $\{512, 256, 128, 64, 32, 16\}$, again we used the LeakyRelu activation function with a final activation of Sigmoid to output a probability associated with the belief a sample is synthetic or real. The discriminator network was trained with a dropout rate of 30%. These networks were trained simultaneously with a learning rate of $5\text{e} - 4$ under the Adam optimiser using binary cross-entropy as a loss function.

B.3 Variational Auto-Encoder

The encoder network learns a mapping to the parameterised latent space $\mathbb{R}$. This network has decreasing numbers of neurons in each layer specified by $\{128, 64, 32, 16\}$ neurons respectively. The decoder learns the mapping from latent space back to observable space. This network has an increasing numbers of neurons in each layer, given by $\{16, 32, 64, 128, 256\}$ neurons respectively. Both networks used the LeakyRelu activation function with no dropout. These networks were trained to minimise a loss function comprised of a mean squared error from the training set and KL-divergence. We used the Adam optimiser with a learning rate of $5\text{e} - 4$.

Appendix C

Supplementary experiments

C.1 Score Matching On One Dimensional Gaussian Distributions

As an example, I will demonstrate how we can apply score matching to estimate the score of a one-dimensional Gaussian variable. Let's define the following model density: $f_{M,S}(x) = \frac{1}{2}(x - M)^T S(x - M),\ M, S \in \mathbb{R}$. Given N samples from a normal distribution (in this experiment I chose $\mu = 0$ and $\sigma = 1$) we can use our estimator for $J(\theta)$ to find the values of M and S which minimise the fisher distance between our model score and the true score. I did this in Python by generating $100,000$ samples from $\mathcal{N}(\mu, \sigma^2)$ and then performed gradient descent on $\hat{J}(\theta)$ to find the parameters θ where this estimator is minimised (Fig. C.1). As there are just two parameters in the one-dimensional Gaussian case, we can visualise the value of $\hat{J}(\theta)$ as M and S vary on the axes. This visualisation is shown below in Fig. C.2. Interestingly, in the case of the multivariate Gaussian, Hyvarinen showed that the estimator in score matching is exactly the maximum likelihood estimator for the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ [5].

```
Optimization terminated successfully.
         Current function value: -0.500391
         Iterations: 16
         Function evaluations: 57
         Gradient evaluations: 19
[2.03381772e-04 9.99608632e-01]
```

Figure C.1: Output of gradient descent on our estimator $\hat{J}(\theta)$ has found the optimal parameters $\hat{M} = 0.000204, \hat{S} = 0.9996$, very close to our true values of 0 and 1.

C.2 Wasserstein Distance Between One Dimensional Gaussian Distributions

Due to the abstract definition of the Wasserstein distance (3.4), I have plotted the Wasserstein distance between two one-dimensional distributions $N_1 \sim \mathcal{N}(0, 1),\ N_2 \sim \mathcal{N}(\mu, 1),\ \mu \in [-2.5, 2.5]$, in Fig. (C.3) below. The left-hand graph shows the overlap and provides visual intuition as to why the distance decreases as $\mu \to 0$. The right-hand graph shows the calculated Wasserstein distance between the two distributions. As expected, the right-hand graph shows the Wasserstein distance decrease as the distributions 'overlap' more and increase again as the mean passes 0 and the distributions 'agree' less.

C.3 Langevin Dynamics On an Unequally Weighted Gaussian Mixture Distribution

In the below example I demonstrate upon attempting to sample from a Gaussian mixture with unequal weights, the generated samples will not respect the weightings. In this example, the true

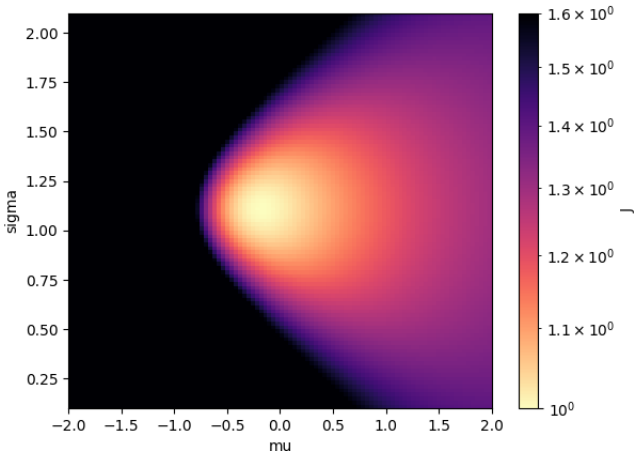

Figure C.2: The heat map appears to have a single minimum in proximity of the true values of M and S.

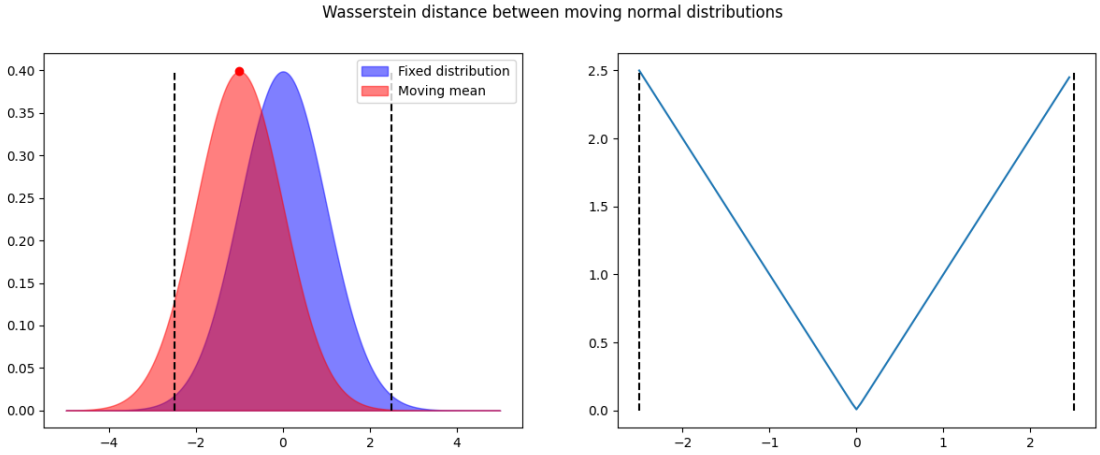

Figure C.3: The red distribution 'moves' from right to left, it's mean is initially on the left dotted line and linearly interpolates to the right dotted line.

score function of the parent distribution was used in order to demonstrate this is an issue derived from Langevin dynamics and not due to score estimation. This can be observed in Figure C.4. We are attempting to generate samples from the distribution:

$$p(x) = \frac{1}{10}\exp(-\frac{1}{2}(x-M_1)^T S_1(x-M_1)) + \frac{9}{10}\exp(-\frac{1}{2}(x-M_2)^T S_2(x-M_2))$$

$$M_1 = \begin{bmatrix} -5 \\ -5 \end{bmatrix},\ M_2 = \begin{bmatrix} 5 \\ 5 \end{bmatrix},\ S_1 = S_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The prior samples x_0 were uniformly scattered around the region $D := \{(x,y)| -7 \leq x \leq 7, -7 \leq y \leq 7\}$. For clarity, I have coloured the points that are initially under the region of low score (approximately the line $y = -x$) as blue and those which are initialised above as purple. From Fig. C.4, we can see only one particle was able to cross this region and therefore as the samples were initially distributed uniformly, they converge to a different mixture with $w_i \approx 0.5$.

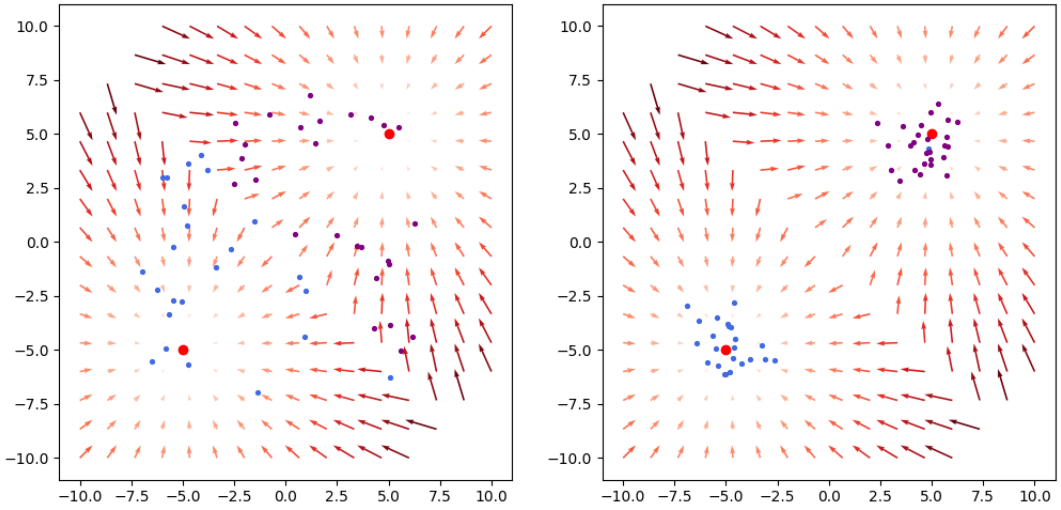

Figure C.4: The left hand side shows the initial dispersion of particles and the right hand side depicts the diffusion at $T = 1$. The relative equality in the two components demonstrates Langevin dynamics not respecting the weightings of each mixture component.

Appendix D

Implementations

D.1 Sliced Wassserstein Distance

In order to efficiently calculate the sliced Wasserstein distance on high dimensional datasets, we require a fast vectorised implementation. This does not exist in public libraries and therefore I wrote my own implementation using Jax.

```
def sliced_wasserstein(test_samples, generated_samples, num_projections
    , key):
    data_dimension = test_samples.shape[1]
    direction_vectors = random.normal(key, (data_dimension,
        num_projections))
    direction_vectors /= vmap(jnp.linalg.norm)(direction_vectors.T).T
    projections_test_samples = jnp.dot(test_samples, direction_vectors)
    projections_generated_samples = jnp.dot(generated_samples,
        direction_vectors)
    result = vmap(wasserstein_distance, projections_test_samples,
        projections_generated_samples, axis=1)
    return jnp.mean(result)
```

This is efficient as the projections are generated and performed as matrix operations enabling full use of the optimisations Jax offers.