Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech

Vadim Popov [*1] Ivan Vovk [*1] Vladimir Gogoryan [1] Tasnima Sadekova [1] Mikhail Kudinov [1]

Abstract

Recently, denoising diffusion probabilistic models and generative score matching have shown high potential in modelling complex data distributions while stochastic calculus has provided a unified point of view on these techniques allowing for flexible inference schemes. In this paper we introduce Grad-TTS, a novel text-to-speech model with score-based decoder producing mel-spectrograms by gradually transforming noise predicted by encoder and aligned with text input by means of Monotonic Alignment Search. The framework of stochastic differential equations helps us to generalize conventional diffusion probabilistic models to the case of reconstructing data from noise with different parameters and allows to make this reconstruction flexible by explicitly controlling trade-off between sound quality and inference speed. Subjective human evaluation shows that Grad-TTS is competitive with state-of-the-art text-to-speech approaches in terms of Mean Opinion Score. We will make the code publicly available shortly.

1. Introduction

Deep generative modelling proved to be effective in various machine learning fields, and speech synthesis is no exception. Modern text-to-speech (TTS) systems often consist of two parts designed as deep neural networks: the first part converts the input text into time-frequency domain acoustic features (*feature generator*), and the second one synthesizes raw waveform conditioned on these features (*vocoder*). Introduction of the conventional state-of-the-art autoregressive models such as Tacotron2 (Shen et al., 2018) used for feature generation and WaveNet (van den Oord et al., 2016) used as vocoder marked the beginning of the neural TTS era. Later, other popular generative modelling frameworks such as Generative Adversarial Networks (Goodfellow et al., 2014) and Normalizing Flows (Rezende & Mohamed, 2015) were used in the design of TTS engines for a parallel generation with comparable quality of the synthesized speech.

Since the publication of the WaveNet paper (2016), there have been various attempts to propose a parallel non-autoregressive vocoder, which could synthesize high-quality speech. Popular architectures based on Normalizing Flows like Parallel WaveNet (van den Oord et al., 2018) and WaveGlow (Prenger et al., 2019) managed to accelerate inference while keeping synthesis quality at a very high level but demonstrated fast synthesis on GPU devices only. Eventually, parallel GAN-based vocoders such as Parallel WaveGAN (Yamamoto et al., 2020), MelGAN (Kumar et al., 2019), and HiFi-GAN (Kong et al., 2020) greatly improved the performance of waveform generation on CPU devices. Furthermore, the latter model is reported to produce speech samples of state-of-the-art quality outperforming WaveNet.

Among feature generators, Tacotron2 (Shen et al., 2018) and Transformer-TTS (Li et al., 2019) enabled highly natural speech synthesis. Producing acoustic features frame by frame, they achieve almost perfect mel-spectrogram reconstruction from input text. Nonetheless, they often suffer from computational inefficiency and pronunciation issues coming from attention failures. Addressing these problems, such models as FastSpeech (Ren et al., 2019) and Parallel Tacotron (Elias et al., 2020) substantially improved inference speed and pronunciation robustness by utilizing non-autoregressive architectures and building hard monotonic alignments from estimated token lengths. However, in order to learn character duration, they still require pre-computed alignment from the teacher model. Finally, the recently proposed Non-Attentive Tacotron framework (Shen et al., 2020) managed to learn durations implicitly by employing the Variational Autoencoder concept.

Glow-TTS feature generator (Kim et al., 2020) based on Normalizing Flows can be considered as one of the most successful attempts to overcome pronunciation and computational latency issues typical for autoregressive solutions. Glow-TTS model made use of Monotonic Alignment Search algorithm (an adoption of Viterbi training (Rabiner, 1989) finding the most likely hidden alignment between two sequences) proposed to map the input text to mel-

[*]Equal contribution [1]Huawei Noah's Ark Lab, Moscow, Russia. Correspondence to: Vadim Popov <vadim.popov@huawei.com>, Ivan Vovk <vovk.ivan@huawei.com>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

spectrograms efficiently. The alignment learned by Glow-TTS is intentionally designed to avoid some of the pronunciation problems models like Tacotron2 suffer from. Also, in order to enable parallel synthesis, Glow-TTS borrows encoder architecture from Transformer-TTS (Li et al., 2019) and decoder architecture from Glow (Kingma & Dhariwal, 2018). Thus, compared with Tacotron2, Glow-TTS achieves much faster inference making fewer alignment mistakes. Besides, in contrast to other parallel TTS solutions such as FastSpeech, Glow-TTS does not require an external aligner to obtain token duration information as Monotonic Alignment Search (MAS) operates in an unsupervised way.

Lately, another family of generative models called Diffusion Probabilistic Models (DPMs) (Sohl-Dickstein et al., 2015) has started to prove its capability to model complex data distributions such as images (Ho et al., 2020), shapes (Cai et al., 2020), graphs (Niu et al., 2020), handwriting (Luhman & Luhman, 2020). The basic idea behind DPMs is as follows: we build a forward diffusion process by iteratively destroying original data until we get some simple distribution (usually standard normal), and then we try to build a reverse diffusion parameterized with a neural network so that it follows the trajectories of the reverse-time forward diffusion. Stochastic calculus offers a continuous easy-to-use framework for training DPMs (Song et al., 2021) and, which is perhaps more important, provides a number of flexible inference schemes based on numerical differential equation solvers.

As far as text-to-speech applications are concerned, two vocoders representing the DPM family showed impressive results in raw waveform reconstruction: WaveGrad (Chen et al., 2021) and DiffWave (Kong et al., 2021) were shown to reproduce the fine-grained structure of human speech and match strong autoregressive baselines such as WaveNet in terms of synthesis quality while at the same time requiring much fewer sequential operations. However, despite such a success in neural vocoding, no feature generator based on diffusion probabilistic modelling is known so far.

This paper introduces Grad-TTS, an acoustic feature generator with a score-based decoder using recent diffusion probabilistic modelling insights. In Grad-TTS, MAS-aligned encoder outputs are passed to the decoder that transforms Gaussian noise parameterized by these outputs into a mel-spectrogram. To cope with the task of reconstructing data from Gaussian noise with varying parameters, we write down a generalized version of conventional forward and reverse diffusions. One of the remarkable features of our model is that it provides explicit control of the trade-off between output mel-spectrogram quality and inference speed. In particular, we find that Grad-TTS is capable of generating mel-spectrograms of high quality with only as few as ten iterations of reverse diffusion, which makes it possible to outperform Tacotron2 in terms of speed on GPU devices. Additionally, we show that it is possible to train Grad-TTS as an end-to-end TTS pipeline (i.e., vocoder and feature generator are combined in a single model) by replacing its output domain from mel-spectrogram to raw waveform.

2. Diffusion probabilistic modelling

Loosely speaking, a process of the diffusion type is a stochastic process that satisfies a stochastic differential equation (SDE)

$$dX_t = b(X_t, t)dt + a(X_t, t)dW_t, \tag{1}$$

where W_t is the standard Brownian motion, $t \in [0, T]$ for some finite time horizon T, and coefficients b and a (called *drift* and *diffusion* correspondingly) satisfy certain measurability conditions. A rigorous definition of the diffusion type processes, as well as other notions from stochastic calculus we use in this section, can be found in (Liptser & Shiryaev, 1978).

It is easy to find such a stochastic process that terminal distribution $Law(X_T)$ converges to standard normal $\mathcal{N}(0, I)$ when $T \to \infty$ for any initial data distribution $Law(X_0)$ (I is $n \times n$ identity matrix and n is data dimensionality). In fact, there are lots of such processes as it follows from the formulae given later in this section. Any process of the diffusion type with such property is called *forward diffusion* and the goal of diffusion probabilistic modelling is to find a *reverse diffusion* such that its trajectories closely follow those of the forward diffusion but in reverse time order. This is, of course, a much harder task than making Gaussian noise out of data, but in many cases it still can be accomplished if we parameterize reverse diffusion with a proper neural network. In this case, generation boils down to sampling random noise from $\mathcal{N}(0, I)$ and then just solving the SDE describing dynamics of the reverse diffusion with any numerical solver (usually a simple first-order Euler-Maruyama scheme (Kloeden & Platen, 1992) is used). If forward and reverse diffusion processes have close trajectories, then the distribution of resulting samples will be very close to that of the data $Law(X_0)$. This approach to generative modelling is summarized in Figure 1.

Until recently, score-based and denoising diffusion probabilistic models were formalized in terms of Markov chains (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020; Song & Ermon, 2020). A unified approach introduced by Song et al. (2021) has demonstrated that these Markov chains actually approximated trajectories of stochastic processes satisfying certain SDEs. In our work, we follow this paper and define our DPM in terms of SDEs rather than Markov chains. As one can see later in Section 3, the task we are solving suggests generalizing DPMs described

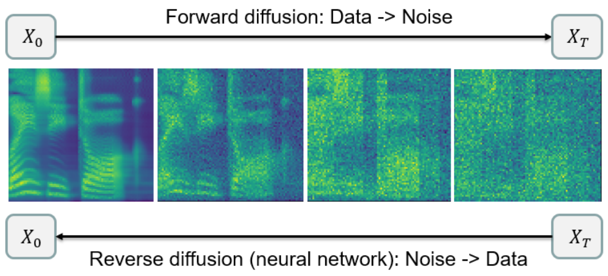
Forward diffusion: Data -> Noise
X_0
X_T
Reverse diffusion (neural network): Noise -> Data

Figure 1. Diffusion probabilistic modelling for mel-spectrograms.

in (Song et al., 2021) in such a way that for infinite time horizon forward diffusion transforms any data distribution into $\mathcal{N}(\mu, \Sigma)$ instead of $\mathcal{N}(0, I)$ for any given mean μ and diagonal covariance matrix Σ. So, the rest of this section contains the detailed description of the generalized forward and reverse diffusions we utilize as well as the loss function we optimize to train the reverse diffusion. All corresponding derivations can be found in Appendix.

2.1. Forward diffusion

First, we need to define a forward diffusion process that transforms any data into Gaussian noise given infinite time horizon T. If n-dimensional stochastic process X_t satisfies the following SDE:

$$dX_t = \frac{1}{2}\Sigma^{-1}(\mu - X_t)\beta_t dt + \sqrt{\beta_t}dW_t, \quad t \in [0, T] \tag{2}$$

for non-negative function β_t, which we will refer to as noise schedule, vector μ, and diagonal matrix Σ with positive elements, then its solution (if it exists) is given by

$$\begin{aligned} X_t = &\left(I - e^{-\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}\right)\mu + e^{-\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}X_0 \\ &+ \int_0^t \sqrt{\beta_s}e^{-\frac{1}{2}\Sigma^{-1}\int_s^t \beta_u du}dW_s. \end{aligned} \tag{3}$$

Note that the exponential of a diagonal matrix is just an element-wise exponential. Let

$$\begin{aligned} \rho(X_0, \Sigma, \mu, t) = &\left(I - e^{-\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}\right)\mu \\ &+ e^{-\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}X_0 \end{aligned} \tag{4}$$

and

$$\lambda(\Sigma, t) = \Sigma\left(I - e^{-\Sigma^{-1}\int_0^t \beta_s ds}\right). \tag{5}$$

By properties of Itô's integral conditional distribution of X_t given X_0 is Gaussian:

$$Law(X_t|X_0) = \mathcal{N}(\rho(X_0, \Sigma, \mu, t), \lambda(\Sigma, t)). \tag{6}$$

It means that if we consider infinite time horizon then for *any* noise schedule β_t such that $\lim_{t\to\infty} e^{-\int_0^t \beta_s ds} = 0$ we have

$$X_t|X_0 \xrightarrow{d} \mathcal{N}(\mu, \Sigma). \tag{7}$$

So, random variable X_t converges in distribution to $\mathcal{N}(\mu, \Sigma)$ independently of X_0, and it is exactly the property we need: forward diffusion satisfying SDE (2) transforms any data distribution $Law(X_0)$ into Gaussian noise $\mathcal{N}(\mu, \Sigma)$.

2.2. Reverse diffusion

While in earlier works on DPMs reverse diffusion was trained to approximate the trajectories of forward diffusion, Song et al. (2021) proposed to use the result by Anderson (1982), who derived an explicit formula for reverse-time dynamics of a wide class of stochastic processes of the diffusion type. In our case, this result leads to the following SDE for the reverse diffusion:

$$\begin{aligned} dX_t = &\left(\frac{1}{2}\Sigma^{-1}(\mu - X_t) - \nabla \log p_t(X_t)\right)\beta_t dt \\ &+ \sqrt{\beta_t}d\widetilde{W}_t, \qquad t \in [0, T], \end{aligned} \tag{8}$$

where $\widetilde{W}_t$ is the reverse-time Brownian motion and p_t is the probability density function of random variable X_t. This SDE is to be solved backwards starting from terminal condition X_T.

Moreover, Song et al. (2021) have shown that instead of SDE (8), we can consider an ordinary differential equation

$$dX_t = \frac{1}{2}\left(\Sigma^{-1}(\mu - X_t) - \nabla \log p_t(X_t)\right)\beta_t dt. \tag{9}$$

Forward Kolmogorov equations corresponding to (2) and (9) are identical, which means that the evolution of probability density functions of stochastic processes given by (2) and (9) is the same.

Thus, if we have a neural network $s_\theta(X_t, t)$ that estimates the gradient of the log-density of noisy data $\nabla \log p_t(X_t)$, then we can model data distribution $Law(X_0)$ by sampling X_T from $\mathcal{N}(\mu, \Sigma)$ and numerically solving either (8) or (9) backwards in time.

2.3. Loss function

Estimating gradients of log-density of noisy data X_t is often referred to as *score matching*, and in recent papers (Song & Ermon, 2019; 2020) L_2 loss was used to approximate these gradients with a neural network. So, in our paper, we use the same type of loss.

Due to the formula (6), we can sample noisy data X_t given only initial data X_0 without sampling intermediate values $\{X_s\}_{s<t}$. Moreover, $Law(X_t|X_0)$ is Gaussian, which means that its log-density has a very simple closed form. If we sample ϵ_t from $\mathcal{N}(0, \lambda(\Sigma, t))$ and then put

$$X_t = \rho(X_0, \Sigma, \mu, t) + \epsilon_t \tag{10}$$

in accordance with (6), then the gradient of log-density of noisy data in this point X_t is given by

$$\nabla \log p_{0t}(X_t|X_0) = -\lambda(\Sigma, t)^{-1}\epsilon_t, \tag{11}$$

where $p_{0t}(\cdot|X_0)$ is the probability density function of the conditional distribution (6). Thus, loss function corresponding to estimating the gradient of log-density of data X_0 corrupted with noise accumulated by time t is

$$\mathcal{L}_t(X_0) = \mathbb{E}_{\epsilon_t}\left[\left\|s_\theta(X_t, t) + \lambda(\Sigma, t)^{-1}\epsilon_t\right\|_2^2\right], \tag{12}$$

where ϵ_t is sampled from $\mathcal{N}(0, \lambda(\Sigma, t))$ and X_t is calculated by formula (10).

3. Grad-TTS

The acoustic feature generator we propose consists of three modules: encoder, duration predictor, and decoder. In this section, we will describe their architectures as well as training and inference procedures. The general approach is illustrated in Figure 2. Grad-TTS has very much in common with Glow-TTS (Kim et al., 2020), a feature generator based on Normalizing Flows. The key difference lies in the principles the decoder relies on.

3.1. Inference

An input text sequence $x_{1:L}$ of length L typically consists of characters or phonemes, and we aim at generating mel-spectrogram $y_{1:F}$ where F is the number of acoustic frames. In Grad-TTS, the encoder converts an input text sequence $x_{1:L}$ into a sequence of features $\tilde{\mu}_{1:L}$ used by the duration predictor to produce hard monotonic alignment A between encoded text sequence $\tilde{\mu}_{1:L}$ and frame-wise features $\mu_{1:F}$. The function A is a monotonic surjective mapping between $[1, F] \cap \mathbb{N}$ and $[1, L] \cap \mathbb{N}$, and we put $\mu_j = \tilde{\mu}_{A(j)}$ for any integer $j \in [1, F]$. Informally speaking, the duration predictor tells us how many frames each element of text input lasts. Monotonicity and surjectiveness of A guarantee that the text is pronounced in the correct order without skipping any text input. As in all TTS models with duration predictor, it is possible to control synthesized speech tempo by multiplying predicted durations by some factor.

The output sequence $\mu = \mu_{1:F}$ is then passed to the decoder, which is a Diffusion Probabilistic Model. A neural network $s_\theta(X_t, \mu, t)$ with parameters θ defines an ordinary differential equation (ODE)

$$dX_t = \frac{1}{2}(\mu - X_t - s_\theta(X_t, \mu, t))\beta_t dt, \tag{13}$$

which is solved backwards in time using the first-order Euler scheme. The sequence μ is also used to define the terminal condition $X_T \sim \mathcal{N}(\mu, I)$. Noise schedule β_t and time horizon T are some pre-defined hyperparameters whose choice mostly depends on the data, while step size h in the Euler scheme is a hyperparameter that can be chosen after Grad-TTS is trained. It expresses the trade-off between the quality of output mel-spectrograms and inference speed.

Reverse diffusion in Grad-TTS evolves according to equation (13) for the following reasons:

- We obtained better results in practice when using dynamics (9) instead of (8): for small values of step size h, they performed equally well, while for larger values the former led to much better sounding results.
- We chose $\Sigma = I$ to simplify the whole feature generation pipeline.
- We used μ as an additional input to the neural network $s_\theta(X_t, \mu, t)$. It follows from (11) that the neural network s_θ essentially tries to predict Gaussian noise added to data X_0 observing only noisy data X_t. So, if for every time t we supply s_θ with an additional knowledge of how the limiting noise $\lim_{T\to\infty} Law(X_T|X_0)$ looks like (note that it is different for different text input), then this network can make more accurate predictions of noise at time $t \in [0, T]$.

We also found it beneficial for the model performance to introduce a temperature hyperparameter τ and to sample terminal condition X_T from $\mathcal{N}(\mu, \tau^{-1}I)$ instead of $\mathcal{N}(\mu, I)$. Tuning τ can help to keep the quality of output mel-spectrograms at the same level when using larger values of step size h.

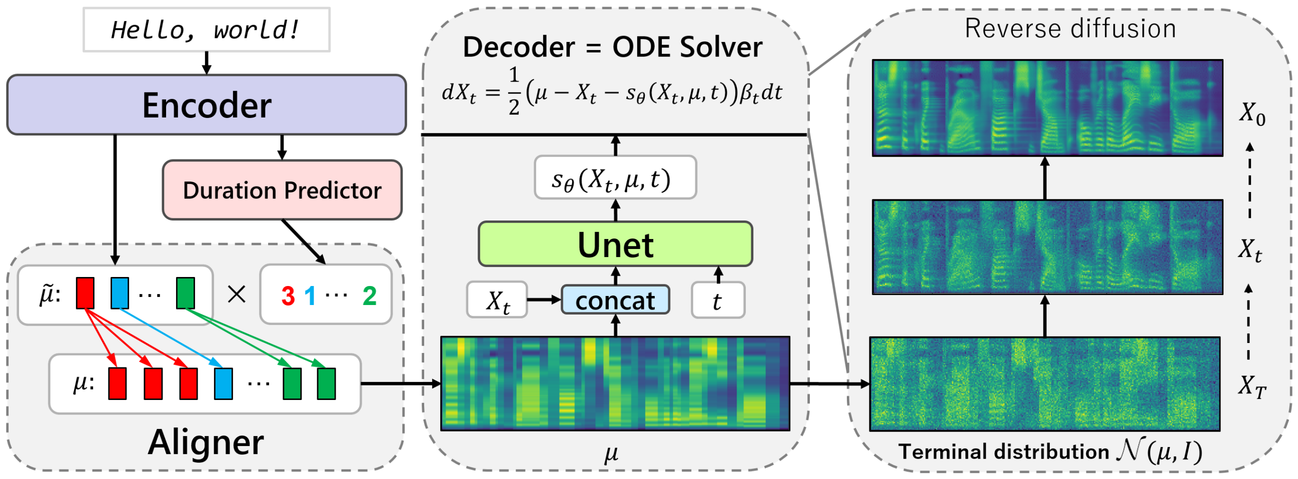

Figure 2. Grad-TTS inference scheme.

3.2. Training

One of Grad-TTS training objectives is to minimize the distance between aligned encoder output μ and target mel-spectrogram y because the inference scheme that has just been described suggests to start decoding from random noise $\mathcal{N}(\mu, I)$. Intuitively, it is clear that decoding is easier if we start from noise, which is already close to the target y in some sense.

If aligned encoder output μ is considered to parameterize an input noise the decoder starts from, it is natural to regard encoder output $\tilde{\mu}$ as a normal distribution $\mathcal{N}(\tilde{\mu}, I)$, which leads to a negative log-likelihood encoder loss:

$$\mathcal{L}_{enc} = -\sum_{j=1}^{F} \log \varphi(y_j; \tilde{\mu}_{A(j)}, I), \tag{14}$$

where $\varphi(\cdot; \tilde{\mu}_i, I)$ is a probability density function of $\mathcal{N}(\tilde{\mu}_i, I)$. Although other types of losses are also possible, we have chosen $\mathcal{L}_{enc}$ (which actually reduces to Mean Square Error criterion) because of this probabilistic interpretation. In principle, it is even possible to train Grad-TTS without any encoder loss at all and let the diffusion loss described further do all the job of generating realistic mel-spectrograms, but in practice we observed that in the absence of $\mathcal{L}_{enc}$ Grad-TTS failed to learn alignment.

The encoder loss $\mathcal{L}_{enc}$ has to be optimized with respect to both encoder parameters and alignment function A. Since it is hard to do a joint optimization, we apply an iterative approach proposed by Kim et al. (2020). Each iteration of optimization consists of two steps: (i) searching for an optimal alignment A^* given fixed encoder parameters; (ii) fixing this alignment A^* and taking one step of stochastic gradient descent to optimize loss function with respect to encoder parameters. We use Monotonic Alignment Search at the first step of this approach. MAS utilizes the concept of dynamic programming to find an optimal (from the point of view of loss function $\mathcal{L}_{enc}$) monotonic surjective alignment. This algorithm is described in detail in (Kim et al., 2020).

To estimate the optimal alignment A^* at inference, Grad-TTS employs the duration predictor network. As in (Kim et al., 2020), we train the duration predictor DP with Mean Square Error (MSE) criterion in logarithmic domain:

$$\begin{gathered} d_i = \log \sum_{j=1}^{F} \mathbb{I}_{\{A^*(j)=i\}}, \quad i = 1, .., L, \\ \mathcal{L}_{dp} = MSE(DP(sg[\tilde{\mu}]), d), \end{gathered} \tag{15}$$

where $\mathbb{I}$ is an indicator function, $\tilde{\mu} = \tilde{\mu}_{1:L}$, $d = d_{1:L}$ and stop gradient operator $sg[\cdot]$ is applied to the inputs of the duration predictor to prevent $\mathcal{L}_{dp}$ from affecting encoder parameters.

As for the loss related to the DPM, it is calculated using formulae from Section 2. As already mentioned, we put $\Sigma = I$, so the distribution of noisy data (6) simplifies, and its covariance matrix becomes just an identity matrix I multiplied by a scalar

$$\lambda_t = 1 - e^{-\int_0^t \beta_s ds}. \tag{16}$$

The overall diffusion loss function $\mathcal{L}_{diff}$ is the expectation of weighted losses associated with estimating gradients of log-density of noisy data at different times $t \in [0, T]$:

$$\mathcal{L}_{diff} = \mathbb{E}_{X_0,t}\left[\lambda_t \mathbb{E}_{\xi_t}\left[\left\| s_\theta(X_t,\mu,t) + \frac{\xi_t}{\sqrt{\lambda_t}}\right\|_2^2\right]\right], \tag{17}$$

where X_0 stands for target mel-spectrogram y sampled from training data, t is sampled from uniform distribution on $[0,T]$, ξ_t – from $\mathcal{N}(0,I)$ and the formula

$$X_t = \rho(X_0,I,\mu,t) + \sqrt{\lambda_t}\xi_t \tag{18}$$

is used to get noisy data X_t according to the distribution (6). The above formulae (17) and (18) follow from (12) and (10) by substitution $\epsilon_t = \sqrt{\lambda_t}\xi_t$. We use losses (12) with weights λ_t according to the common heuristics that these weights should be proportional to $1/\mathbb{E}\left[\|\nabla \log p_{0t}(X_t|X_0)\|_2^2\right]$.

To sum it up, the training procedure consists of the following steps:

- Fix the encoder, duration predictor, and decoder parameters and run MAS algorithm to find the alignment A^* that minimizes $\mathcal{L}_{enc}$.
- Fix the alignment A^* and minimize $\mathcal{L}_{enc}+\mathcal{L}_{dp}+\mathcal{L}_{diff}$ with respect to encoder, duration predictor, and decoder parameters.
- Repeat the first two steps till convergence.

3.3. Model architecture

As for the encoder and duration predictor, we use exactly the same architectures as in Glow-TTS, which in its turn borrows the structure of these modules from Transformer-TTS (Li et al., 2019) and FastSpeech (Ren et al., 2019) correspondingly. The duration predictor consists of two convolutional layers followed by a projection layer that predicts the logarithm of duration. The encoder is composed of a pre-net, 6 Transformer blocks with multi-head self-attention, and the final linear projection layer. The pre-net consists of 3 layers of convolutions followed by a fully-connected layer.

The decoder network s_θ has the same U-Net architecture (Ronneberger et al., 2015) used by Ho et al. (2020) to generate 32×32 images, except that we use twice fewer channels and three feature map resolutions instead of four to reduce model size. In our experiments we use 80-dimensional mel-spectrograms, so s_θ operates on resolutions $80 \times F$, $40 \times F/2$ and $20 \times F/4$. We zero-pad mel-spectrograms if the number of frames F is not a multiple of 4. Aligned encoder output μ is concatenated with U-Net input X_t as an additional channel.

4. Experiments

LJSpeech dataset (Ito, 2017) containing approximately 24 hours of English female voice recordings sampled at 22.05kHz was used to train the Grad-TTS model. The test set contained around 500 short audio recordings (duration less than 10 seconds each). The input text was phonemized before passing to the encoder; as for the output acoustic features, we used conventional 80-dimensional mel-spectrograms. We tried training both on original and normalized mel-spectrograms and found that the former performed better. Grad-TTS was trained for $1.7m$ iterations on a single GPU (NVIDIA RTX 2080 Ti with 11GB memory) with mini-batch size 16. We chose Adam optimizer and set the learning rate to 0.0001.

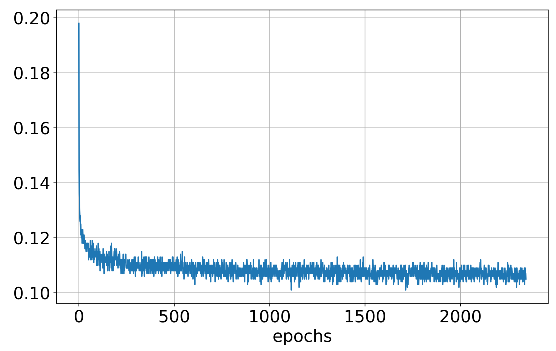

Figure 3. Diffusion loss at training.

We would like to mention several important things about Grad-TTS training:

- We chose $T = 1$, $\beta_t = \beta_0 + (\beta_1 - \beta_0)t$, $\beta_0 = 0.05$ and $\beta_1 = 20$.
- As in (Bińkowski et al., 2020; Donahue et al., 2021), we use random mel-spectrogram segments of fixed length (2 seconds in our case) as training targets y to allow for memory-efficient training. However, MAS and the duration predictor still use whole mel-spectrograms.
- Although diffusion loss $\mathcal{L}_{diff}$ seems to converge very slowly after the beginning epochs, as shown on Figure 3, such long training is essential to get a good model because the neural network s_θ has to learn to estimate gradients accurately for all $t \in [0,1]$. Two models with almost equal diffusion losses can produce mel-spectrograms of very different quality: inaccurate predictions for a small subset $S \subset [0,1]$ may have a small impact on $\mathcal{L}_{diff}$ but be crucial for the output mel-spectrogram quality if ODE solver involves calculating s_θ in at least one point belonging to S.

Once trained, Grad-TTS enables the trade-off between quality and inference speed due to the ability to vary the number of steps N the decoder takes to solve ODE (13) at inference. So, we evaluate four models which we denote by Grad-TTS-N where $N \in [4, 10, 100, 1000]$. We use $\tau = 1.5$ at synthesis for all four models. As baselines, we take an official implementation of Glow-TTS (Kim et al., 2020), the model which resembles ours to the most extent among the existing feature generators, FastSpeech (Ren et al., 2019), and state-of-the-art Tacotron2 (Shen et al., 2018). Recently proposed HiFi-GAN (Kong et al., 2020) is known to provide excellent sound quality, so we use this vocoder with all models we compare.

4.1. Subjective evaluation

To make subjective evaluation of TTS models, we used the crowdsourcing platform Amazon Mechanical Turk. For Mean Opinion Score (MOS) estimation we synthesized 40 sentences from the test set with each model. The assessors were asked to estimate the quality of synthesized speech on a nine-point Likert scale, the lowest and the highest scores being 1 point ("Bad") and 5 points ("Excellent") with a step of 0.5 point. To ensure the reliability of the obtained results, only Master assessors were assigned to complete the listening test. Each audio was evaluated by 10 assessors. A small subset of speech samples used in the test is available at `https://grad-tts.github.io/`.

Table 1. Ablation study of proposed generalized diffusion framework. Grad-TTS reconstructing data from $\mathcal{N}(0, I)$ for N reverse diffusion iterations is compared with the baseline Grad-TTS-10 – the model reconstructing data from $\mathcal{N}(\mu, I)$ for 10 iterations.

N	Worse, %	Identical, %	Better, %
10	93.8	0.5	5.7
20	82.3	2.9	14.8
50	60.3	5.7	34.0

MOS results with 95% confidence intervals are presented in Table 2. It demonstrates that although the quality of the synthesized speech gets better when we use more iterations of the reverse diffusion, the quality gain becomes marginal starting from a certain number of iterations. In particular, there is almost no difference between Grad-TTS-1000 and Grad-TTS-10 in terms of MOS, while the gap between Grad-TTS-10 and Grad-TTS-4 (4 was the smallest number of iterations leading to satisfactory quality) is much more significant. As for other feature generators, Grad-TTS-10 is competitive with all compared models, including state-of-the-art Tacotron2. Furthermore, Grad-TTS-1000 achieves almost natural synthesis with MOS being less than that for ground truth recordings by only 0.1. We would like to note that the relatively low results of FastSpeech could possibly be explained by the fact that we used its unofficial implementation `https://github.com/xcmyz/FastSpeech`.

To verify the benefits of the proposed generalized DPM framework we trained the model with the same architecture as Grad-TTS to reconstruct mel-spectrograms from $\mathcal{N}(0, I)$ instead of $\mathcal{N}(\mu, I)$. The preference test provided in Table 1 shows that Grad-TTS-10 is significantly better ($p < 0.005$ in sign test) than this model taking 10, 20 and even 50 iterations of the reverse diffusion. It demonstrates that the model trained to generate from $\mathcal{N}(0, I)$ needs more steps of ODE solver to get high-quality mel-spectrograms than Grad-TTS we propose. We believe this is because the task of reconstructing mel-spectrogram from pure noise $\mathcal{N}(0, I)$ is more difficult than the one of reconstructing it from its noisy copy $\mathcal{N}(\mu, I)$. One possible objection could be that the model trained with $\mathcal{N}(0, I)$ as terminal distribution can just add μ to this noise at the first step of sampling (it is possible because s_θ has μ as its input) and then repeat the same steps as our model to generate data from $N(\mu, I)$. In this case, it would generate mel-spectrograms of the same quality as our model taking only one step more. However, this argument is wrong, since reverse diffusion removes noise not arbitrarily, but according to the reverse trajectories of the forward diffusion. Since forward diffusion adds noise gradually, reverse diffusion has to remove noise gradually as well, and the first step of the reverse diffusion cannot be adding μ to Gaussian noise with zero mean because the last step of the forward diffusion is not a jump from μ to zero.

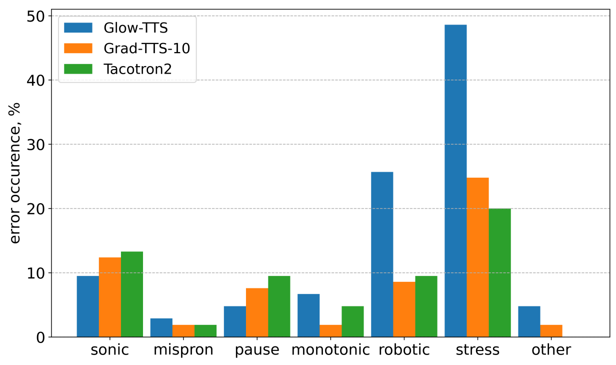

Figure 4. Typical errors occurrence.

We also made an attempt to estimate what kinds of mistakes are characteristic of certain models. We compared Tacotron2, Glow-TTS, and Grad-TTS-10 as the fastest version of our model with high synthesis quality. Each record was estimated by 5 assessors. Figure 4 demonstrates the results of the multiple-choice test whose participants had to choose which kinds of errors (if any) they could hear: sonic artifacts like clicking sounds or back-

Table 2. Model comparison.

Model	Enc params[1]	Dec params	RTF	Log-likelihood	MOS
Grad-TTS-1000	$7.2m$	$7.6m$	3.663	$\mathbf{0.174 \pm 0.001}$	$\mathbf{4.44 \pm 0.05}$
Grad-TTS-100			0.363		4.38 ± 0.06
Grad-TTS-10			0.033		4.38 ± 0.06
Grad-TTS-4			0.012		3.96 ± 0.07
Glow-TTS	$7.2m$	$21.4m$	0.008	0.082	4.11 ± 0.07
FastSpeech	$24.5m$		**0.004**	–	3.68 ± 0.09
Tacotron2	$28.2m$		0.075	–	4.32 ± 0.07
Ground Truth	–		–	–	4.53 ± 0.06

ground noise ("sonic" in the figure), mispronunciation of words/phonemes ("mispron"), unnatural pauses ("pause"), monotone speech ("monotonic"), robotic voice ("robotic"), wrong word stressing ("stress") or others. It is clear from the figure that Glow-TTS frequently stresses words in a wrong way, and the sound it produces is perceived as "robotic" in around a quarter of cases. These are the major factors that make Glow-TTS performance inferior to that of Grad-TTS and Tacotron2, which in their turn have more or less the same drawbacks in terms of synthesis quality.

4.2. Objective evaluation

Although DPMs can be shown to maximize weighted variational lower bound (Ho et al., 2020) on data log-likelihood, they do not explicitly optimize exact data likelihood. In spite of this, Song et al. (2021) show that it is still possible to calculate it using the instantaneous change of variables formula (Chen et al., 2018) if we look at DPMs from the "continuous" point of view. However, it is necessary to use Hutchinson's trace estimator to make computations feasible, so in Table 2 log-likelihood for Grad-TTS comes with a 95% confidence interval.

We randomly chose 50 sentences from the test set and calculated their average log-likelihood under two probabilistic models we consider – Glow-TTS and Grad-TTS. Interestingly, Grad-TTS achieves better log-likelihood than Glow-TTS even though the latter has a decoder with 3x larger capacity and was trained to maximize exact data likelihood. Similar phenomena were observed by Song et al. (2021) in the image generation task.

4.3. Efficiency estimation

We assess the efficiency of the proposed model in terms of Real-Time Factor (RTF is how many seconds it takes to generate one second of audio) computed on GPU and the number of parameters. Table 2 contains efficiency information for all models under comparison. Additional information regarding absolute inference speed dependency on the input text length is given in Figure 5.

Due to its flexibility at inference, Grad-TTS is capable of real-time synthesis on GPU: if the number of decoder steps is less than 100, it reaches RTF < 0.37. Moreover, although it cannot compete with Glow-TTS and FastSpeech in terms of inference speed, it still can be approximately twice faster than Tacotron2 if we use 10 decoder iterations sufficient for getting high-fidelity mel-spectrograms. Besides, Grad-TTS has around $15m$ parameters, thus being significantly smaller than other feature generators we compare.

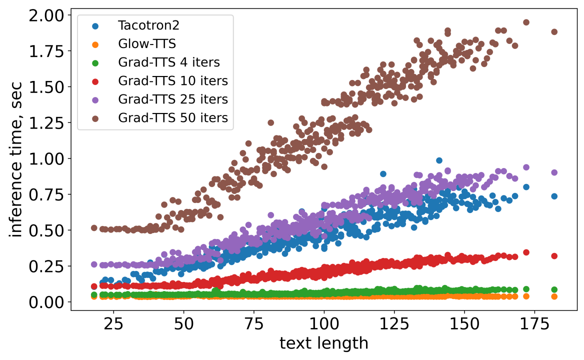

Figure 5. Inference speed comparison. Text length is given in characters.

4.4. End-to-end TTS

The results of our preliminary experiments show that it is also possible to train an end-to-end TTS model as a DPM. In brief, we moved from U-Net to WaveGrad (Chen et al., 2021) in Grad-TTS decoder: the overall architecture resembles WaveGrad conditioned on the aligned encoder output μ instead of ground truth mel-spectrograms y as in original WaveGrad. Although synthesized speech quality is fair enough, it cannot compete with the results reported above, so we do not include our end-to-end model in the listening test but provide demo samples at `https://grad-tts.github.io/`.

[1]Encoder and duration predictor parameters are calculated together.

5. Future work

End-to-end speech synthesis results reported above show that it is a promising future research direction for text-to-speech applications. However, there is also much room for investigating general issues regarding DPMs.

In the analysis in Section 2, we always assume that both forward and reverse diffusion processes exist, i.e., SDEs (2) and (8) have strong solutions. It applies some Lipschitz-type constraints (Liptser & Shiryaev, 1978) on noise schedule β_t and, what is more important, on the neural network s_θ. Wasserstein GANs offer an encouraging example of incorporating Lipschitz constraints into neural networks training (Gulrajani et al., 2017), suggesting that similar techniques may improve DPMs.

Little attention has been paid so far to the choice of the noise schedule β_t – most researchers use a simple linear schedule. Also, it is mostly unclear how to choose weights for losses (12) at time t in the global loss function optimally. A thorough investigation of such practical questions is crucial as it can facilitate applying DPMs to new machine learning problems.

6. Conclusion

We have presented Grad-TTS, the first acoustic feature generator utilizing the concept of diffusion probabilistic modelling. The main generative engine of Grad-TTS is the diffusion-based decoder that transforms Gaussian noise parameterized with the encoder output into mel-spectrogram while alignment is performed with Monotonic Alignment Search. The model we propose allows to vary the number of decoder steps at inference, thus providing a tool to control the trade-off between inference speed and synthesized speech quality. Despite its iterative decoding, Grad-TTS is capable of real-time synthesis. Moreover, it can generate mel-spectrograms twice faster than Tacotron2 while keeping synthesis quality competitive with common TTS baselines.

References

Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313 – 326, 1982. ISSN 0304-4149.

Bińkowski, M., Donahue, J., Dieleman, S., Clark, A., et al. High Fidelity Speech Synthesis with Adversarial Networks. In *International Conference on Learning Representations*, 2020.

Cai, R., Yang, G., Averbuch-Elor, H., Hao, Z., Belongie, S., Snavely, N., and Hariharan, B. Learning Gradient Fields for Shape Generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

Chen, N., Zhang, Y., Zen, H., Weiss, R. J., Norouzi, M., and Chan, W. WaveGrad: Estimating Gradients for Waveform Generation. In *International Conference on Learning Representations*, 2021.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, volume 31, pp. 6571–6583. Curran Associates, Inc., 2018.

Donahue, J., Dieleman, S., Binkowski, M., Elsen, E., and Simonyan, K. End-to-end Adversarial Text-to-Speech. In *International Conference on Learning Representations*, 2021.

Elias, I., Zen, H., Shen, J., Zhang, Y., Jia, Y., Weiss, R., and Wu, Y. Parallel Tacotron: Non-Autoregressive and Controllable TTS, 2020.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27*, pp. 2672–2680. Curran Associates, Inc., 2014.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved Training of Wasserstein GANs. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5767–5777. Curran Associates, Inc., 2017.

Ho, J., Jain, A., and Abbeel, P. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, virtual*, 2020.

Ito, K. The LJ Speech Dataset, 2017. URL `https://keithito.com/LJ-Speech-Dataset/`.

Kim, J., Kim, S., Kong, J., and Yoon, S. Glow-TTS: A Generative Flow for Text-to-Speech via Monotonic Alignment Search. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, virtual*, 2020.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, pp. 10236–10245, 2018.

Kloeden, P. E. and Platen, E. *Numerical Solution of Stochastic Differential Equations*, volume 23 of *Stochastic Modelling and Applied Probability*. Springer-Verlag Berlin Heidelberg, 1992.

Kong, J., Kim, J., and Bae, J. HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, virtual*, 2020.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. DiffWave: A Versatile Diffusion Model for Audio Synthesis. In *International Conference on Learning Representations*, 2021.

Kumar, K., Kumar, R., de Boissiere, T., Gestin, L., et al. MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis. In *Advances in Neural Information Processing Systems 32*, pp. 14910–14921. Curran Associates, Inc., 2019.

Li, N., Liu, S., Liu, Y., Zhao, S., and Liu, M. Neural Speech Synthesis with Transformer Network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6706–6713, 07 2019.

Liptser, R. and Shiryaev, A. *Statistics of Random Processes*, volume 5 of *Stochastic Modelling and Applied Probability*. Springer-Verlag, 1978.

Luhman, T. and Luhman, E. Diffusion models for Handwriting Generation, 2020.

Niu, C., Song, Y., Song, J., Zhao, S., Grover, A., and Ermon, S. Permutation Invariant Graph Generation via Score-Based Generative Modeling. In *AISTATS*, 2020.

Prenger, R., Valle, R., and Catanzaro, B. Waveglow: A Flow-based Generative Network for Speech Synthesis. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3617–3621. IEEE, May 2019.

Rabiner, L. A Tutorial on Hidden Markov Models and Selected Applications. *Proceedings of the IEEE*, 1989.

Ren, Y., Ruan, Y., Tan, X., Qin, T., et al. FastSpeech: Fast, Robust and Controllable Text to Speech. In *Advances in Neural Information Processing Systems 32*, pp. 3171–3180. Curran Associates, Inc., 2019.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 1530–1538, 2015.

Ronneberger, O., Fischer, P., and Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241. Springer International Publishing, 2015.

Shen, J., Pang, R., et al. Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4779–4783, April 2018.

Shen, J., Jia, Y., Chrzanowski, M., Zhang, Y., Elias, I., Zen, H., and Wu, Y. Non-Attentive Tacotron: Robust and Controllable Neural TTS Synthesis Including Unsupervised Duration Modeling. *ArXiv*, abs/2010.04301, 2020.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, Proceedings of Machine Learning Research, pp. 2256–2265. PMLR, 2015.

Song, Y. and Ermon, S. Generative Modeling by Estimating Gradients of the Data Distribution. In *Advances in Neural Information Processing Systems*, volume 32, pp. 11918–11930. Curran Associates, Inc., 2019.

Song, Y. and Ermon, S. Improved Techniques for Training Score-Based Generative Models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, virtual*, 2020.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*, 2021.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. WaveNet: A Generative Model for Raw Audio. In *9th ISCA Speech Synthesis Workshop*, pp. 125–125, 2016.

van den Oord, A., Li, Y., et al. Parallel WaveNet: Fast High-Fidelity Speech Synthesis. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3918–3926. PMLR, 10–15 Jul 2018.

Yamamoto, R., Song, E., and Kim, J.-M. Parallel Wavegan: A Fast Waveform Generation Model Based on Generative Adversarial Networks with Multi-Resolution Spectrogram. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6199–6203, 2020.

Appendix

We include an appendix with detailed derivations, proofs and additional information. Our proposed diffusion probabilistic framework employs generalized terminal distribution $\mathcal{N}(\mu, \Sigma)$ instead of $\mathcal{N}(0, I)$ as proposed by Song et al. (2021). The derivation for the solution (3) of SDE (2) that transforms the original data distribution to the terminal distribution is described in Appendix A. In Appendix B we also derive the distribution which the solution (3) for the diffused data X_t follows. Then, the goal of diffusion probabilistic modelling is to reconstruct the reverse-time trajectories of the forward diffusion process, and Song et al. (2021) showed that these dynamics can follow two different differential equations: either SDE (8) proposed by Anderson (1982) or ODE (9). So, Appendix C contains these differential equations for $\mathcal{N}(\mu, \Sigma)$ serving as terminal distribution. They depend on time-dependent gradient field $\nabla \log p_{0t}(X_t|X_0)$ supposed to be modelled using neural network. In order to train it, we show how to compute the gradient in Appendix D.

A. Solving forward diffusion SDE

Forward diffusion SDE is given by

$$dX_t = \frac{1}{2}\Sigma^{-1}(\mu - X_t)\beta_t dt + \sqrt{\beta_t}dW_t, \quad t \in [0, T], \tag{19}$$

where X_t is n-dimensional stochastic process, W_t is the standard n-dimensional Brownian motion, $\mu = (\mu_1 ... \mu_n)^{\mathbf{T}}$ is n-dimensional vector, Σ is $n \times n$ diagonal matrix with positive diagonal elements $\{\sigma_{ii}^2\}_1^n$ and noise schedule β_t is non-negative function $[0, T] \to \mathbb{R}^+$. Consider change of variables $Y_t = X_t - \mu$. Then we can rewrite forward diffusion SDE as

$$dY_t = -\frac{1}{2}\Sigma^{-1}Y_t\beta_t dt + \sqrt{\beta_t}dW_t. \tag{20}$$

For every $i = 1, .., n$ we have

$$\begin{aligned} d\left(e^{\frac{1}{2\sigma_{ii}^2}\int_0^t \beta_s ds} Y_t^i\right) &= e^{\frac{1}{2\sigma_{ii}^2}\int_0^t \beta_s ds} \cdot \frac{1}{2\sigma_{ii}^2}\beta_t Y_t^i dt + e^{\frac{1}{2\sigma_{ii}^2}\int_0^t \beta_s ds} \cdot \left(-\frac{1}{2\sigma_{ii}^2}Y_t^i\beta_t dt + \sqrt{\beta_t}dW_t^i\right) = \\ &= e^{\frac{1}{2\sigma_{ii}^2}\int_0^t \beta_s ds}\sqrt{\beta_t}dW_t^i. \end{aligned} \tag{21}$$

Exponential of a diagonal matrix is just element-wise exponential, so we can rewrite it in multidimensional form as

$$d\left(e^{\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}Y_t\right) = \sqrt{\beta_t}e^{\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}dW_t \implies e^{\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}Y_t - Y_0 = \int_0^t \sqrt{\beta_s}e^{\frac{1}{2}\Sigma^{-1}\int_0^s \beta_u du}dW_s, \tag{22}$$

or writing this down in terms of X_t:

$$X_t = e^{-\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}X_0 + \left(I - e^{-\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}\right)\mu + \int_0^t \sqrt{\beta_s}e^{-\frac{1}{2}\Sigma^{-1}\int_s^t \beta_u du}dW_s, \tag{23}$$

where I is $n \times n$ identity matrix.

B. Derivation of conditional distribution of $\mathbf{X_t}$

Let $A(s) = \sqrt{\beta_s}e^{-\frac{1}{2}\Sigma^{-1}\int_s^t \beta_u du}$. It is a diagonal matrix and its i-th diagonal element $a_{ii}(s)$ equals $\sqrt{\beta_s}e^{-\frac{1}{2\sigma_{ii}^2}\int_s^t \beta_u du}$. Assume $a_{ii}(s) \in L_2[0, T]$ for each i. Itô's integral $\int_0^t a_{ii}(s)dW_s^i$ is defined as the limit of integral sums when mesh of partition Δ tends to zero:

$$\begin{aligned}\int_0^t a_{ii}(s)dW_s^i = \lim_{\Delta\to 0}\sum_k a_{ii}(s_k)\Delta W_{s_k}^i \overset{d}{=} \lim_{\Delta\to 0}\mathcal{N}\left(0, \sum_k a_{ii}^2(s_k)\Delta s_k\right) \overset{d}{=} \\ \overset{d}{=} \mathcal{N}\left(0, \lim_{\Delta\to 0}\sum_k a_{ii}^2(s_k)\Delta s_k\right) = \mathcal{N}\left(0, \int_0^t a_{ii}^2(s)ds\right),\end{aligned} \tag{24}$$

where the first equality in distribution holds due to the properties of Brownian motion and the fact that $a_{ii}(s_k)$ are deterministic (implying that $a_{ii}(s_k)\Delta W_{s_k}^i = a_{ii}(s_k)(W_{s_{k+1}}^i - W_{s_k}^i)$ are independent normal random variables with mean 0 and variance $a_{ii}^2(s_k)(s_{k+1}-s_k) = a_{ii}^2(s_k)\Delta s_k$) and the second equality in distribution follows from Lévy's continuity theorem (it is easy to check that the sequence of characteristic functions of random variables on the left-hand side converges point-wise to the characteristic function of the random variable on the right-hand side). Then, simple integration gives

$$\int_0^t a_{ii}^2(s)ds = \int_0^t \beta_s e^{-\frac{1}{\sigma_{ii}^2}\int_s^t \beta_u du}ds = \int_0^t \sigma_{ii}^2 d\left(e^{-\frac{1}{\sigma_{ii}^2}\int_s^t \beta_u du}\right) = \sigma_{ii}^2\left(1 - e^{-\frac{1}{\sigma_{ii}^2}\int_0^t \beta_s ds}\right). \tag{25}$$

It implies that in multidimensional case we have:

$$\int_0^t \sqrt{\beta_s}e^{-\frac{1}{2}\Sigma^{-1}\int_s^t \beta_u du}dW_s = \int_0^t A(s)dW_s \sim \mathcal{N}\left(0, \lambda(\Sigma, t)\right), \quad \lambda(\Sigma,t) = \Sigma\left(I - e^{-\Sigma^{-1}\int_0^t \beta_s ds}\right), \tag{26}$$

and it follows from (23) that

$$Law(X_t|X_0) = \mathcal{N}(\rho(X_0,\Sigma,\mu,t), \lambda(\Sigma,t)), \quad \rho(X_0,\Sigma,\mu,t) = e^{-\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}X_0 + \left(I - e^{-\frac{1}{2}\Sigma^{-1}\int_0^t \beta_s ds}\right)\mu. \tag{27}$$

C. Reverse dynamics

The result by Anderson (1982) implies that if n-dimensional process of the diffusion type X_t satisfies

$$dX_t = f(X_t,t)dt + g(t)dW_t, \quad t\in[0,T], \tag{28}$$

where $g(t)$ is a function $[0,T]\to\mathbb{R}$, then its reverse-time dynamics is given by

$$dX_t = (f(X_t,t) - g^2(t)\nabla\log p_t(X_t))dt + g(t)d\widetilde{W}_t, \quad t\in[0,T], \tag{29}$$

where $p_t(\cdot)$ is the probability density function of random variable X_t and $\widetilde{W}_t$ is the reverse-time standard Brownian motion such that X_t is independent of its past increments $\widetilde{W}_s - \widetilde{W}_t$ for $s<t$. Reverse-time dynamics means that all the integrals associated with reverse-time differentials have t as their lower limit (e.g. dX_t relates to $\int_t^T dX_s = X_T - X_t$). Anderson's result is obtained under the assumption that Kolmogorov equations (for probability density functions) associated with all considered processes have unique smooth solutions. On the other hand, Song et al. (2021) argued that SDE (28) has the same forward Kolmogorov equation as the following ODE:

$$dX_t = (f(X_t,t) - \frac{1}{2}g^2(t)\nabla\log p_t(X_t))dt, \quad t\in[0,T], \tag{30}$$

which means that processes following (28) and (30) are equal in distribution if they start from the same initial distribution $Law(X_0)$. In our case $f(X_t,t) = \frac{1}{2}\Sigma^{-1}(X_t-\mu)\beta_t$ and $g(t) = \sqrt{\beta_t}$, so we have two equivalent reverse diffusion dynamics:

$$dX_t = \left(\frac{1}{2}\Sigma^{-1}(X_t-\mu) - \nabla\log p_t(X_t)\right)\beta_t dt + \sqrt{\beta_t}d\widetilde{W}_t \tag{31}$$

and

$$dX_t = \frac{1}{2}\left(\Sigma^{-1}(X_t - \mu) - \nabla \log p_t(X_t)\right)\beta_t dt, \tag{32}$$

where both differential equations are to be solved backwards.

D. Score estimation

If X_0 is known, then (27) implies that

$$\begin{gathered}\log p_{0t}(X_t|X_0) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\det\lambda(\Sigma,t) - \frac{1}{2}(X_t - \rho(X_0,\Sigma,\mu,t))^{\mathbf{T}}\lambda(\Sigma,t)^{-1}(X_t - \rho(X_0,\Sigma,\mu,t)) \implies \\ \nabla \log p_{0t}(X_t|X_0) = -\lambda(\Sigma,t)^{-1}(X_t - \rho(X_0,\Sigma,\mu,t)),\end{gathered} \tag{33}$$

where $p_{0t}(\cdot|X_0)$ is the probability density function of conditional distribution $Law(X_t|X_0)$. So, if we sample X_t by the formula $X_t = \rho(X_0,\Sigma,\mu,t) + \epsilon_t$ where $\epsilon_t \sim \mathcal{N}(0,\lambda(\Sigma,t))$, then $\nabla \log p_{0t}(X_t|X_0) = -\lambda(\Sigma,t)^{-1}\epsilon_t$. In the simplified case when $\Sigma = I$ we have $\lambda(I,t) = \lambda_t I$ where $\lambda_t = 1 - e^{-\int_0^t \beta_s ds}$. In this case gradient of noisy data log-density reduces to $\nabla \log p_{0t}(X_t|X_0) = -\epsilon_t/\lambda_t$. If $\epsilon_t = \sqrt{\lambda_t}\xi_t$, then we have

$$X_t = \rho(X_0,I,\mu,t) + \sqrt{\lambda_t}\xi_t, \quad \xi_t \sim \mathcal{N}(0,I), \quad \nabla \log p_{0t}(X_t|X_0) = -\xi_t/\sqrt{\lambda_t}. \tag{34}$$