1 Introduction

The aim of the paper is to design the framework to learn the target distribution based on random process.

2 The process of target data

2.1 Geometric Brownian Motion

The stochastic process X_t is said to follow GBM if it satisfying the following SDE

$$dX_t = \mu X_t dt + \sigma X_t dW_t \tag{1}$$

W is the Brownian motion which determine the process from beginning $S_{t=0}$ to $S_{t=T}$.

$$W_k = \sum_{t=1}^{k} b_t, \quad k = 1, \ldots, m \tag{2}$$

where b is the added randomness to the model. which stores a random number coming from the standard normal distribution $N(0, 1)$.

The solution of above SDE has the analytic solution

$$S_k = S_0 \prod_{i=1}^{k} e^{\left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma W(t)} \tag{3}$$

2.2 Ornstein-Uhlenbeck process

The Ornstein-Uhlenbeck process differential equation is given by

$$dX_t = aX_t dt + \sigma dW_t \tag{4}$$

An additional drift term is sometimes added:

$$dX_t = \sigma dW_t + a(\mu - x_t)dt \tag{5}$$

where σ and a is constants and $\{W_t, t \leq 0\}$ is a standard Brownian motion.

$$X_t = e^{at} X_0 + \sigma \int_0^t e^{a(s-t)} dW_s \tag{6}$$

To approximate the numerical solution, we use Euler-Maruyama method to estimate X_t.

Take (5) for example. First, partition the interval $[0, T]$ into N equidistance sunintervals. Then, recursively solving X_{n+1} by

$$X_{n+1} = X_n + a(\mu - x_t) \triangle t + \sigma \triangle dW_s \tag{7}$$

where $\triangle W_n$ is obtained by sampling a random number from normal distribution with expected value zero and variance $\triangle t$.

3 Model

3.1 Generative Adversarial Networks (GANs)

During the training, discriminator D and generator G play the following two-player minimax game during the training. The discriminator takes the input data from real and fake data and calculate the score individually to discriminate whether a data sample comes from real data or generated by generator. The score calcualted by discriminator represents the probability of the input comes from real data. The generator takes the random noise data z as inout. It tries to output the data samples from distribution to fool the discriminator.

The optimizing approach for GAN is based on [1] which is composed with two loss function. One aim to increase the value of $D(s)$ to let the discriminator believe the data s comes from real data. The other one aims to train generator G to maximize the $D(G(z))$.

In the first inner loop of training, it updates the discriminator by ascending its stochastic gradient:

$$\bigtriangledown_{x_d} \frac{1}{m} \sum_{t=1}^{m} \left[\log D(x^{(t)}) + \log \left(1 - D \left(G \left(z^{(t)} \right) \right) \right) \right] \tag{8}$$

After finishing the first inner loop of training, it updates the generator by descending its stochastic gradient with only one step:

$$\nabla_{x_g} \frac{1}{m} \sum_{t=1}^{m} \log \left(1 - D\left(G\left(z^{(t)}\right)\right)\right) \tag{9}$$

where x_d and x_g represents the variables for discriminator and generator, t indicates the time for each vector and $z^{(t)}$ is noise sample. As the value of $D(G(z))$ increase, discriminator evaluate the output of G is real data.

3.1.1 Noise data

Generator takes the random noise data as input data. Then, it captures the distribution of real examples to imitate the data sample from real data. We found out that the initial distribution of noise data for generator influence the performance of GAN.

The results of the GAN's output distribution are shown in Figure 2 and 1. The path simulation is show in Figure 3. The detail of procedure for experiment is in section 3.

3.2 Wassersein GAN

The framework of Wassersein-GAN(WGAN) is provided by [2]. It is an alternative to traditional GAN training. The mode is trained to minimize the Earth-Mover(EM) distance between real distribution and model distribution. The EM distance between distribution of two point processes is:

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \prod(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma} \left[||x - y||\right] \tag{10}$$

Since the distance is continuous and differentiable, it can can be trained until optimally.

4 Empirical results

4.1 The simulation of Geometric Brownian Motion by GAN

Given the real data $\{s(i) \in A\}, i = 1, \ldots, 10\}$ for all $A \subset \mathbb{R}^{10}$. The real data is followed by geometric Brownian motion which is generated by following (3). We implemented two experiments to investigate the effect of noise data for the model distribution. The first experiment takes the independent process as noise data the other one take dependent process. Then, we estimated the probability of the each dimensional data under generators distribution by fitting lognormal distribution to the samples generated with G.

In the first experiment, the generator nets used normal distribution as input to represented independent process. Results are reported in Figure 1. In Figure 1, we compared the real distribution with model distribution for each dimension after training. We observe that the model distribution does not match with real distribution in some dimension.

To compare two distribution statistically, we implemented two-sample AndersonDarling test the hypothesis that two samples are drawn from identical distribution in both experiment. First, we combined data set Z and ordered it ascending. Then, the two-sample Anderson-Darling test statistic is calculated by

$$AD = \frac{1}{mn} \sum_{i=1}^{n+m} \frac{(N_i Z_{(n+m)} - ni)^2}{i Z_{n+m-i}} \tag{11}$$

where n and m is the number of observations in two samples respectively and N_i represents the number of observations in z_n that are equal to or smaller than the ith observation in $Z_{(n+m)}$.

The AD value is over 100 which is larger than the correspondent critical value. Hence, the null hypothesis that two samples come from same distribution is rejected. Therefore, we conclude that the the model distribution does not have the same distribution of geometric brownian motion.

In the second experiment, since the Geometric Brownian Motion is dependent, we assume the discriminator is easily fooled by another dependent process. Thus, we choose Brownian Motion to be the input of generator to see whether it can improve the simulation ability.

Results are reported in Figure 2. The improvement is not significantly. The AD values is also over 100, so we reject the hypothesis that two samples come from same distribution in the experiment.

4.1.1 Simulation Results

Figure 3 plots path simulation. It is easily observe from figure that it occurs model collapse in GAN in two experiments especially in dependent process. Some of paths dominate the generative sampling while training. The generator was be pushed towards producing samples that come from that specific mode, ignoring the other classes most of time. Moreover, if the model distribution is close to the real distribution in some specific dimension, the mode collapse have larger impact on those dimension.

4.2 The simulation of Geometric Brownian Motion by WGAN

5 Conclusion

When the generator in traditional GAN tries to learn the multidimensional random process, it tends to output specific paths to reinforce model distribution match with real distribution.

References

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS14, page 26722680, Cambridge, MA, USA, 2014. MIT Press.

[2] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *ArXiv*, abs/1701.07875, 2017.

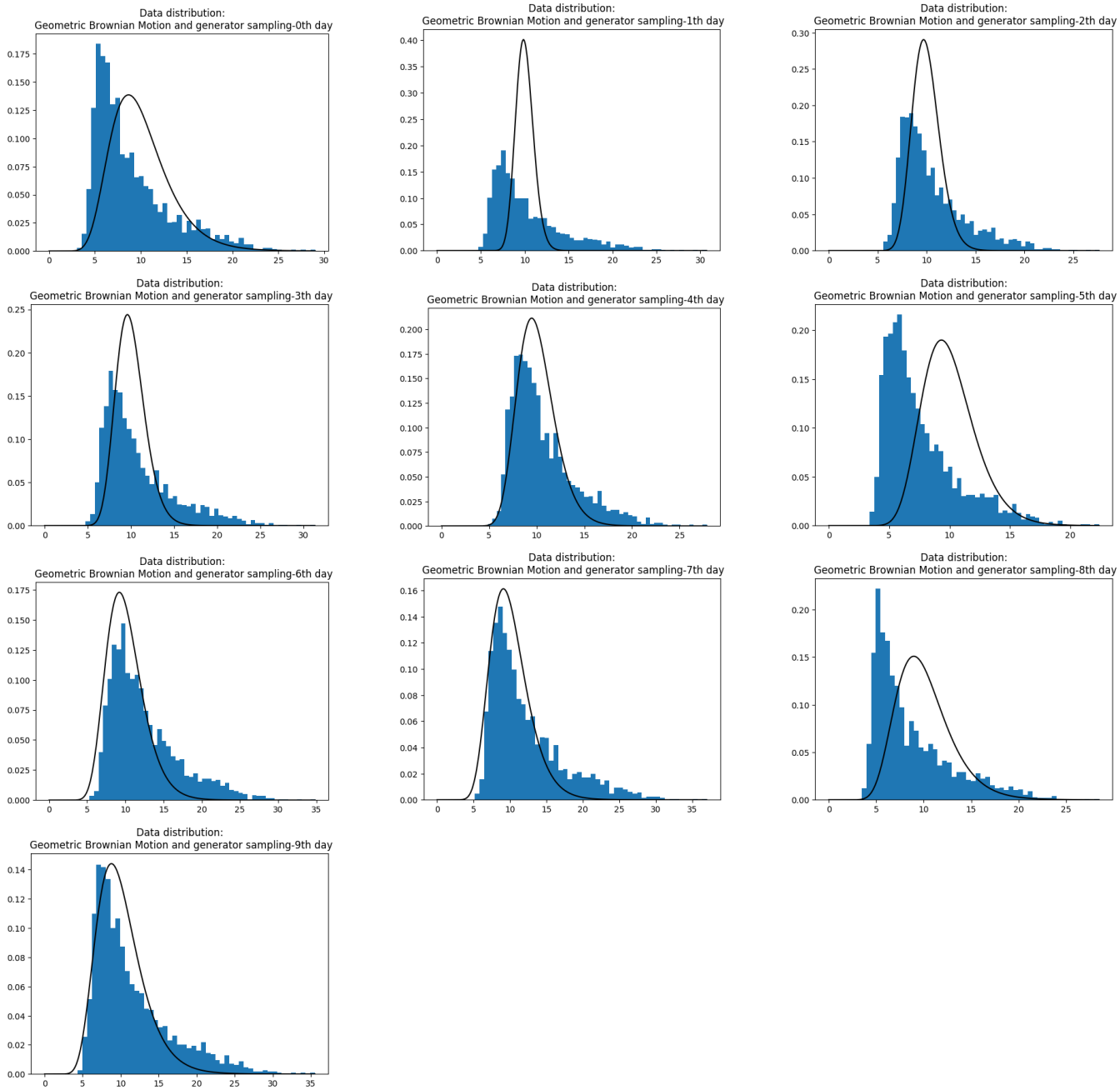


Figure 1: Distribution in specific day with Brownian motion as the input for generator. The histogram is the distribution of sampling generated from generator. The black line is the distribution for the real data in each dimension.

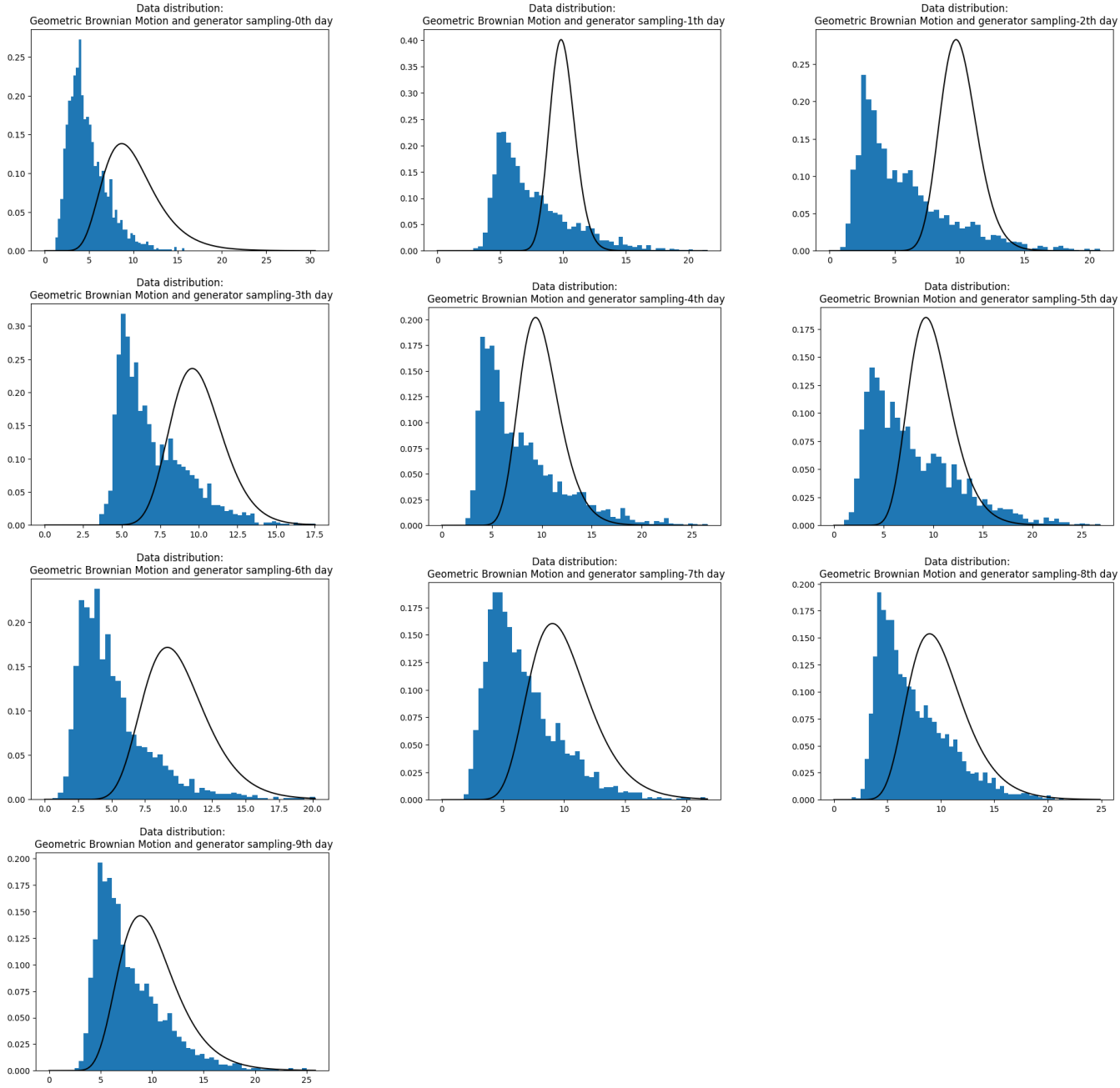
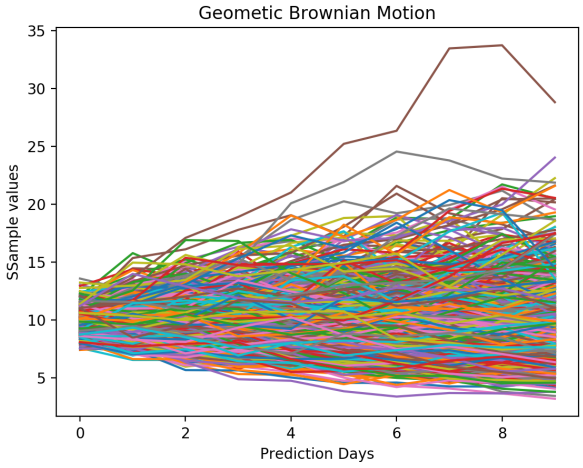
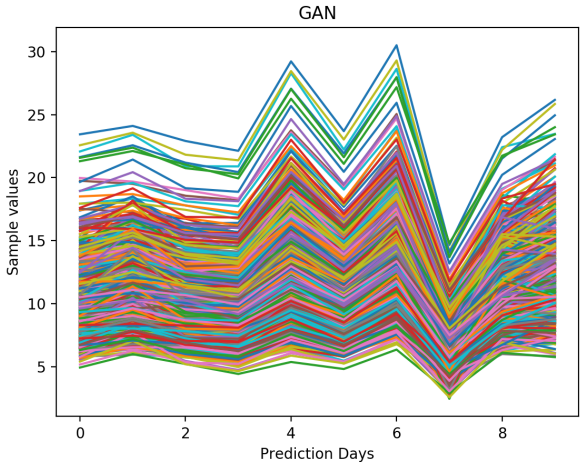


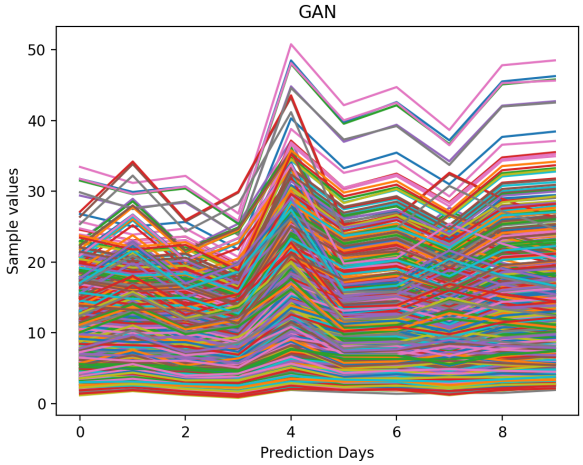
Figure 2: Distribution in specific day with Brownian motion as the input for generator. The histogram is the distribution of sampling generated from generator. The black line is the distribution for real data in each dimension.



(a) Geometric Brownian Motion



(b) Independent noise data: normal distribution



(c) Dependent noise data: Brownian Motion

Figure 3: Path Simulation. It is easily to observe that there is model collapse while training GAN.