CS726 Programming Assignment – 2 Report

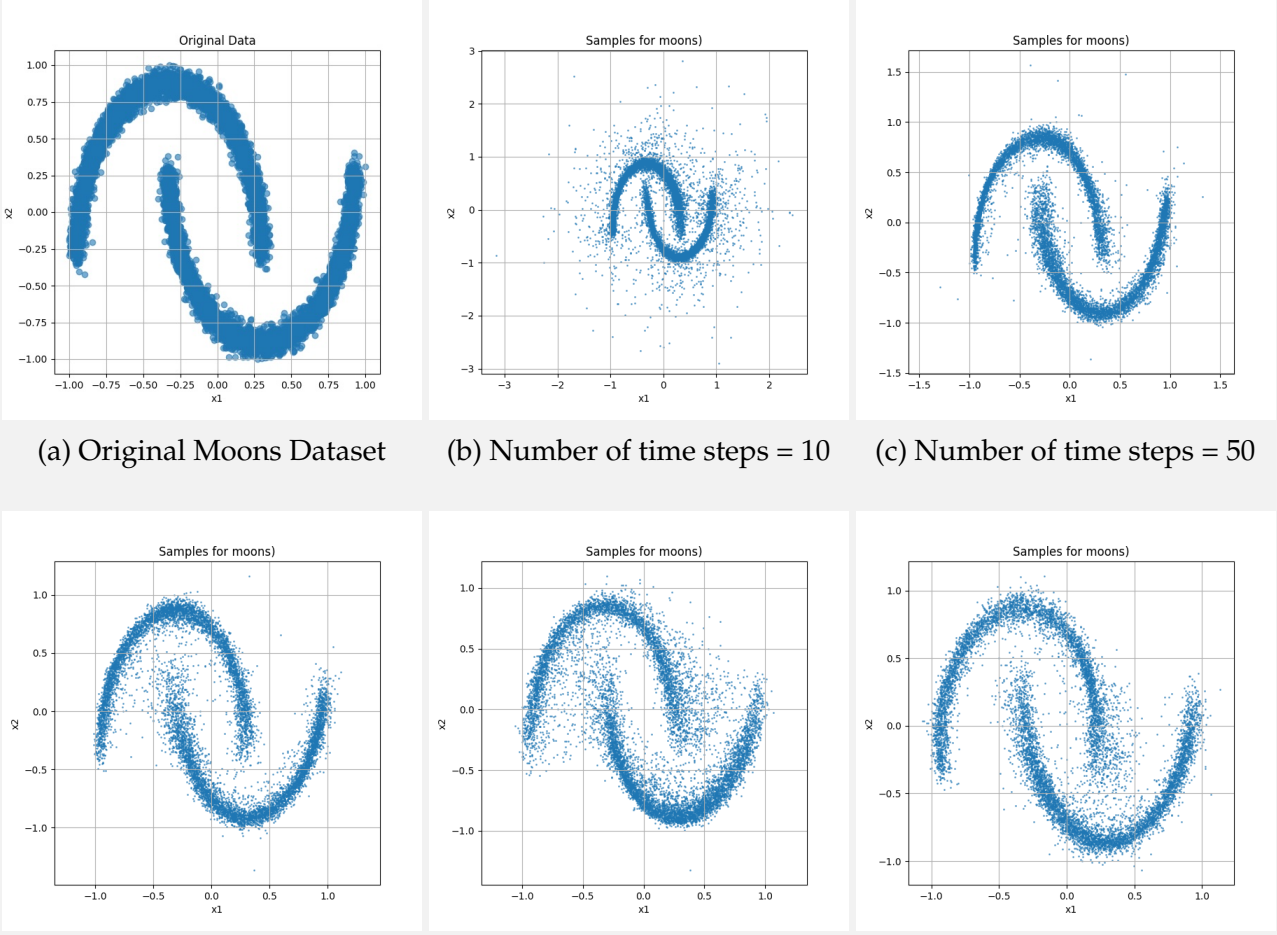
Saksham Rathi (22B1003)
Sharvanee Sonawane (22B0943)
Deeksha Dhiwakar (22B0988)
Department of Computer Science,
Indian Institute of Technology Bombay

Denoising Diffusion Probabilistic Models

Here are the results of unconditional DDPMs on various datasets (with respect to the number of time steps). We had fixed all other parameters (the best settings observed):

- lbeta=0.0001
- ubeta=0.02
- lr=0.0001 (so that training loss decreases across epochs)
- n_samples=10000
- n_dim=2 (for helix it is 3)
- batch_size=128 (to avoid CUDA memory errors and produce optimal results)
- epochs=40

Moons



(a) Original Moons Dataset (b) Number of time steps = 10 (c) Number of time steps = 50

(d) Number of time steps = 100 (e) Number of time steps = 150 (f) Number of time steps = 200

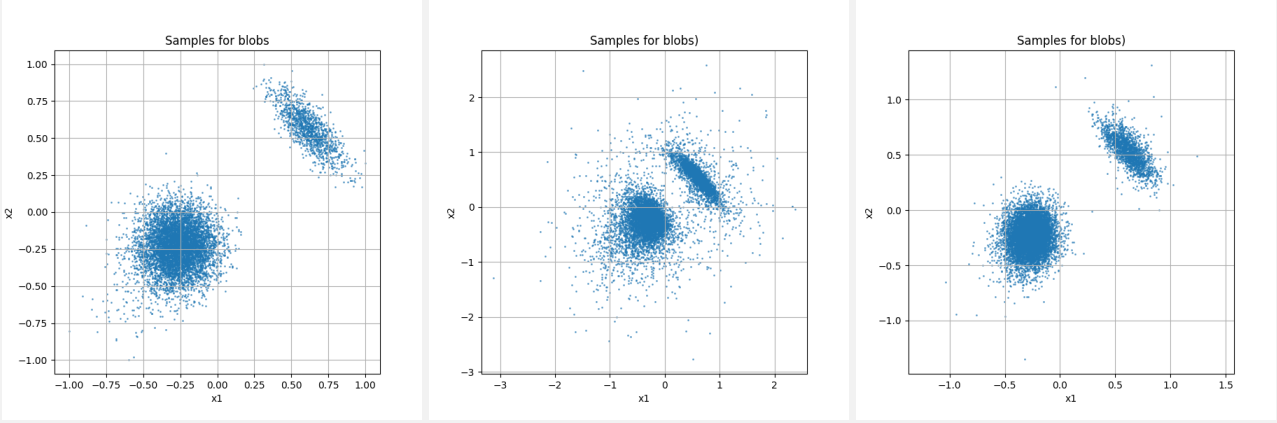
Figure 1: Moons Dataset

Here are the NLL values:

- $T = 10$: 1.048
- $T = 50$: 0.9599
- $T = 100$: 0.9519
- $T = 150$: 0.9218
- $T = 200$: 0.9321

As, we can see from both NLL values and the images, $T = 150$ performed the best.

Blobs



(a) Original Blobs Dataset (b) Number of time steps = 10 (c) Number of time steps = 50

(d) Number of time steps = 100 (e) Number of time steps = 150 (f) Number of time steps = 200

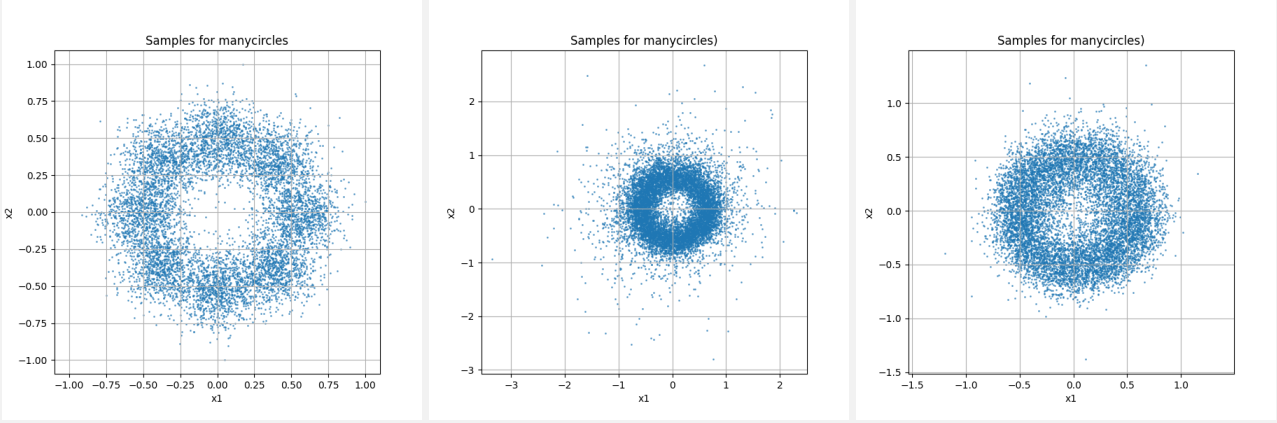
Figure 2: Blobs Dataset

Here are the NLL values:

- $T = 10$: 0.37
- $T = 50$: 0.0152
- $T = 100$: 0.0232
- $T = 150$: -0.0223
- $T = 200$: 0.0045

As, we can see from both NLL values and the images, $T = 150$ performed the best. Moreover, there is a sudden decrease in NLL from 10 to 50, which shows the significant impact of increasing the number of time steps.

Many-Circles



(a) Original ManyCircles Dataset (b) Number of time steps = 10 (c) Number of time steps = 50

(d) Number of time steps = 100 (e) Number of time steps = 150 (f) Number of time steps = 200

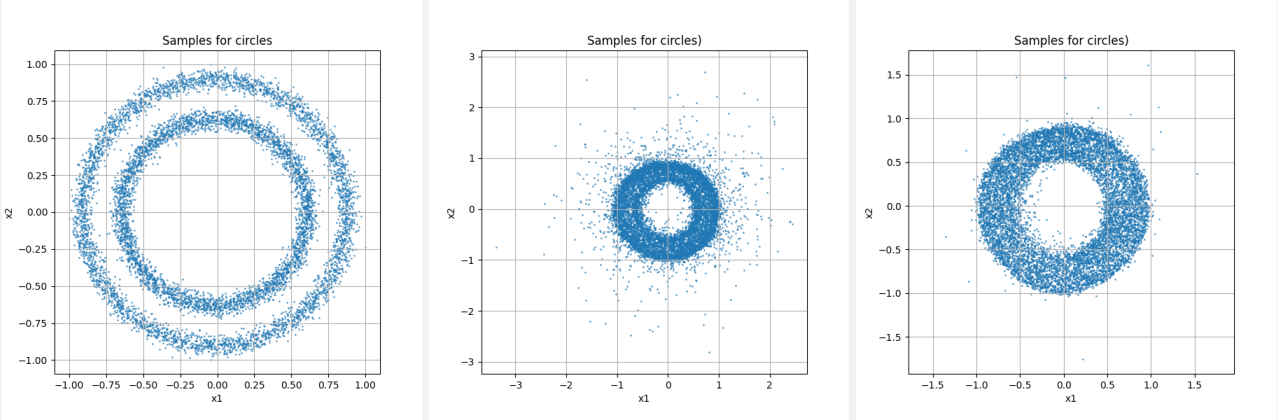
Figure 3: Many Circles Dataset

Here are the NLL values:

- $T = 10$: 0.75
- $T = 50$: 0.548
- $T = 100$: 0.545
- $T = 150$: 0.558
- $T = 200$: 0.522

As, we can see from both NLL values and the images, $T = 200$ performed the best.

Circles



(a) Original Circles Dataset (b) Number of time steps = 10 (c) Number of time steps = 50

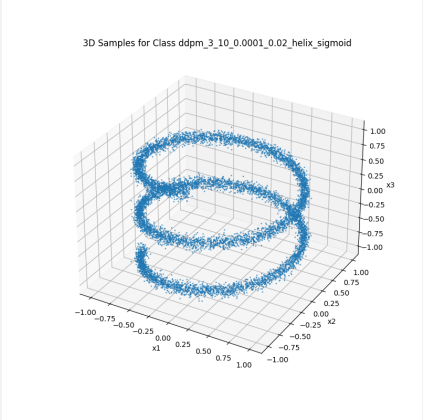
(d) Number of time steps = 100 (e) Number of time steps = 150 (f) Number of time steps = 200

Figure 4: Circles Dataset

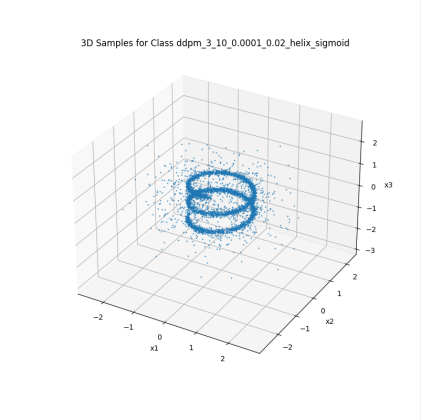
Here are the NLL values:

- $T = 10$: 1.081
- $T = 50$: 0.991
- $T = 100$: 0.9869
- $T = 150$: 1.004
- $T = 200$: 0.992

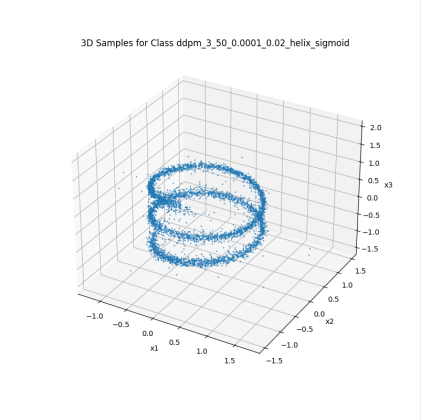
Helix



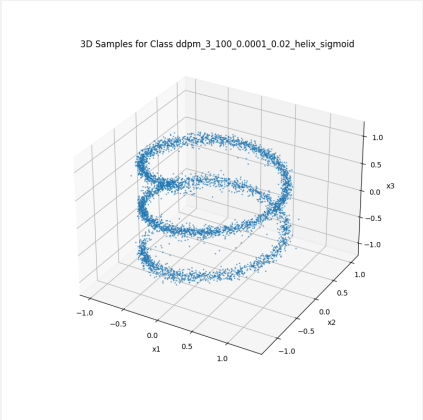
(a) Original Helix Dataset



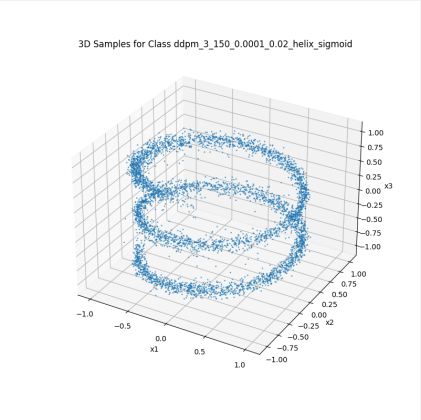
(b) Number of time steps = 10



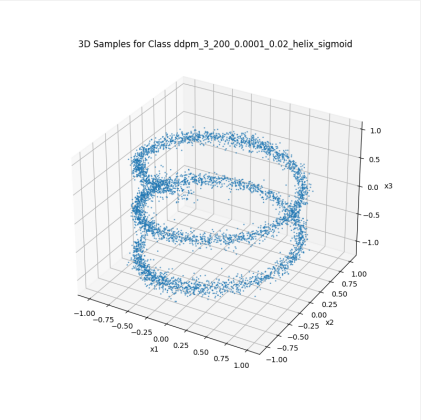
(c) Number of time steps = 50



(d) Number of time steps = 100



(e) Number of time steps = 150



(f) Number of time steps = 200

Figure 5: Helix Dataset

Here are the NLL values:

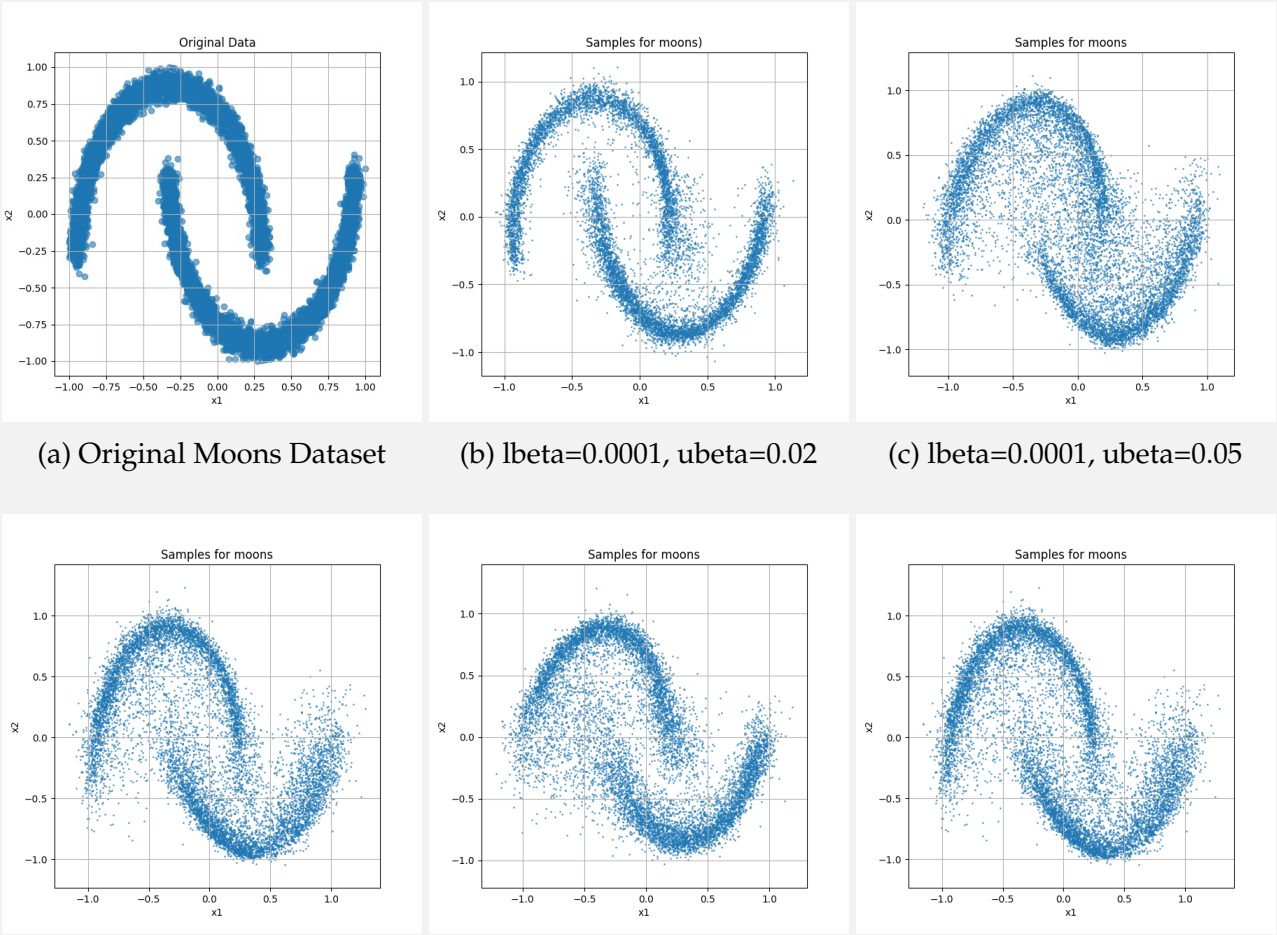
- $T = 10$: 1.6179
- $T = 50$: 1.514
- $T = 100$: 1.5198
- $T = 150$: 1.528
- $T = 200$: 1.528

As we can see from the images (and the NLL values), 50 performs the best.

Noise Schedule Settings

We trained the DDPM for various combinations of ubeta and lbeta values across all the datasets, and selected the best one using NLL value comparison. Here are the NLL values for the moons and blobs datasets:

Moons



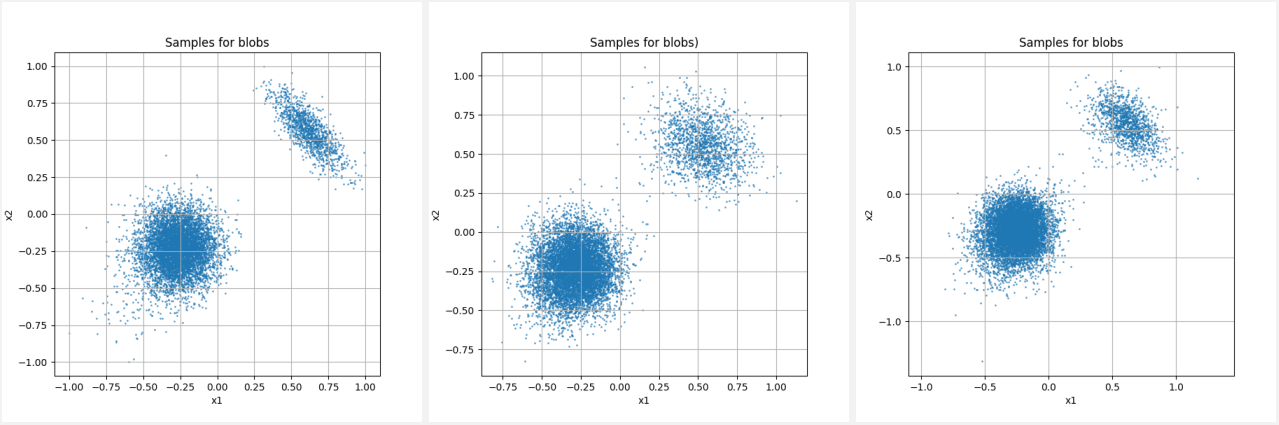
(a) Original Moons Dataset (b) lbeta=0.0001, ubeta=0.02 (c) lbeta=0.0001, ubeta=0.05

(d) lbeta=0.0005, ubeta=0.02 (e) lbeta=0.0005, ubeta=0.05 (f) lbeta=0.001, ubeta=0.05

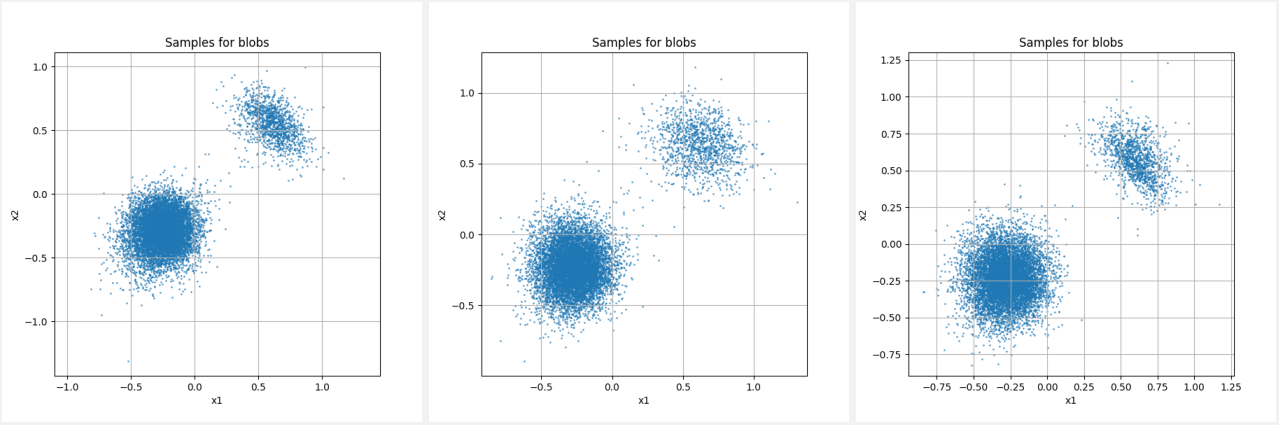
Figure 6: Moons Dataset

- lbeta = 0.0001, ubeta = 0.02: 0.9184
- lbeta = 0.0001, ubeta = 0.05: 0.9223
- lbeta = 0.0005, ubeta = 0.02: 0.9321
- lbeta = 0.0005, ubeta = 0.05: 0.9265
- lbeta = 0.001, ubeta = 0.05: 0.9498

Blobs



(a) Original Blobs Dataset (b) lbeta=0.0001, ubeta=0.02 (c) lbeta=0.0001, ubeta=0.05



(d) lbeta=0.0005, ubeta=0.02 (e) lbeta=0.0005, ubeta=0.05 (f) lbeta=0.001, ubeta=0.05

Figure 7: Moons Dataset

- lbeta = 0.0001, ubeta = 0.02: -0.0145
- lbeta = 0.0001, ubeta = 0.05: -0.0102
- lbeta = 0.0005, ubeta = 0.02: -0.0099
- lbeta = 0.0005, ubeta = 0.05: -0.0104
- lbeta = 0.001, ubeta = 0.05: -0.0098

We observe similar NLL trends for the other datasets as well, leading us to conclude that lbeta=0.0001 and ubeta=0.02 are the best hyperparameters for the noise schedule.

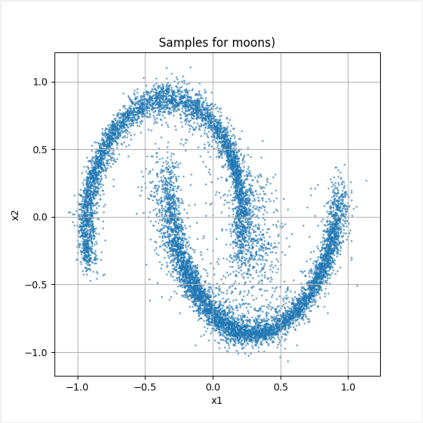
Comparison with Cosine and Sigmoid Noise Schedules

Along with the linear noise schedule, we have studied the effect of cosine and signmoid noise schedules for the DDP model. The results of the comparison are as shown below. In each of the cases we have set the other hyperparameters as follows:

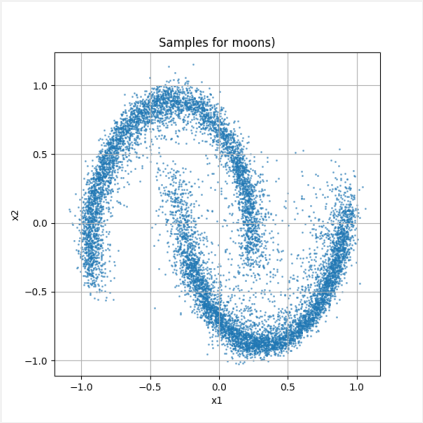
- Number of time steps = 200
- lbeta=0.0001
- ubeta=0.02

- lr=0.0001
- n_samples=10000
- n_dim=2
- batch_size=128
- epochs=40

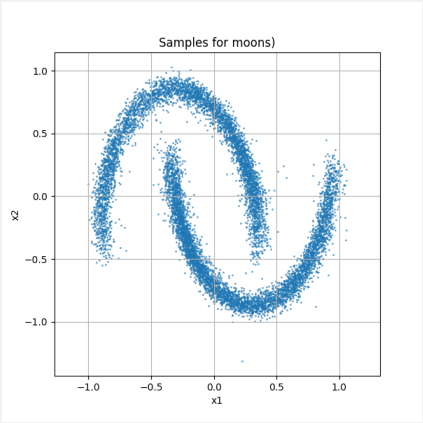
Moons



(a) Linear Noise Schedule
NLL=0.932

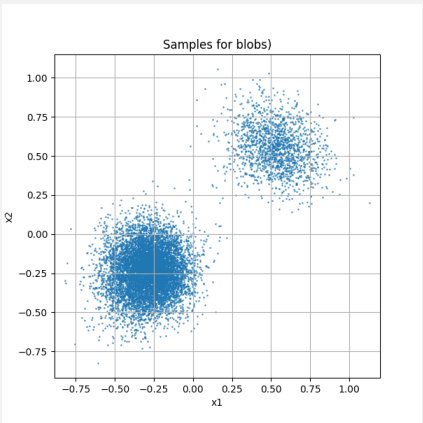


(b) Cosine Noise Schedule
NLL=0.949

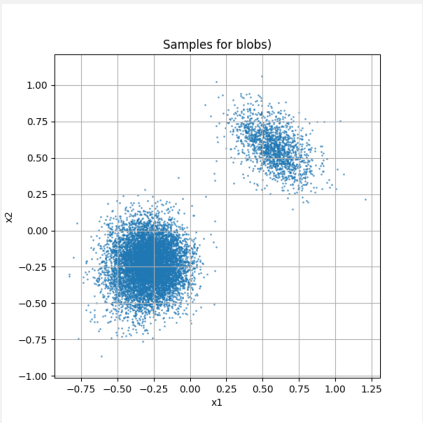


(c) Sigmoid Noise Schedule
NLL=0.928

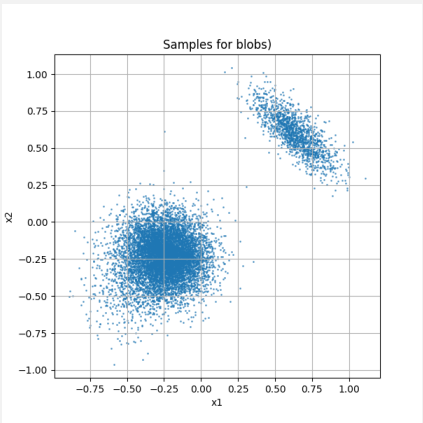
Blobs



(a) Linear Noise Schedule
NLL=0.0045



(b) Cosine Noise Schedule
NLL=0.0043



(c) Sigmoid Noise Schedule
NLL=0.0066

In both cases (as well as on the other datasets) we observe that the NLL is least for the sigmoid noise schedule.

Classifier-Free Guidance

Difference between Guided Sampling and Conditional Sampling

In conditional sampling, we model $p(x|y)$ directly, where y is a conditioning variable (like a class label). During generation, we sample from this conditional distribution to get samples that match the condition.

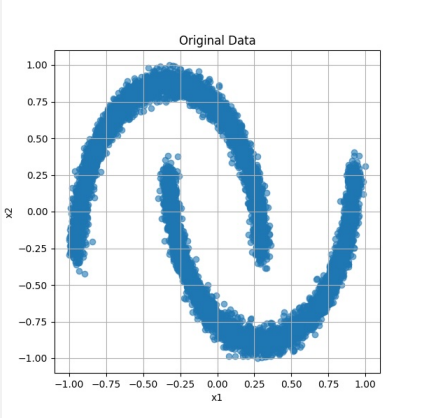
In guided sampling (specifically classifier-free guidance), we train two models: one conditional $p(x|y)$ and one unconditional $p(x)$. During sampling, we interpolate between them with a guidance scale w:

$$\epsilon_\theta(x_t, t, y) = (1 + w) * \epsilon_\theta(x_t, t, y) - w * \epsilon_\theta(x_t, t)$$

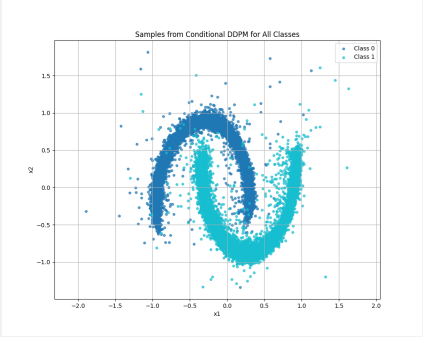
Firstly, guided sampling is more expensive in terms of computation required to train and sample points (almost double, since we are training two models, and using both of them to sample points). However, this guidance increases the impact of the conditioning information and can produce higher quality samples that better match the condition, but with a potential loss of diversity.

Effect of guidance scale

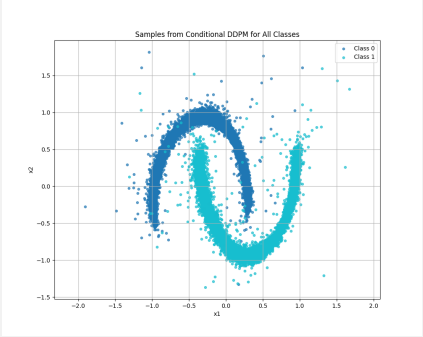
We sampled points using CFG on a variety of guidance scale values, here are the images from the moon dataset:



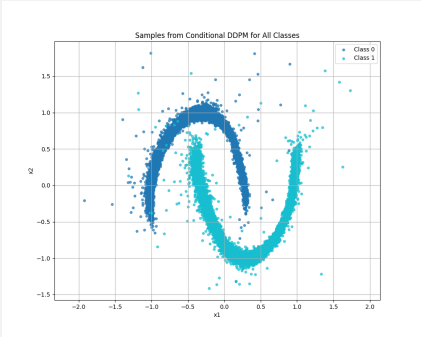
(a) Original Moons Dataset



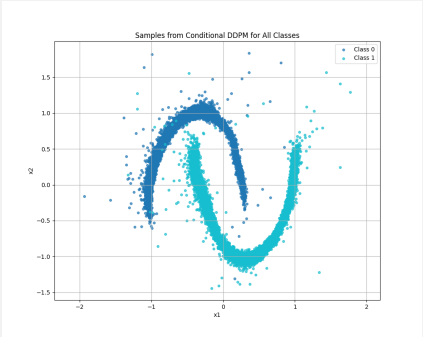
(b) Guidance Scale = 0 (Equivalent to Conditional Sampling)



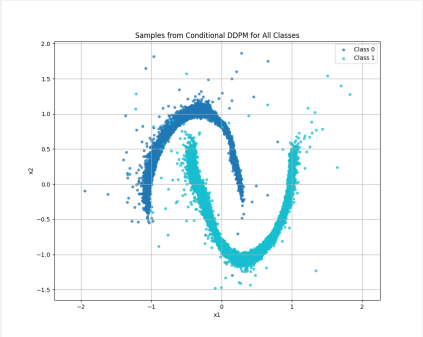
(c) Guidance Scale = 0.2



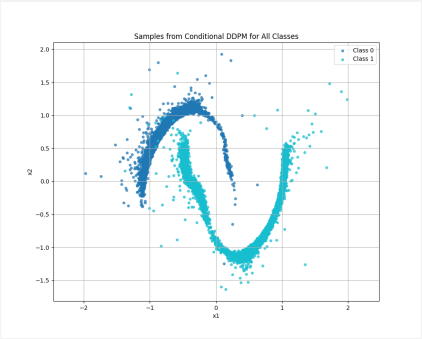
(d) Guidance Scale = 0.5



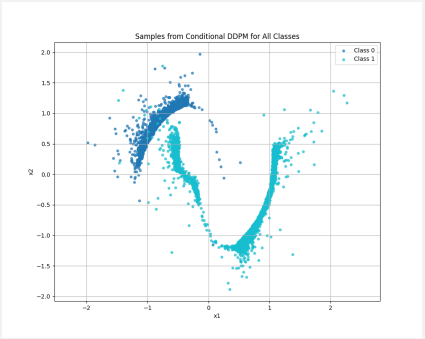
(e) Guidance Scale = 0.7



(f) Guidance Scale = 1.0



(g) Guidance Scale = 2.0



(h) Guidance Scale = 4.0

Figure 10: Moons - CFG

Here are the NLL values for the two class labels:

Guidance Scale	NLL for Class Label 0	NLL for Class Label 1
0.0	0.53	0.58
0.2	0.51	0.58
0.5	0.50	0.59
0.7	0.50	0.60
1.0	0.51	0.62
2.0	0.57	0.66
4.0	0.65	0.71

Table 1: NLL Values for the two class labels of the Moons Dataset

As, it is evident from the NLL values and the images shown above, guidance scales 0.2 and 0.5 perform the best. With higher guidance scales, we typically lose focus on randomly generated data, and rely too much on the training labels provided, thus the NLL values are low, when another sample

is chosen from the actual dataset. These observations match with the ones presented in the CFG paper too.

Reward Guidance