Make new directory: ckpoint; linear_data; nonlinear_data

Running configuration

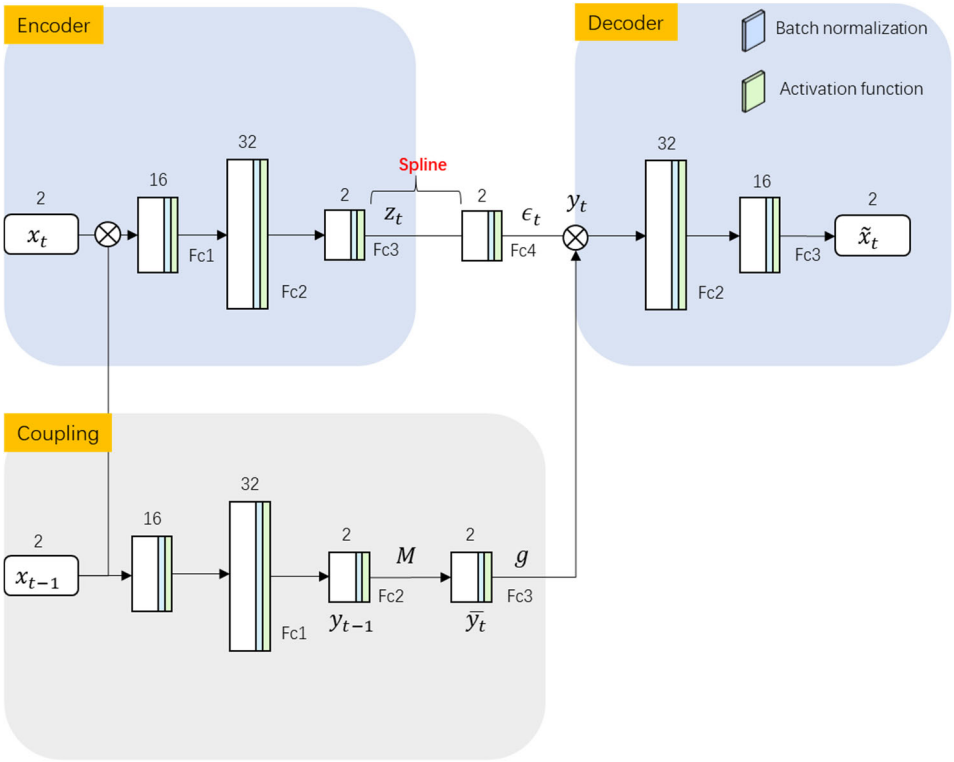
- GPU: Tesla P100
- Cuda version: 11.2
- Packages: torch, tqdm, matplotlib

Data generation process (linear and nonlinear)

- $\begin{cases} y_t^1 = y_{t-1}^1 - 1.5 * y_{t-1}^2 + \epsilon_t^1 \\ y_t^2 = 1.2 * y_{t-1}^2 + \epsilon_t^2 \\ x_t^1 = y_t^1 + y_t^2 \\ x_t^2 = y_t^1 - \sin(y_t^2) \end{cases}$ and $\begin{cases} y_t^1 = y_{t-1}^1 - (y_{t-1}^2)^2 + \epsilon_t^1 \\ y_t^2 = 1.2 * y_{t-1}^2 + \epsilon_t^2 \\ x_t^1 = y_t^1 + y_t^2 \\ x_t^2 = y_t^1 - \sin(y_t^2) \end{cases}$
- batch size = 64, length = 50, chunks = 200

VAE structure:

- framework:



- self.encoder:

```
Sequential(
  (0): Linear(in_features=4, out_features=16, bias=True)
  (1): BatchNorm1d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (2): LeakyReLU(negative_slope=0.01)
  (3): Linear(in_features=16, out_features=32, bias=True)
  (4): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (5): LeakyReLU(negative_slope=0.01)
)
```

- self.decoder:

```
Sequential(
  (0): Linear(in_features=32, out_features=16, bias=True)
  (1): BatchNorm1d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (2): LeakyReLU(negative_slope=0.01)
```

(3): Linear(in_features=16, out_features=2, bias=True)
(4): BatchNorm1d(2, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(5): LeakyReLU(negative_slope=0.01)
)

- self.coupling/ self.f1:

Sequential(
(0): Linear(in_features=2, out_features=16, bias=True)
(1): BatchNorm1d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(2): LeakyReLU(negative_slope=0.01)
(3): Linear(in_features=16, out_features=32, bias=True)
(4): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(5): LeakyReLU(negative_slope=0.01)
(6): Linear(in_features=32, out_features=2, bias=True)
(7): BatchNorm1d(2, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(8): LeakyReLU(negative_slope=0.01)
)

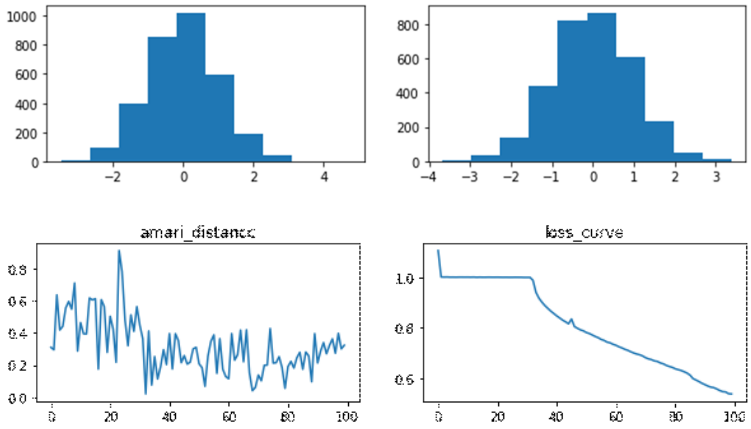
- VAE process

$$\begin{cases} \epsilon_t = h(x_t, x_{t-1}) \\ p(\epsilon_t) = \prod p(\epsilon_t^d) \\ \overline{y_t} = M * y_{t-1} \\ y_t = g(\overline{y_t}) + \epsilon_t \end{cases}$$

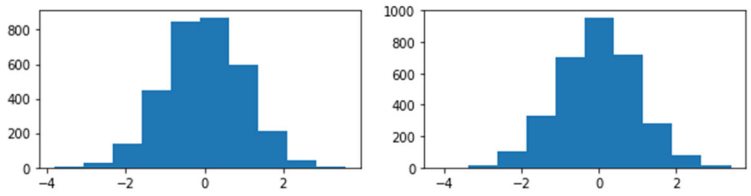
- Loss: $mse(recon, original) + KL(\epsilon_t | x_t, x_{t-1}; normal) + L1(M)$

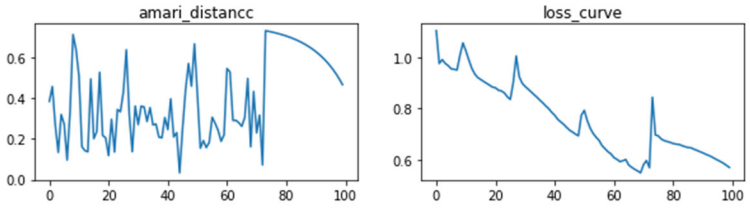
Result:

- VAE-linear

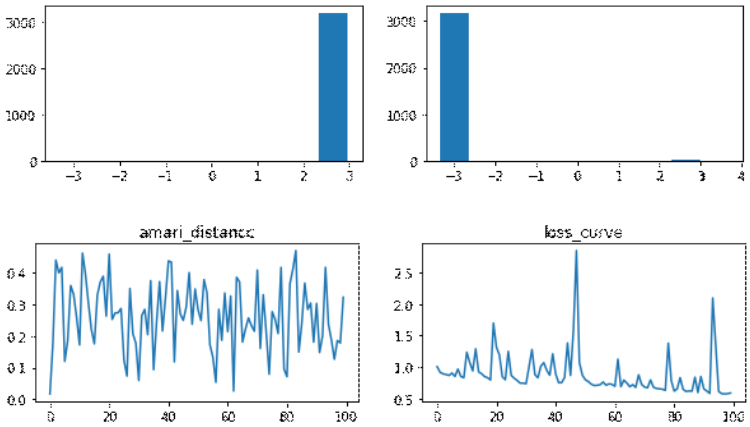


- VAE-nonlinear





- VAE_spline_linear



- VAE_spline_nonlinear

pass