A PATH-SPACE ADVERSARIAL LOSS FOR TIME-SERIES GENERATION

Vassia Kyrieri

July 21, 2022

Abstract

Models describing a molecular system, often remain trapped around some local minimum of the potential energy function or take a long time to switch to another one. In order to solve the metastability problem, we used cumulant generative adversarial networks instead of direct numerical methods. Our goal is to generate transition paths via a deep autoregressive generative model, ie switch from one local minimum to another, and to train the deep autoregressive generative model via Rényi divergence minimization. After training, the model can recognize multiple wells for different potentials.

1 Introduction

One of the known problems of molecular systems occurs when the training model remains trapped for long times in some regions of potential well unable to switch to another one. This behavior is called metastability. Different machine learning techniques have been used to study transition pathways [1]. Our model's goal is to learn to generate samples, whose distribution approximates some unknown target distribution, using generative adversarial networks (GAN)[2]. This model should preserve temporal dynamics, which means that new sequences respect the original relationships between the variables across time. Also, the model's task is to capture the distribution of the state vector within each time point [2].

GANs are capable of drawing new samples from an unknown distribution, when only samples from the distribution are available. We choose GANs because they generate good results, for instance in terms of quality of audio and image generations, even for high-dimensional and complex distributions. A GAN consists of two Neural Networks (NN), known as the generator and the discriminator. The generator model is trained to generate new samples and the discriminator model classifies data as either real or fake. There are three important modules that constitute a GAN: the two NN architectures, the training algorithm and the loss function, which is further divided into the objective functional to be optimized and the function space where the discriminator belongs to [3]. The loss function is based on the cumulant generating function of the distributions

(Cumulant GAN). The optimization problem is equivalent to Rényi divergence minimization, which is a (partially) unified perspective of GAN losses since the Rényi family encompasses Kullback-Leibler divergence (KLD), reverse KLD, Hellinger distance, χ2-divergence and Wasserstein distance.

2 Time-series Generation

In molecular dynamics, many efforts have been devoted to the development of rare events sampling methods, i.e. simulating transition paths linking one metastable state to another [1]. These methods should learn to characterize transition paths and to compute associated transition rates and mean transition times.

2.1 Preliminaries

In molecular dynamics, our goal is to simulate the physical movement of atoms in order to sample the Boltzmann−Gibbs probability measure and the associated trajectories, and to compute macroscopic properties using Monte Carlo estimates [1] . Assume a data set of N observed values of time-series $\mathbf{x} = (x_1, x_2, ..., x_N), x_i \in \mathbf{R}^{dx}$, of a probability measure $p(\cdot)$.
From the chain rule of probability, any joint probability over N variables can be written as the following form:

$$p(\mathbf{x}_{1:N}) = p(x_1)p(x_2|x_1)...p(x_N|x_{N-1}, .., x_1) = \prod_{i=1}^{N} p(x_i|x_{i-1}, ..., x_1). \tag{1}$$

For our data, we use the Markov chain and

$$p(x_i|x_{1:i-1}) = p(x_i|x_{i-1}) \tag{2}$$

which means that x_i depends only on the previous data x_{i-1}, $x_i = g(x_{i-1}, z_i)$, where g is an arbitrary function and z_i some noise [4].

2.2 Overdamped Langevin dynamics

Our real data are generated from a stochastic differential equation and particularly the overdamped Langevin equation. So, consider a diffusion process $(x_t)_{t>=0}$, where $x_t \in \mathbf{R}^{dx}$, and its drift derived by potential $V : \mathbf{R}^{dx} \to \mathbf{R}$. Two potentials are applied with four and three local minima, respectively. The real data are samples from the Boltzmann-Gibbs distribution

$$\mu(dx) = Z^{-1} e^{-\beta V(x)} dx, \tag{3}$$

where $Z = \int\limits_{\mathbb{R}^3} e^{-\beta V(\mathbf{x})} d\mathbf{x}$ is the partition function and $\beta = \frac{1}{K_B T}$, K_B the Boltzmann constant.

The evolution of molecular systems can be modelled by Langevin dynamics, which are stochastic perturbations of the Hamiltonian dynamics and, in this project, we used the Langevin samplers that consist of stochastic perturbations of the overdamped Langevin diffusion [5].

$$dX_t = -\nabla V(X_t)dt + \sqrt{\frac{2}{\beta}} dW_t, \tag{4}$$

where $(W_t)_{t>=0}$ is a standard 2-dimensional Wiener process. The simplest numerical method for solving of SDEs is the stochastic Euler-Maruyama scheme. Considering that the drift of the dynamics is Lipschitz or satisfies Lyapunov conditions and after the discretization we have the following form:]

$$x_i = x_{i-1} - \nabla V(x_{i-1})\delta t + \sqrt{\frac{2\delta t}{\beta}} w_i, \quad \forall i = 1, .., N \tag{5}$$

where $w_i \sim N(0, 1)$ is an independent Gaussian random variable and δt is the time or discretization step.

3 Different Loss Functions

As we mentioned, our goal is to approximate the probability of the real data p_r, i.e. to minimize the "distance" between this probability and the one from the fake data q_θ. So we have to solve a minimization problem:

$$\min_\theta \mathbb{D}(p_r, q_\theta) \tag{6}$$

The original formulation of GANs was to consider that GAN is a game between the generator and the discriminator, the eq.(6) has the minimax objective:

$$\min_\theta \max_D H(G_\theta, D) = \min_\theta \max_D \{\mathbb{E}_{x\sim p_r}[\log(D(\mathbf{x}))] - \mathbb{E}_{x'\sim q_\theta}[\log(1 - D(\mathbf{x}'))]\} \tag{7}$$

As the discriminator tries to maximize the average of the log probability of real samples and the log of the inverse probability for fake samples, the generator wants to minimize the log of the inverse probability predicted by the discriminator for fake samples.

There are several options for the "closeness" quantity $\mathbb{D}(p_r, q_\theta)$, two families are f-divergences(e.g. f-Divergence) and integral probability metrics(e.g. Wasserstein distance of first order).

f-Divergence GAN

The f-divergence is a large class of divergences, such as the Kullback-Leibler divergence, that measures the difference between two given probability distributions, P and Q, with density functions p and q, respectively, and we define it with the following form:

$$D_f(P||Q) = \int q(x) f\left(\frac{p(x)}{q(x)}\right) dx, \tag{8}$$

where the function $f : \mathbb{R}_+ \to \mathbb{R}$ is a convex lower-semicontinuous function satisfying $f(1) = 0$ [6].

Wasserstein GAN

Wasserstein GAN requires that the discriminator must lie within the space of 1-Lipschitz functions, and is constructed using the Kantorovich-Rubinstein duality to obtain

$$\min_\theta \max_{D\in\Gamma} \{\mathbb{E}_{x\sim p_r}[D(x)] - \mathbb{E}_{x'\sim q_\theta}[D(x')]\}, \tag{9}$$

where Γ is the set of 1-Lipschitz functions. The property of the Wasserstein metric, to train a Lipschitz continuous discriminator, is its advantage over f-divergence (such KL divergence) because the training is more stable. Another Wasserstein metric property is the symmetry property where $D_{KL}(P||Q) \neq D_{KL}(Q||P)$. [7] .

Cumulant GAN

As we have already mentioned, the optimization problem in Cumulant GAN is equivalent to Rényi divergence minimization and its advantage pertains to its ability to encompasses various divergences. The Rényi divergence which is applied for the training of our models is defined by

$$\min_{\theta} R_{\alpha}(p_r||q_{\theta}) = \min_{\theta} \frac{1}{\alpha(1-\alpha)} \log\left(\mathbb{E}_q[(\frac{p}{q})^{\alpha}]\right) \tag{10}$$

$$\min_{\theta} R_{\alpha}(p_r||q_{\theta}) = \min_{\theta} \max_{D\in\Gamma'} \{-\frac{1}{\alpha} \log \mathbb{E}_{x\sim p_r}[e^{-\alpha D(x)}] - \frac{1}{1-\alpha} \log \mathbb{E}_{x'\sim q_{\theta}}[e^{(1-\alpha)D(x')}]\}, \tag{11}$$

where Γ' is the set of all measurable and bounded functions [3].

4 Novel Path-space Adversarial Loss

Assume that data consists of N variables of a time-series, where $x_i \in \mathbb{R}^2, x_i = (x_{1,i}, x_{2,i})$. We describe the real data distribution according to Markov chain

$$p(x_{1:N}) = \prod_{i=1}^{N} p(x_i|x_{i-1}) \tag{12}$$

and the generated data according to

$$q(x_{1:N}) = \prod_{i=1}^{N} q(x_i|x_{i-1}). \tag{13}$$

For the discriminator, a straightforward application of a variational formula requires to have the following scheme

$$D(x_{1:N}) = D(x_1, .., x_N). \tag{14}$$

Although, this is not practical as the discriminator takes as input all the variables of the time-series. Using the property:

$$D_{KL}(p(x_{1:N})||q(x_{1:N})) = ND_{KL}(p(x_i||x_{i-1})||q(x_i||x_{i-1}))+O(1) = ND_{KL}(p(x||x')||q(x||x'))+O(1) \tag{15}$$

where D_{KL} is the Kullback-Leibler divergence and x' is the variable of the previous time point. Thus, according to the last expression (eq. (15)), the discriminator takes as input only two consecutive variables.

According to the theory, to learn the generator's distribution $p_z(\mathbf{x}) \sim N(\mathbf{0}, \mathbf{I})$ of the real data, we define a distribution of a set of input noise variables $\mathbf{z}$, $p_z(z)$. After passing the noise to the generator and training both models, the generator G returns fake samples $\mathbf{x}' = (x'_1, x'_2, ...), \mathbf{x}' = G(\mathbf{x}, \mathbf{z})$ [8].

So for the generator, we choose an autoregressive generator

$$x = G(x', z) \tag{16}$$

where x' is the previous variable and the variable z represents the noise whose statistical properties are learned by the generator.

Algorithm

- **for** number of training iterations **do**
  - **for** k steps **do**
    - sample mini-batch of $n < N$ real data $x_1, .., x_n$
    - sample mini-batch of $n < N$ noise examples $z_1, .., z_n$ from $N(\mathbf{0}, \mathbf{I})$ distribution
    - update the discriminator and generator according to

$$\min_{\theta} \max_{D \in \Gamma'} \{-\frac{1}{\alpha} \log \mathbb{E}_{p_r}[e^{-\alpha D(x,x')}] - \frac{1}{1-\alpha} \log \mathbb{E}_{p_\theta}[e^{(1-\alpha)D(x,x')}]\}$$

  end for
  - update the parameters of generator and discriminator

end for

---

5 Demonstrations

We start from an initial condition $x_0 = \mu + \sigma z_0$, where $\mu, \sigma (= (0, 1))$ are constant parameters and z_0 is a normal random variable ($z_0 \sim N(0, 1)$). Then, according to eq.(5) we integrate the dynamic and we have a discrete trajectory $(x_0, x_1, .., x_N)$, where $x_i \in \mathbf{R}^2$ and $N = 100.000$ with $dt = \frac{1}{100}$. We used two potentials:

- First Potential:

$$V(x, y) = \frac{1}{4}x^4 - \frac{2.25}{2}x^2 + \frac{1}{4}y^4 - \frac{1}{2}y^2 \tag{17}$$

- Second Potential:

$$V(x,y) = 3\exp\Big(-x^2 - (y - \frac{1}{3})^2\Big) - 3\exp\Big(-x^2 - (y - \frac{5}{3})^2\Big) - 5\exp\big(-(x-1)^2 - y^2\big) - 0.2x^4) + 0.2(y - \frac{1}{3})^4 \tag{18}$$

The first potential has four wells (located at $(1, 1.5), (-1, 1.5), (1, -1.5) and (-1, -1.5)$) and the second potential has three wells.

Other important details for our models are the training algorithm, Adam's Algorithm, with learning rate 0.0001 and we tested three activation functions, relu, elu and tanh. Also, the parameter α from eq.(11) is equal to 0.5.

For our model, we check:

- the transition path
- the distribution of real and fake data
- how good the generated quantities $\frac{dV(x,y)}{dx}$ and $\frac{dV(x,y)}{dy}$ approximate the exact values

Figures of the first potential

Plot transition:

For the following figures, we have 10.000 iterations. In the first row of the figures are placed the real data and in the second row the generated data.

For each variable x, y, our goal is to have transitions from one local minimum to another.

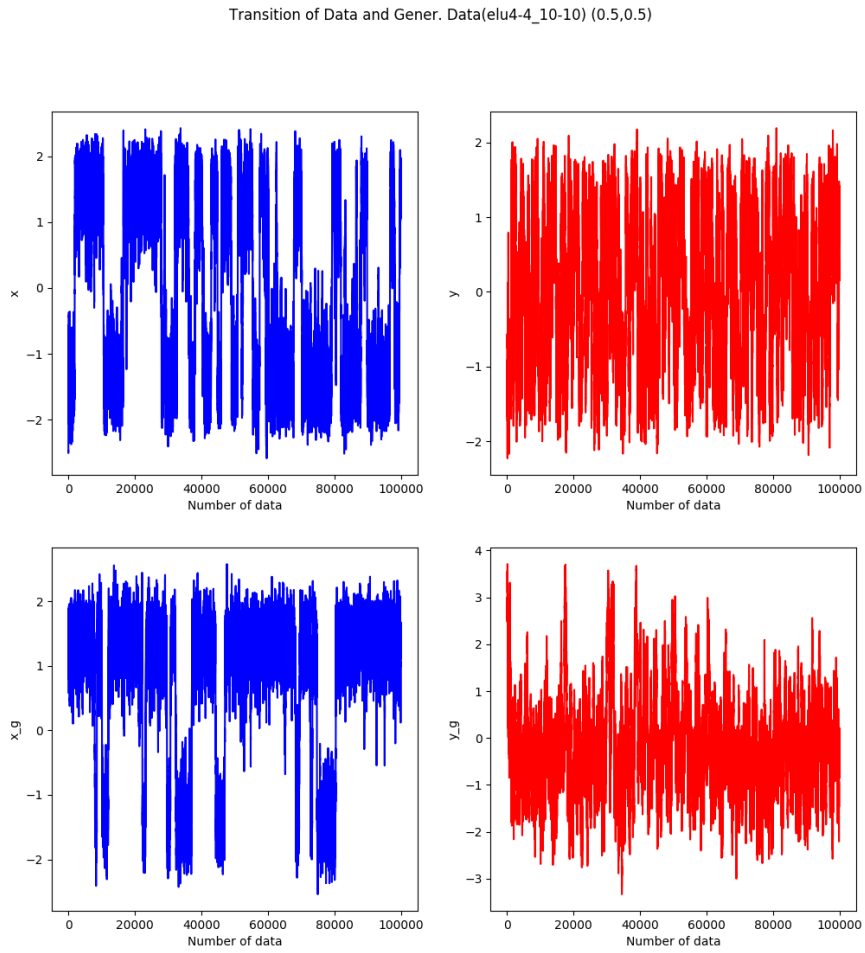

Figure 1: Activation function: elu, Nodes for Generator per layer: 4, Nodes for Discriminator per layer: 10

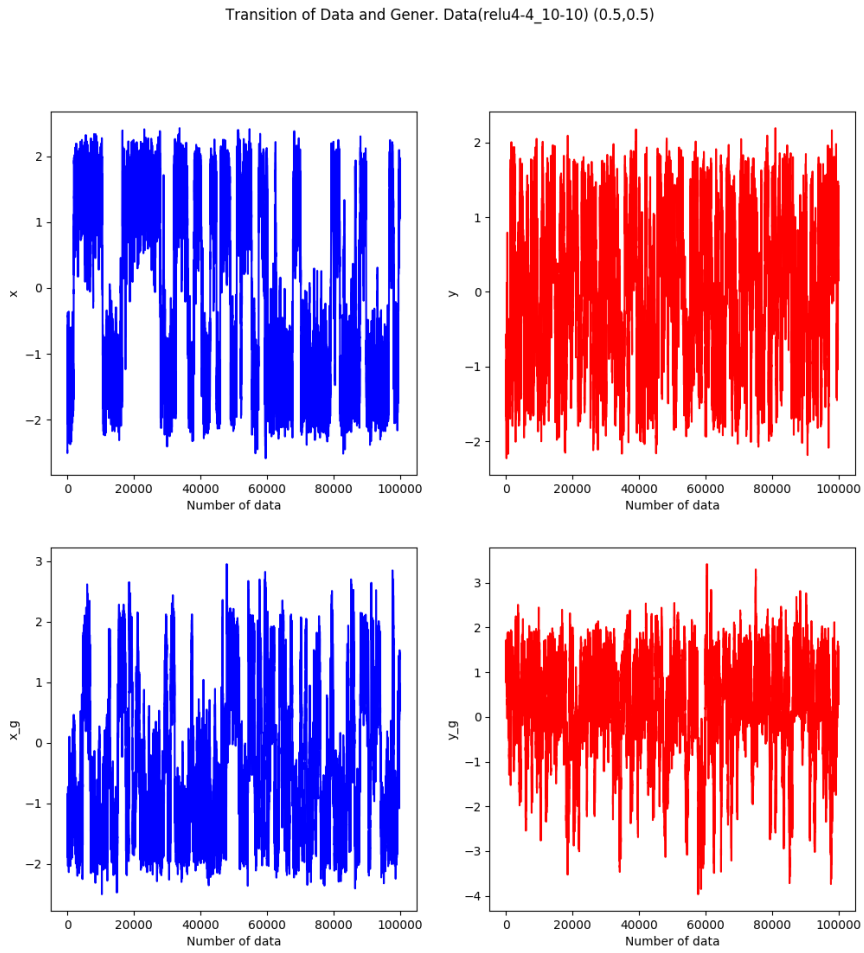

Figure 2: Activation function: relu, Nodes for Generator per layer: 4, Nodes for Discriminator per layer: 10

Plot data:

For the following figures, we have 10.000 epochs. In the first line of the figures are placed the real data and in the second line the generated data.

Our goal here is our model to approximate the four wells.

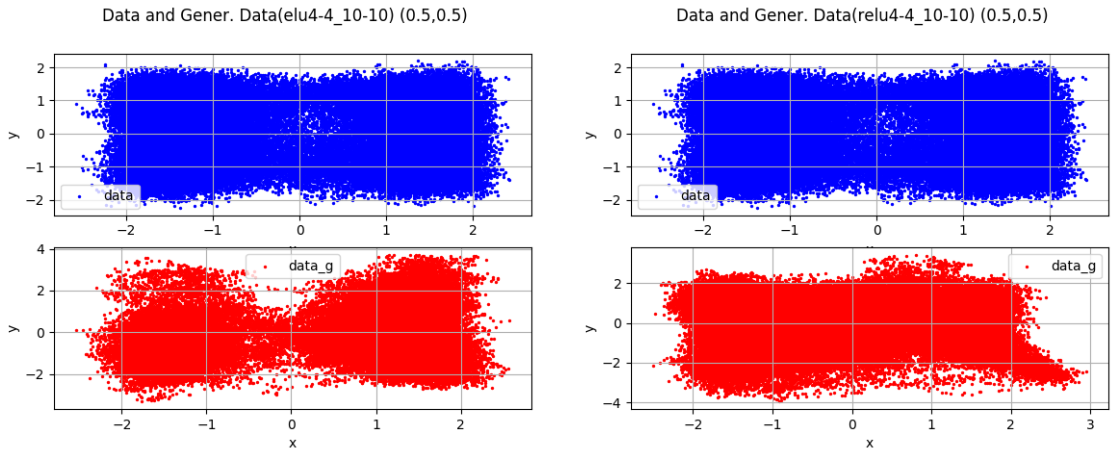

Figure 3: Activation function: elu(left) and relu(right), Nodes for Generator per layer: 4, Nodes for Discriminator per layer: 10

Plot drift:

Also for these figures we have the same model. Here we want to approximate the quantities: $\frac{dV}{dx}$ (first line) and $\frac{dV}{dy}$ (second line).

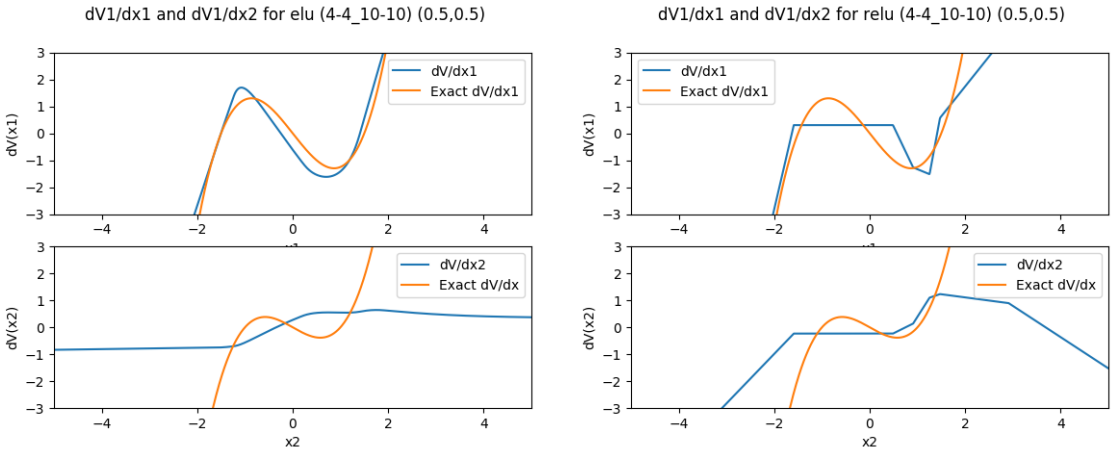

Figure 4: Activation function: elu(left) and relu(right), Nodes for Generator per layer: 4, Nodes for Discriminator per layer: 10

Figures of the second potential

Transition:

For the following figures, we have 50.000 epochs. In the first line of the figures are placed the real data and in the second line the generated data.

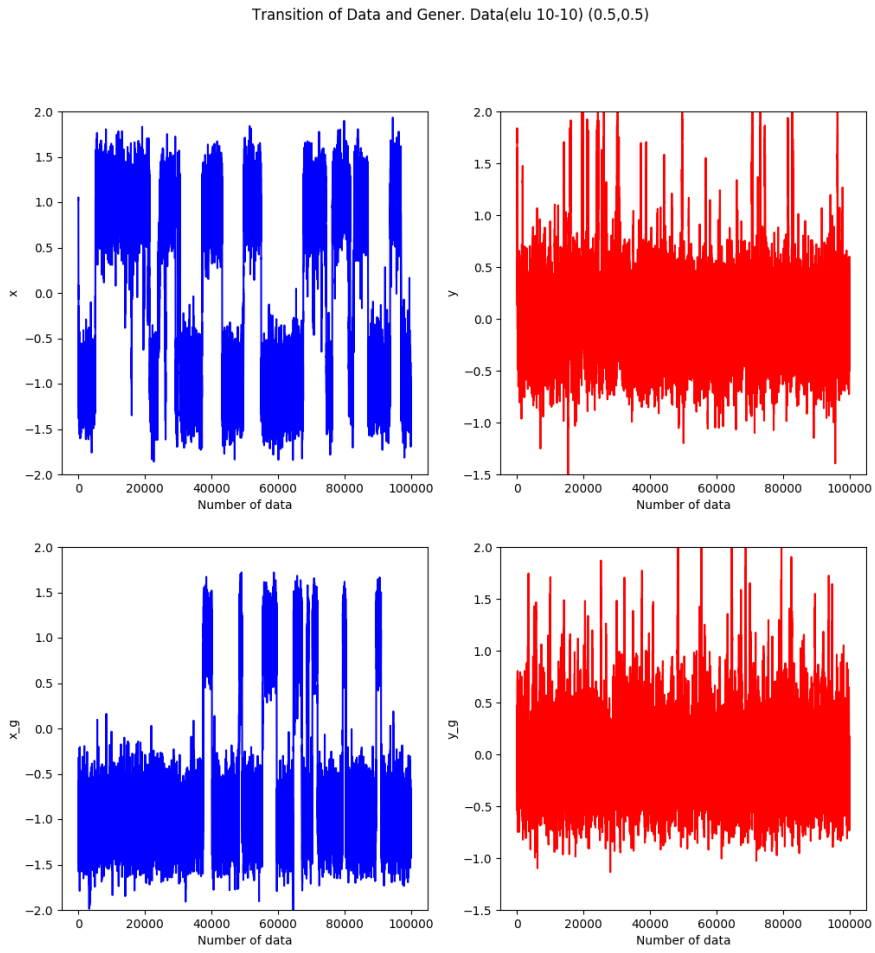

Figure 5: Activation function: elu, Nodes for Generator and Discriminator per layer: 10

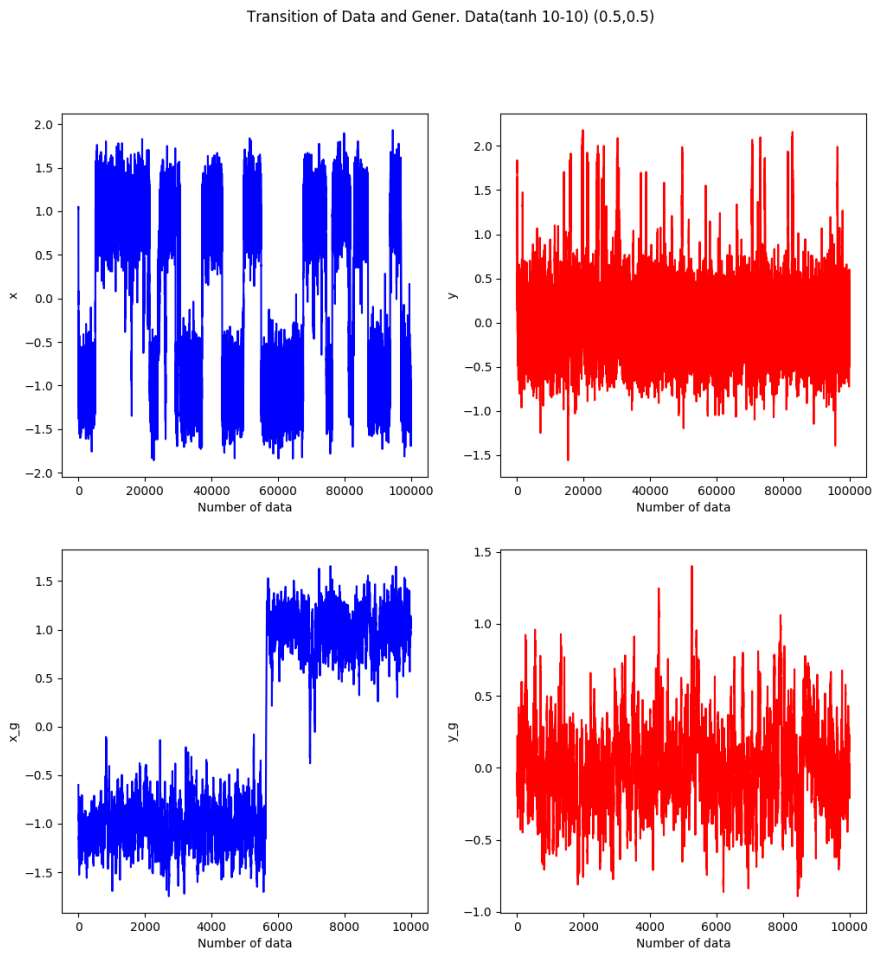

Figure 6: Activation function: tanh, Nodes for Generator and Discriminator per layer: 10

Plot data:

For the following figures, we have 50.000 epochs. In the first line of the figures are placed the real data and in the second line the generated data.

Our goal here is our model to approximate the three wells.

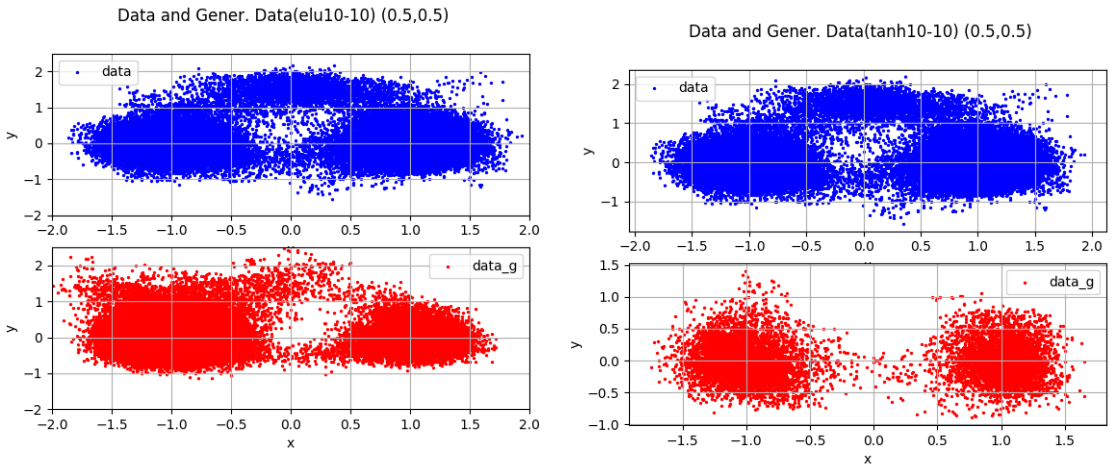

Figure 7: Activation function: elu(left) and tanh(right), Nodes for Generator and Discriminator per layer: 10

6 Conclusions

For the tested molecular systems, the generative models for both potentials approximate the location of the wells relatively well. In particular, the activation function that has the most promising results for both models is the elu function. For the other characteristics of the models, there are differences, such as in architecture-the two NN are not equal for the first potential, and the second potential needs a larger number of iterations to converge.

References

[1] Tony Lelièvre, Geneviève Robin, Inass Sekkat, Gabriel Stoltz, and Gabriel Victorino Cardoso. Generative methods for sampling transition paths in molecular dynamics. *arXiv preprint arXiv:2205.02818*, 2022.

[2] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

[3] Yannis Pantazis, Dipjyoti Paul, Michail Fasoulakis, Yannis Stylianou, and Markos A Katsoulakis. Cumulant gan. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[4] Kevin P Murphy. *Probabilistic machine learning: Advanced Topics*. MIT press, 2022.

[5] Zohren S Lim, B. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379((2194)):20200209, 2021.

[6] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. *Advances in neural information processing systems*, 29, 2016.

[7] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.