Lecture 0: Introductions

Zhongjian Wang*

Abstract

Some live examples of SDE, course introductions

1 Real World models

Here I will list two models that closely relate to our course.

1.1 Diffusion Models in Machine Learning

You may heard of the DALL·E by OpenAI. It is a new AI system that can create realistic images and art from a description in natural language.

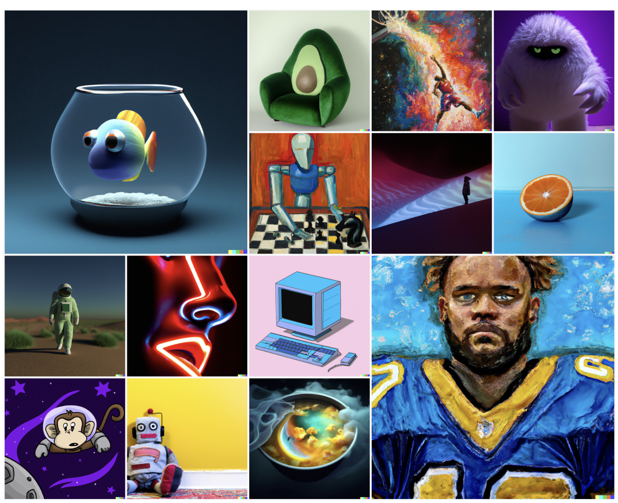

Figure 1: Screenshot from Dalle's website

On Wikipedia, they describe DALL·E as a diffusion model conditioned on CLIP image embeddings, which are generated from CLIP text embeddings by a prior model. Then what is a diffusion model?

*Department of Statistics and CCAM, University of Chicago

Figure 2: Paint by Dalle, with phrase: lecturer giving the first lecture of numerical solution to stochastic differential equation link https://labs.openai.com/s/vGuZsRao4m3n9uiEyqO43xst

In the field of computer vision , the pictures are viewed as a long vectors with values reflecting colors at each point. In the diffusion model, the long vector is viewed as a stochastic process. As time elapses, we start from a picture to a Gaussian random noise. This is called the forward process. The key point is how to reverse the process. In fact

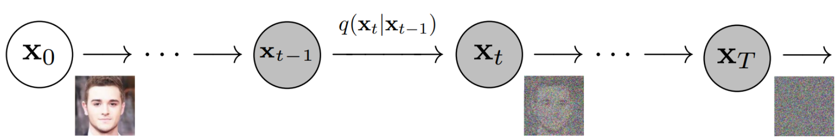

Figure 3: Forward propagation

we can find a 'conditioned' process that going backward in time, that gives you reasonable image from Gaussian noise!

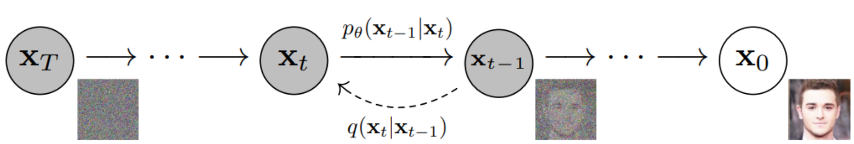

Figure 4: Backward propagation

More on math side The forward process satisfies the following SDE,

$$dX_s = -\frac{1}{2}X_s ds + dW_s, \quad s \in [0, T], \tag{1.1}$$

starting from 'data' distribution $X_0 \sim \mu_0$.

The backward (sampling) process follows the reverse-time SDE,

$$dX_s = -(\frac{1}{2}X_s + \nabla_X \log p(t, X_s))ds + d\tilde{W}_s \tag{1.2}$$

where p is a forward Kolmogorov equation of Eq.(1.1) with initial data distribution μ_0.

Then what does the computer do?

- Find a approximation to p. One of the approach is to solve forward SDE by some solver and apply Feynman Kac formula.
- Solve backward process (1.2) by some SDE solver.

What do an SDE and an SDE solver look like?

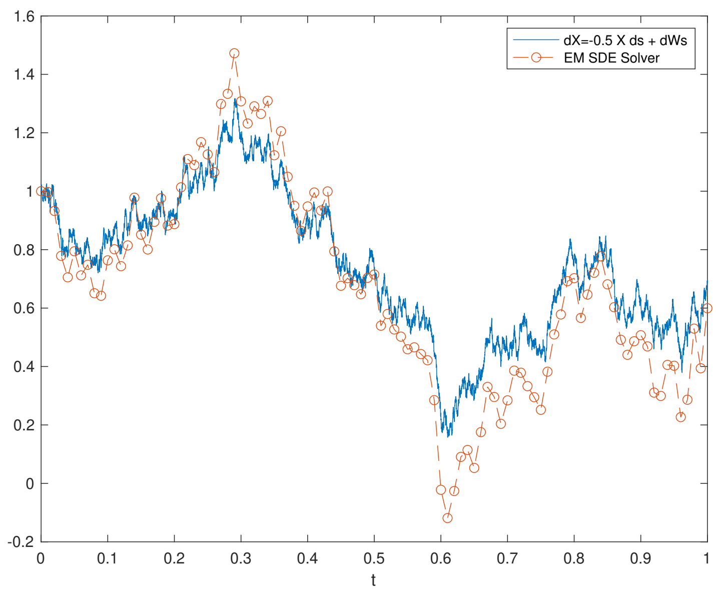

Figure 5: SDE Solver in 1D

Matlab Codes

```
T=1;
Nt=10000;
t=[1:Nt]/Nt*T;
dt=t(2)-t(1);
dw=randn(Nt,1);
x0=1;
x=x0+zeros(Nt+1,1);
for i=1:Nt
x(i+1)=x(i)*exp(-dt/2)+sqrt(1-exp(-dt/2))*dw(i);
end
figure(1)
t=[0,t];
plot(t,x)
hold on
divn=100;
t1=t(1:divn:end);
dt1=t1(2)-t1(1);
nt1=length(t1);
x1=x0+zeros(nt1,1);
for i = 1:nt1-1
x1(i+1)=x1(i)-0.5*x(i)*dt1+sqrt(dt)*sum(dw(i*divn-divn+1:i*divn));
end
plot(t1,x1,'o–')
xlabel('t')
legend('dX=-0.5 X ds + dWs','EM SDE Solver')
hold off
```

1.2 Ant problem

Another example relates to the following post in an infamous company's website.

An ant leaves its anthill in order to forage for food. It moves with the speed of 10cm per second, but it doesn't know where to go, therefore every second it moves randomly 10cm directly north, south, east or west with equal probability.

1. If the food is located on east-west lines 20cm to the north and 20cm to the south, as well as on north-south lines 20cm to the east and 20cm to the west from the anthill, how long will it take the ant to reach it on average?

2. What is the average time the ant will reach food if it is located only on a diagonal line passing through (10cm, 0cm) and (0cm, 10cm) points?

3. Can you write a program that comes up with an estimate of average time to find food for any closed boundary around the anthill? What would be the answer if food is located outside an defined by $((x-2.5cm)/30cm)^2 + ((y-2.5cm)/40cm)^2 < 1$ in coordinate system where the anthill is located at $(x = 0cm, y = 0cm)$? Provide us with a solution rounded to the nearest integer.

By this toy problem, the company wants to test your abilities of coding and understanding of statistics. What will be a real model behind this? Here's something I come up with,

The joint of two stock price (x, y) follow a two dimensional standard Brownian motion after some normalization, which makes it to start at $(0, 0)$ with standard derivation 10 at time $t = 1$.

1. If there is a game (actually an option) that stops when the one of stock price comes out of range $(-20, 20)$. What is the expected time of this game comes to an end?

2. (Same assumption with 1) If the game brings you one dollar when it ends with $x > y$, how do you price this game?

3. What if the closed boundary becomes $((x-2.5)/30)^2 + ((y-2.5)/40)^2 < 1$?

Solution to the first question Discrete case:

Let $A_{i,j}$ denotes the expected time to starting from $X_{i,j}$ get out. We know,

$$\begin{cases} A_{i,j} = \frac{1}{4}(A_{i+1,j} + A_{i-1,j} + A_{i,j+1} + A_{i,j-1}) + 1 \quad \forall i, j = 1 \cdots 3 \\ A_{0,i} = A_{j,0} = A_{4,i} = A_{j,4} = 0 \end{cases} \tag{1.3}$$

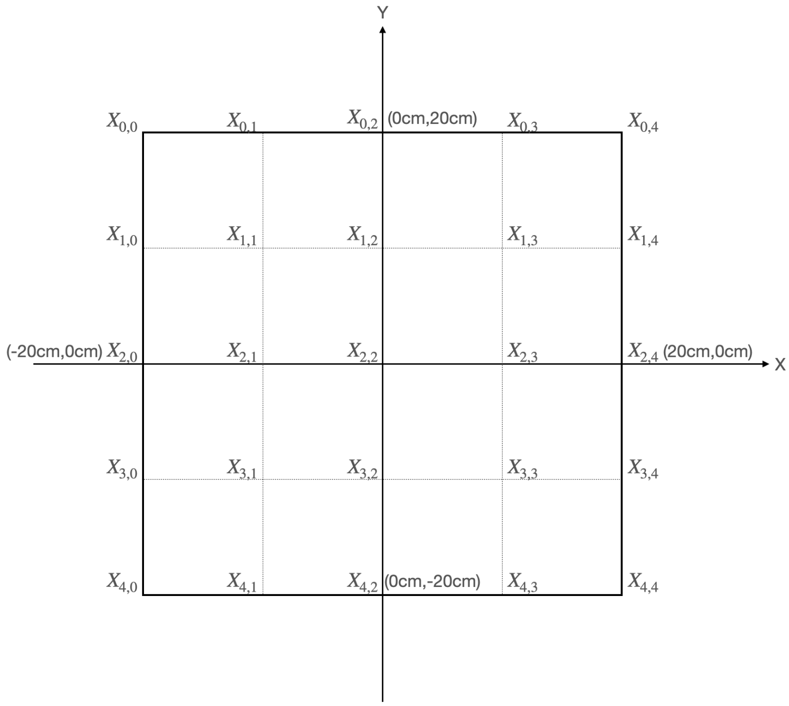

Figure 6: illustration of grid

Continuous case:

$X = (x, y)$ follows 2D SDE,

$$dX = \frac{10}{\sqrt{2}} dW \text{ within domain } D \tag{1.4}$$

$f(x)$ be the exiting time starting at position x. Let

$$u(t, x) = E[t + f(X_t) | X_0 = x, X_t \in D]. \tag{1.5}$$

by law of total expectation, $u(t, x) = u(0, x) = f(x)$.

On the other side, by Feynman Kac formula,

$$\begin{cases} \frac{d}{dt} u = 1 + \frac{1}{2}(\frac{10}{\sqrt{2}})^2 (u_{xx} + u_{yy}) \\ u|_{\partial D} = 0 \end{cases} \tag{1.6}$$

If we apply finite element on mesh grids, $U_{i,j} = f(10 \times i - 20, 10 \times j - 20)$, we have

$$\begin{cases} U_{i,j} = \frac{1}{4}(U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1}) + 1 \quad \forall i, j = 1 \cdots 3 \\ U_{0,i} = U_{j,0} = U_{4,i} = U_{j,4} = 0 \end{cases} \tag{1.7}$$

Another way? We can apply SDE solver for (1.4) and take expectations over all samples!

2 About our course

- The final output of above examples are directly given by solving stochastic differential equations by numerical schemes.
- We are not aiming at math (SDE) formulation of these complicate models, but after taking the course, you will be able to solve the SDE behind the models!

You can get Matlab under UChicago license from the underlining website
https://www.mathworks.com/academia/tah-portal/university-of-chicago-719588.html
I suggest you to install full package or at least include the statistics toolbox when installing. Again, you are free to use any programming language.