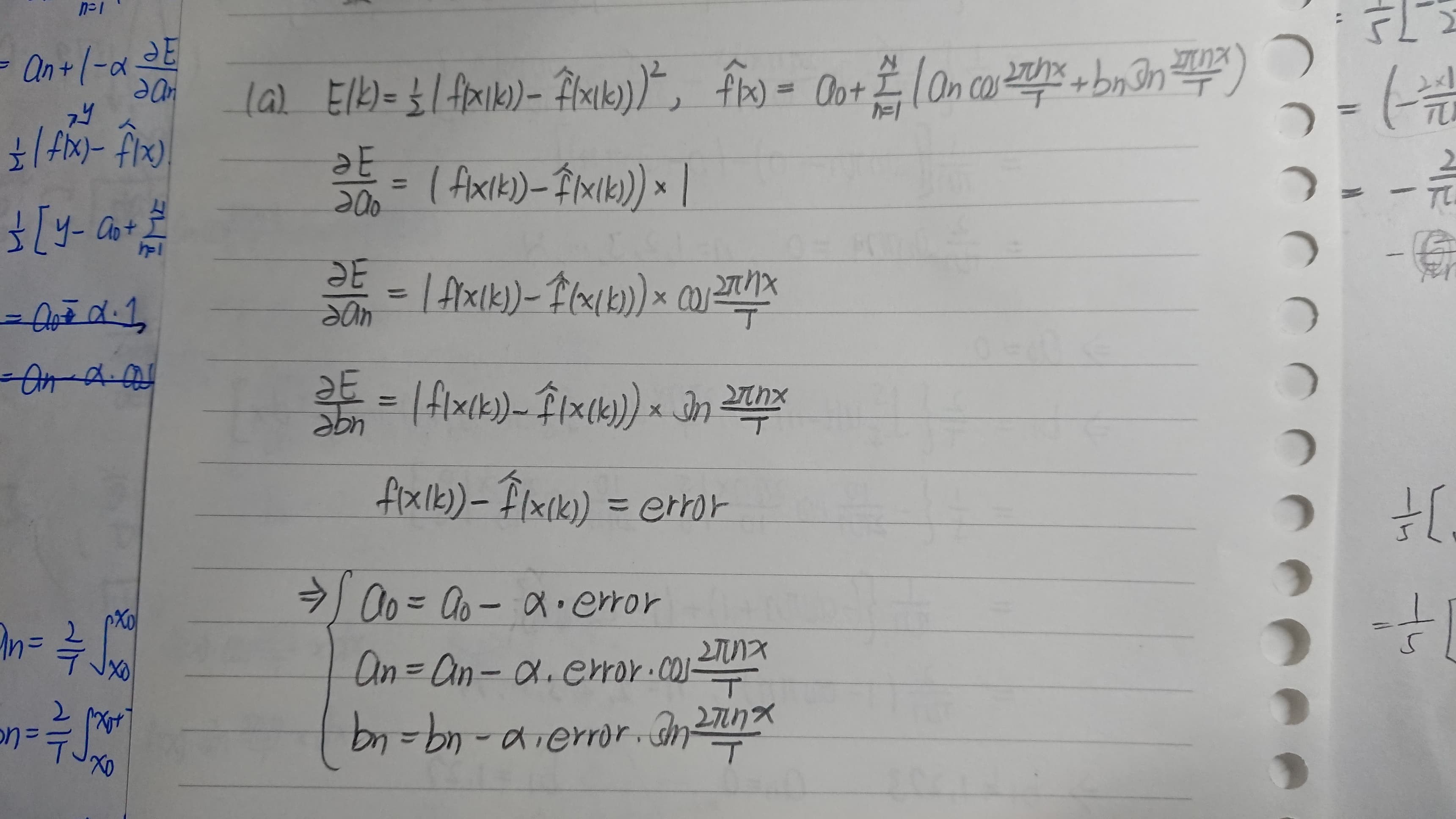
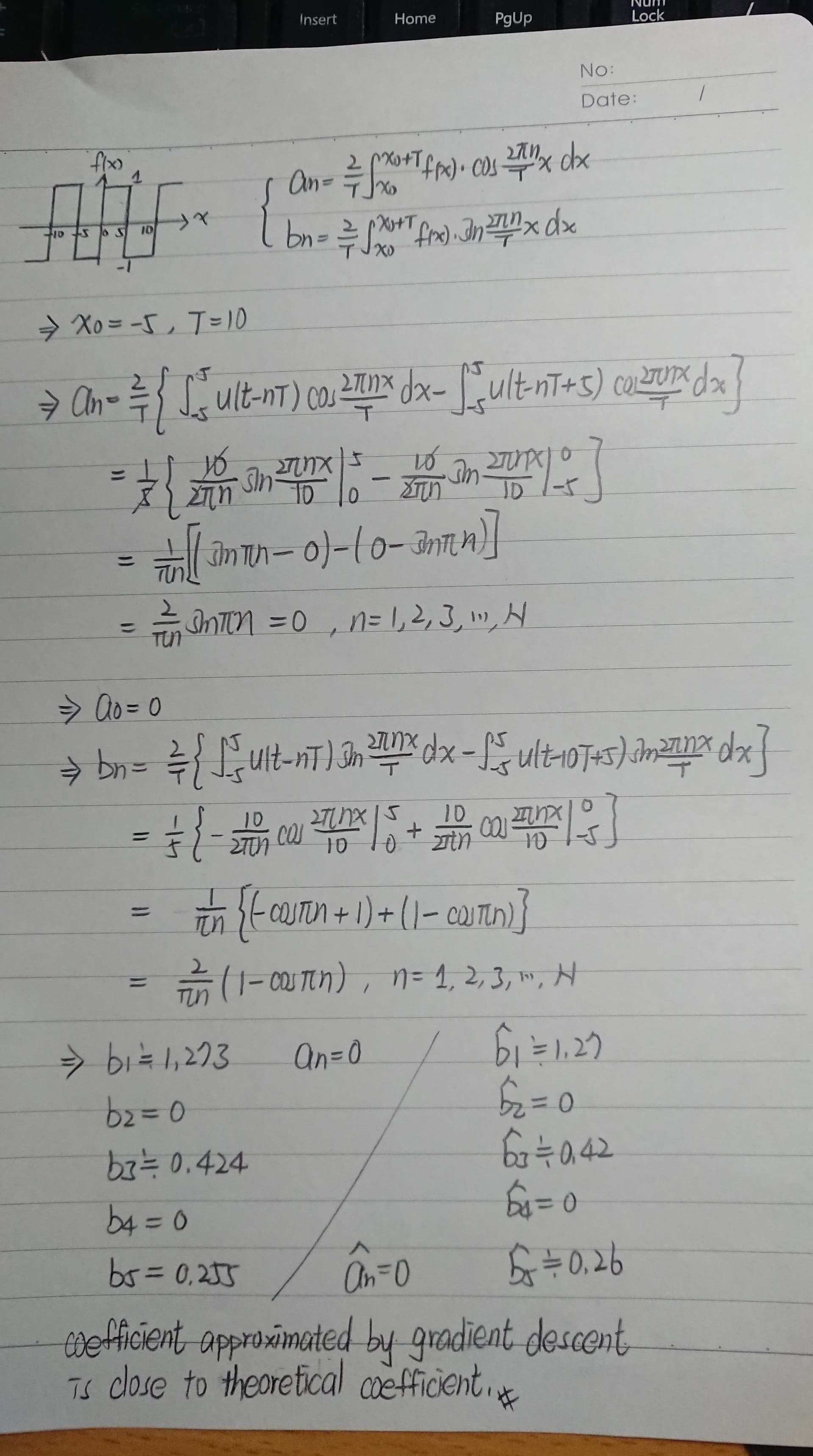
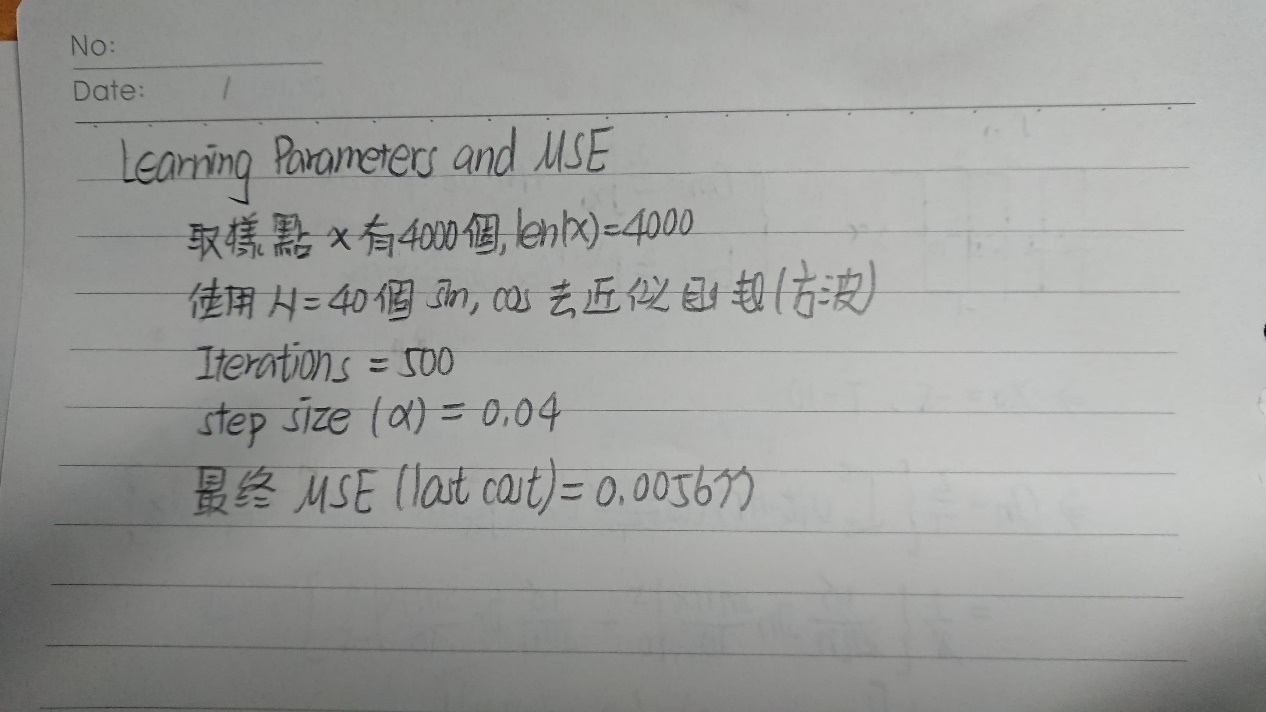
PART A

According the gradient descent method, please derive the update laws for *n a*ˆ and *bn* ˆ , *n*=0,1,….



B0 = 0

PART B



PART C

Give a detailed discussion for *order N* vs. *mean square error:*

傅立葉級數是把類似波的函數表示成簡單正弦波的方式，這裡的N指的是使用了幾組正弦與餘弦函數去表示目標函數。根據傅立葉級數的理論，只要N夠大，就可以近似任何你想要的函數，也越精準。

在這個前提下，我可以假設，如果gradient descent計算傅立葉係數的結果有收斂到理論值，那麼只要N越大，使用傅立葉級數去近似目標函數的MSE就會越小。

經由假設，我實際去改變N的值，慢慢變大，所得出來的最終MSE是越來越小的，符合我剛剛的假設。綜合理論以及實驗結果我可以得出一個結論，N越大時，模擬函數與目標函數之間的MSE越小。

不過傅立葉級數有一個缺點，當目標函數有很多不連續點時，例如這次我所使用的方波，就會使得不連續點位置的函數無法完美的模擬出來。這個現象稱為Gibbs phenomenon(吉布斯現象)。吉布斯現象可以使用σ近似來阻止。

PART D

Give a detailed discussion for *learning rate* vs. *mean square error:*

根據這兩次作業實作gradient descent的經驗，可以知道通常當learning rate越大，MSE就會越快變小。這邊說的是通常兩個字，因為在我們使用的權重更新原則中，並沒有去限制步伐大小，也沒有適時地調整learning rate，因此當選擇的learning rate過大時，很容易造成權重無法收斂的結果。以圖像來說明的話，一個簡單的二次曲線有一個谷底，當learning rate過大時，會無法使權重收斂到谷底的位置，反而會越跑越遠，造成MSE越來越大最終發散。

結論就是，在不更改更新準則的情況下，learning rate的選擇會決定權重是否能夠收斂，在這次的作業中，權重就是我的An以及Bn。若要改變更新準則，則可以乘上抑制步伐大小的係數(和權重有關)，或者是使用adapted learning rate。

PART E

Give a brief discussion for pattern learning and batch learning with the same learning rate:

Batch learning也就是批次學習，Batch size決定每一個iteration要輸入多少data和相對應的output。與之對應的就是pattern learning，也就是單次學習，一次輸入一個data和一個output。

批次學習的優點是可以更準確地往極值的方向前進，和單次學習比較，也不需要太多的iteration，因為他可以更好的代表全體資料。

批次學習的運算速度更快(減少iteration次數)，當我們選定一個batch的大小後，將會以batch的大小將資料輸入深度學習的網路中，然後計算這個batch的所有樣本的平均損失。

單次學習就是Batch size=1的批次學習，每次修正方向以各自樣本的梯度方向修正，橫衝直撞各自為政，難以達到收斂。

P.S.這兩次作業使用的都是全批次學習，一次性把所有要訓練的data全部丟進去學習，缺點是當資料量很大的時候運算速度會下降。

P.S.

（1）batch size：批大小。在深度學習中，一般採用SGD訓練，即每次訓練在訓練集中取batch size個樣本訓練；

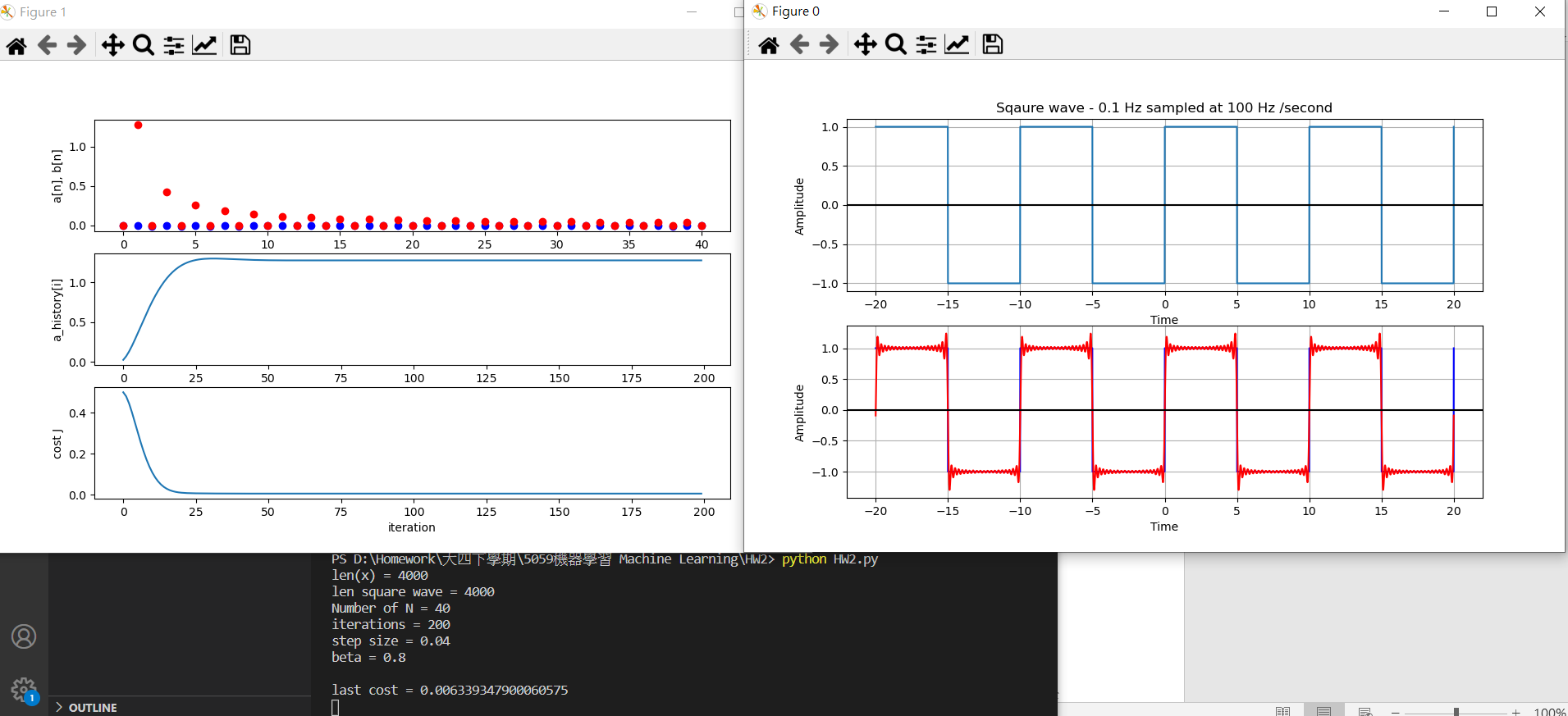
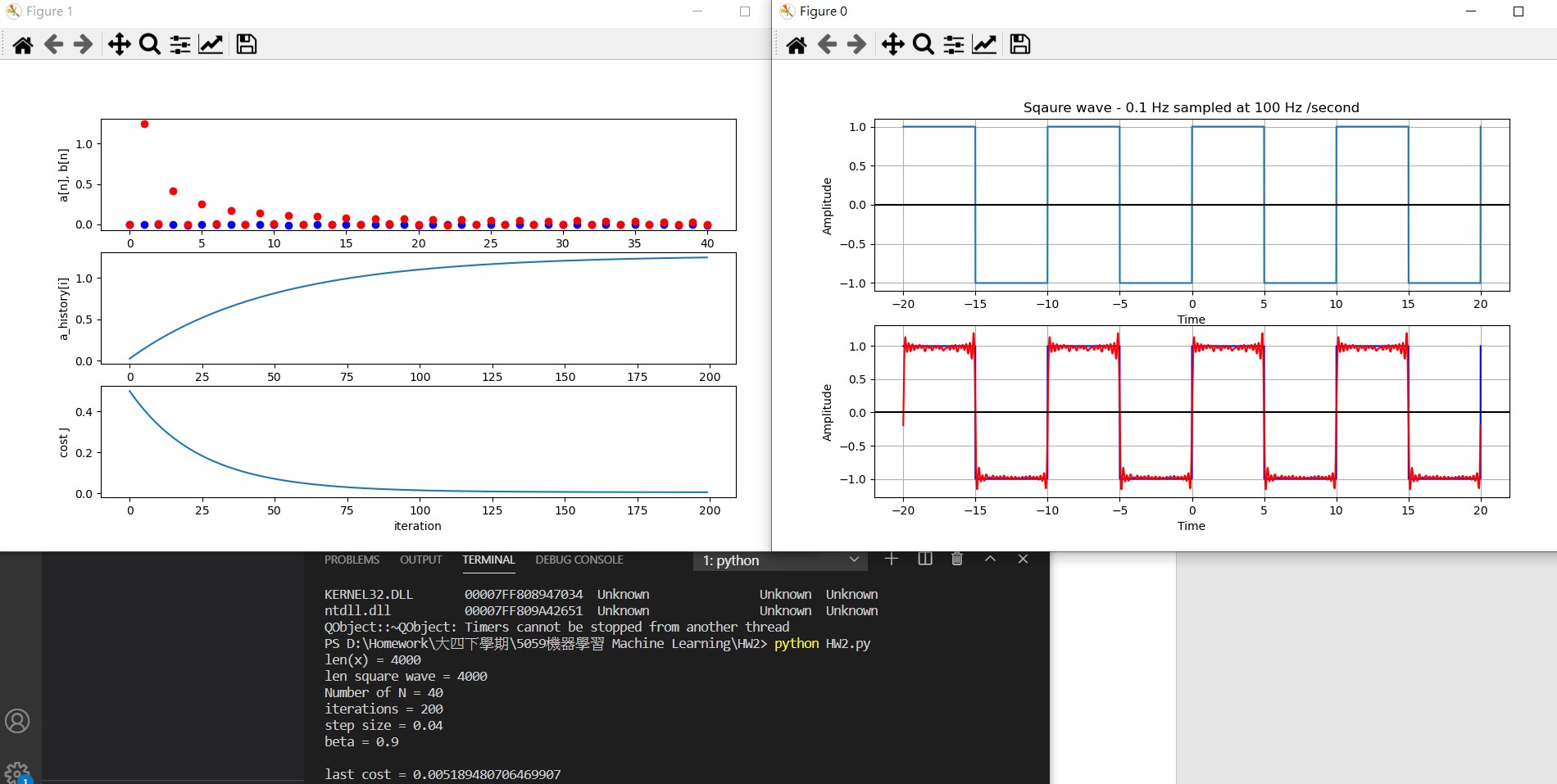
（2）iteration：1個iteration等於使用batch size個樣本訓練一次；

（3）epoch：1個epoch等於使用訓練集中的全部樣本訓練一次；

PART F

Consider the gradient descent method with momentum

Repeat part (b) and compare these two methods. Please give your observation:

 首先，由結果可以得知，兩種方法都可以得到與理論值接近的傅立葉係數。再執行兩次相同參數，其中一個使用一般的gradient descent，另一個使用momentum gradient descent。由下圖可以觀察到使用momentum可以非常快的得到收斂的結果，不過cost相對較大一些。

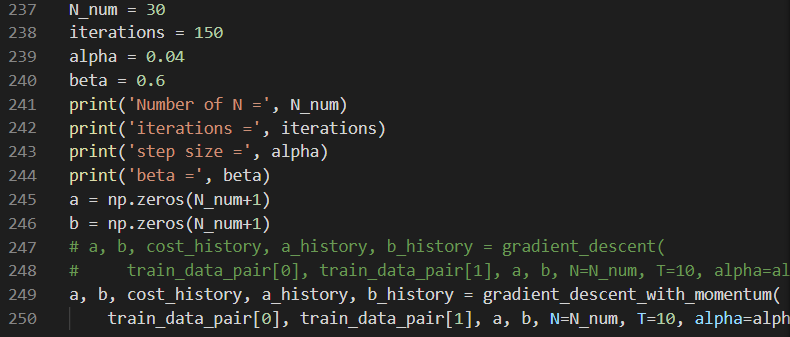
P.S. Momentum的參數beta由0~1，beta越大越容易脫離局部最小值，收斂到接近最小值很快，但收斂到最小值的速度會比較慢一些。當beta=0時，相當於一般的gradient descent。

PART G

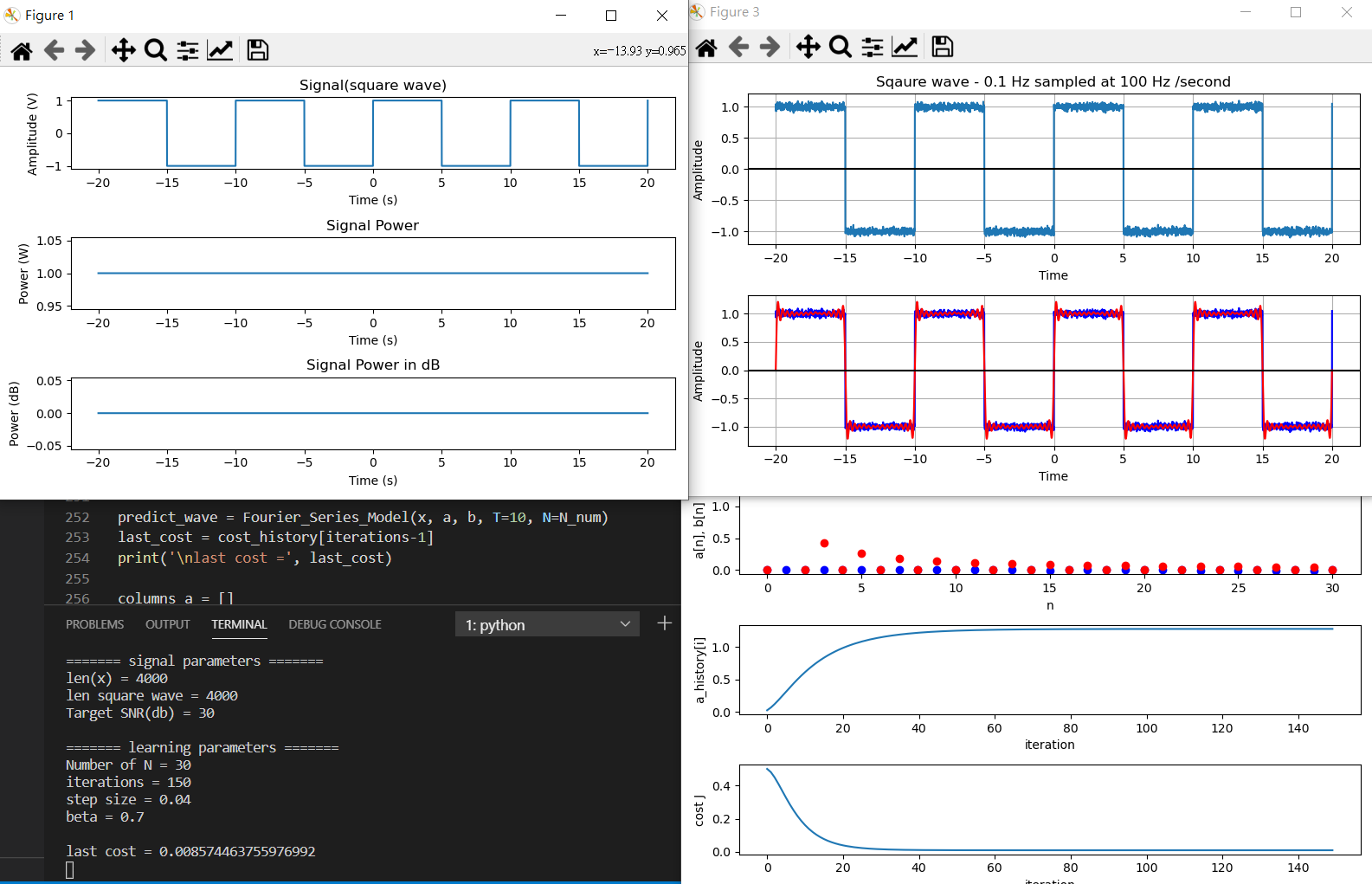
In actual system, the measure signals usually having sensor noise. Please add noise with different

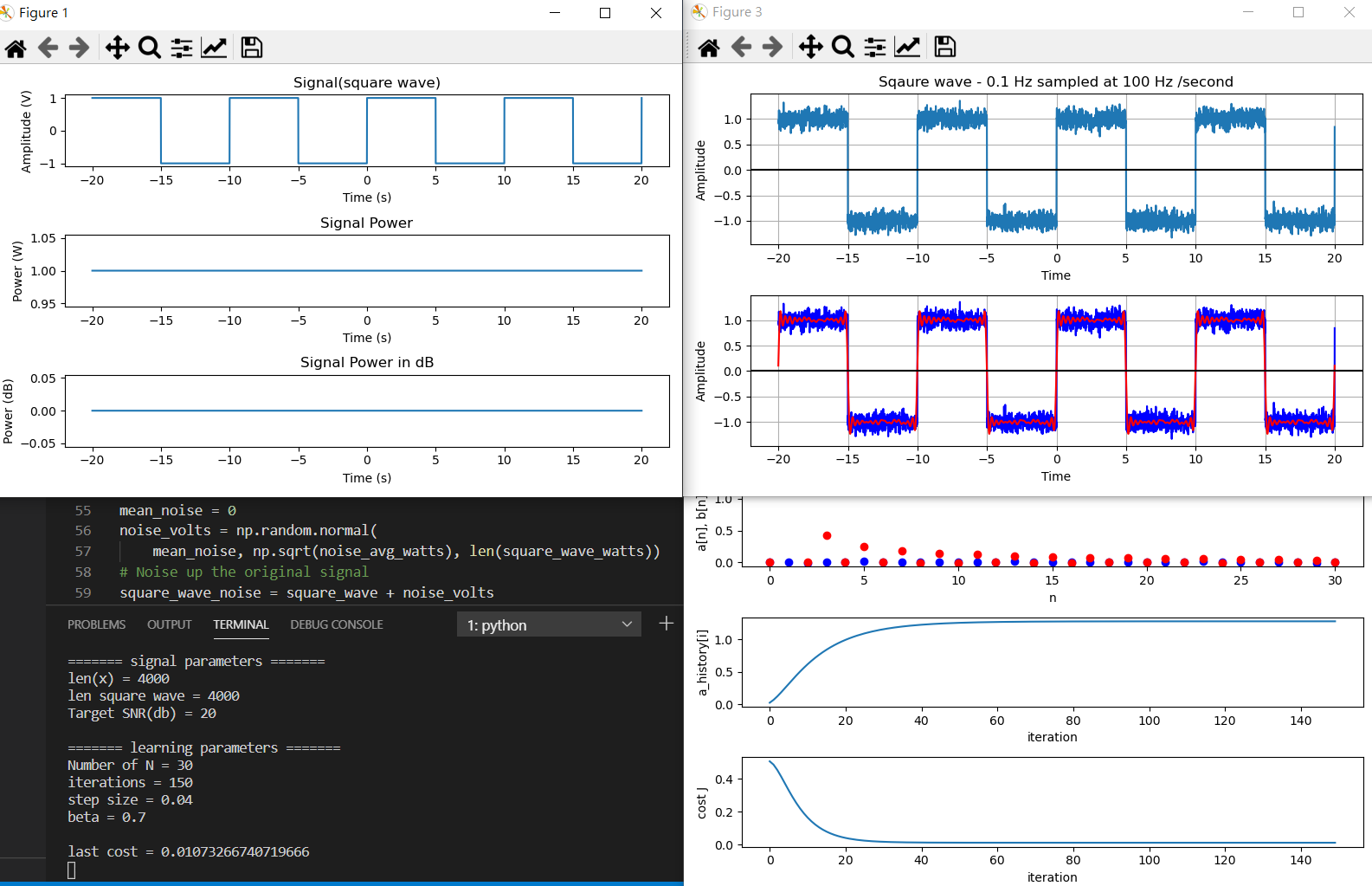
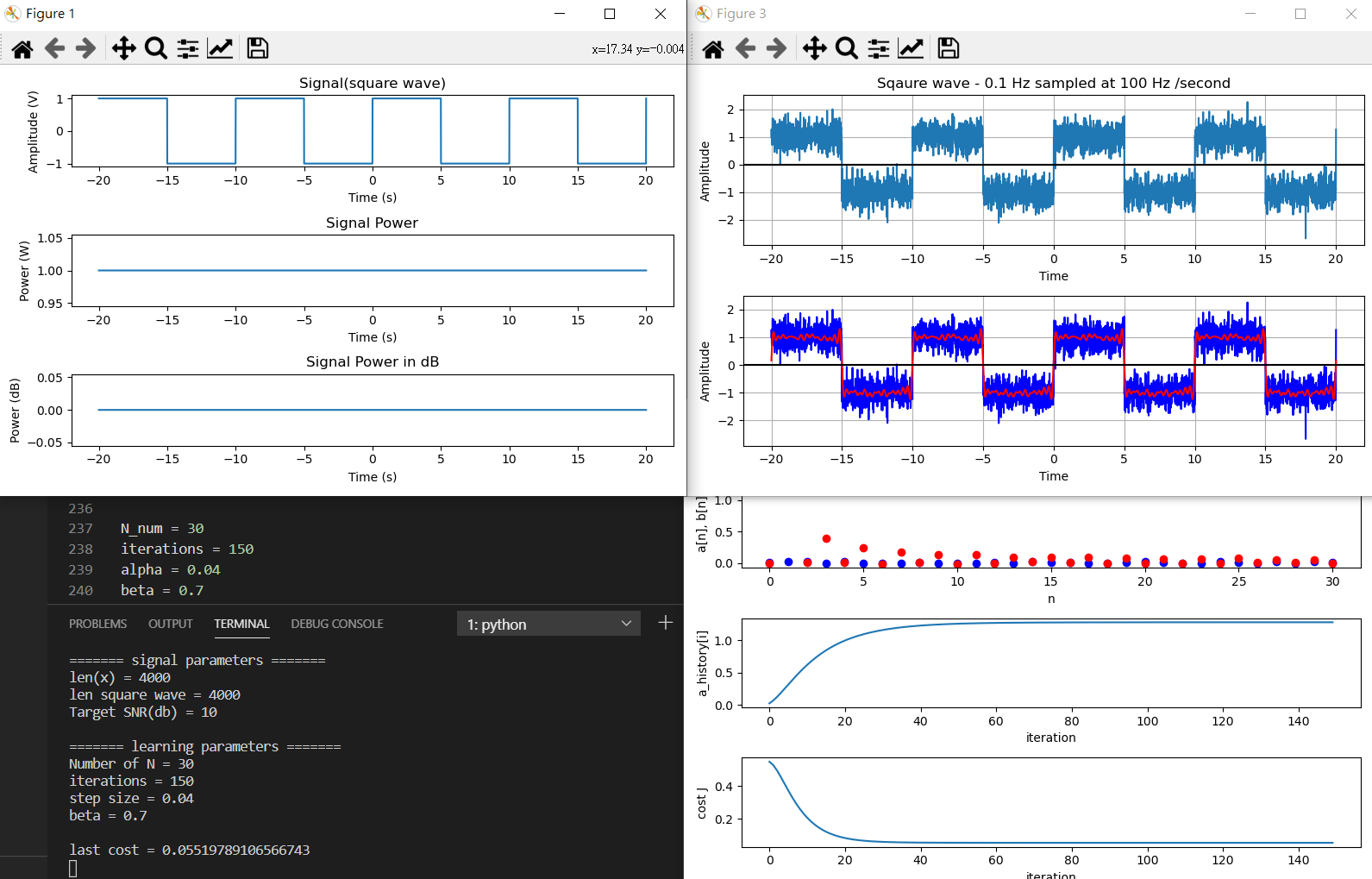
signal noise ratio (SNR), and repeat part (b) to obtain the analysis (SNR vs. MSE):

這邊我使用Momentum gradient descent來得到傅立葉係數的值，並且去近似包含雜訊的方波訊號。調整不同的SNR(db)來觀察MSE的變化。可以發現傅立葉係數仍然能夠收斂，不過MSE會比沒有雜訊來得大，雜訊越大，則MSE也會越大。



上圖為改變SNR(db)時的固定learning parameters

SNR(db)使用30,20,10來測試



SNR(db) = 30 last cost = 0.008574463755976992

SNR(db) = 20 last cost = 0.01073266740719666

SNR(db) = 10 last cost = 0.057933787953448265

P.S. 因為使用的是全批次學習(平均所有error)，所以模擬出來的方波不會被雜訊(雜訊震幅平均為0)帶偏，不會過學習。

PART H

As above, it can be viewed as *f*(*x*) is approximated by linear combination of sinusoidal basis. Would we use polynomial basis to treat it? Why?

可以使用多項函數去模擬任何函數，因為弦波函數的線性組合可以寫成多項式函數，也就是泰勒級數。如下圖。

