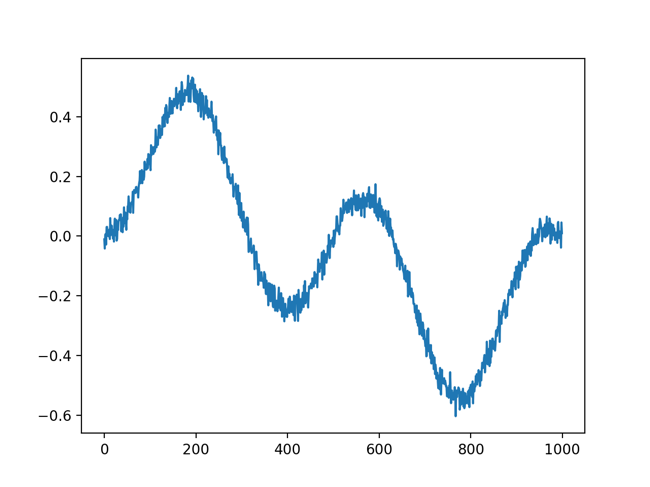
李少琪 人工智慧學程 0860908

Optimization Theory and Application HW3

題目：xcor = xˆ + v, 從xcor找出xˆ。

1. 讀出xcor

xcor：



xcor = np.load('denoise.npy')

x\_axis = np.linspace(0, len(xcor)-1, len(xcor))

plt.plot(x\_axis, xcor)

plt.show()

* 設定Ｄ

D = np.zeros((999, 1000))

*for* i in range(D.shape[0]):

D[i][i] = -1

D[i][i+1] = 1

I = np.identity((1000))

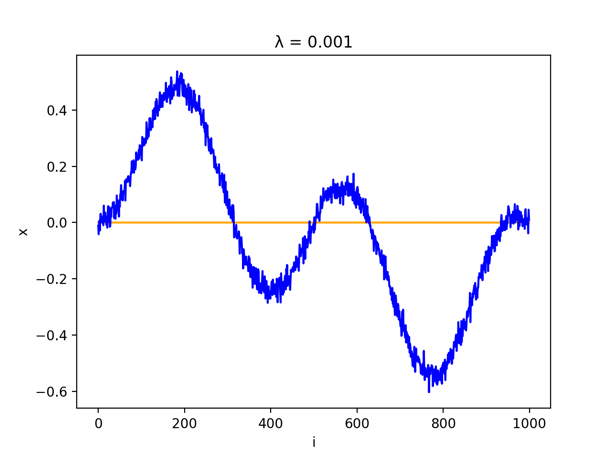
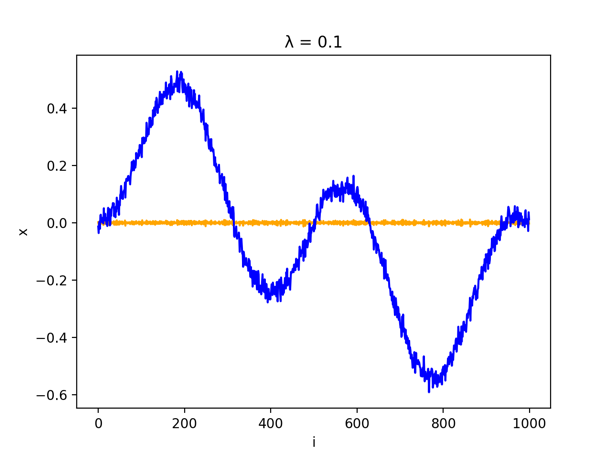
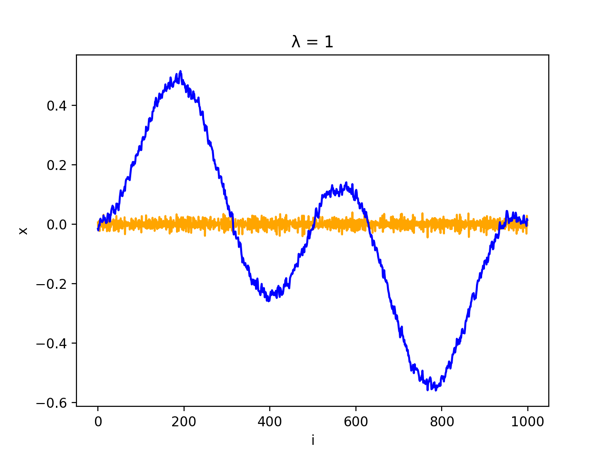
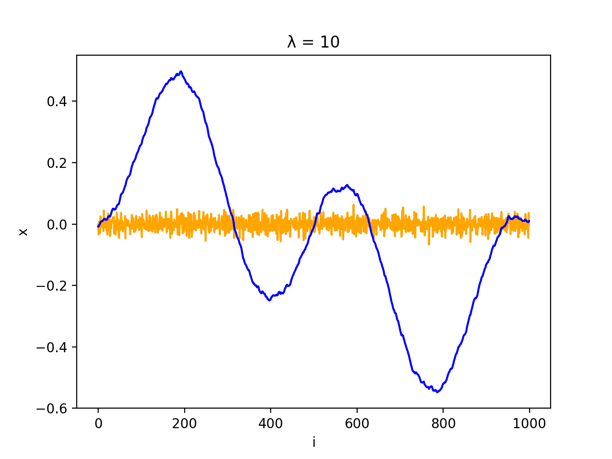
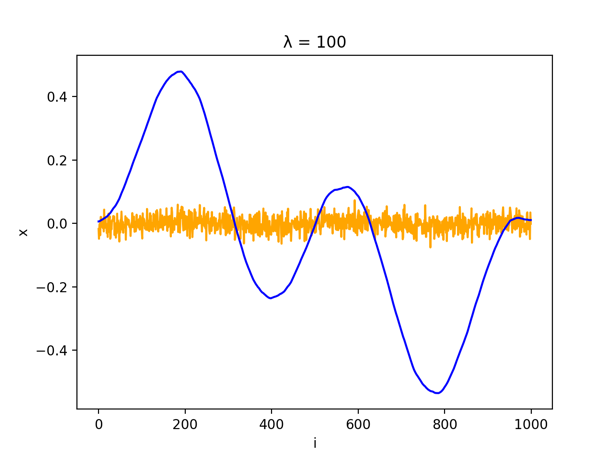
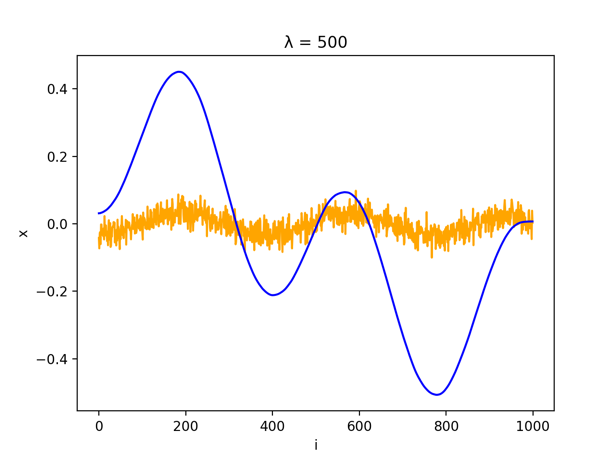
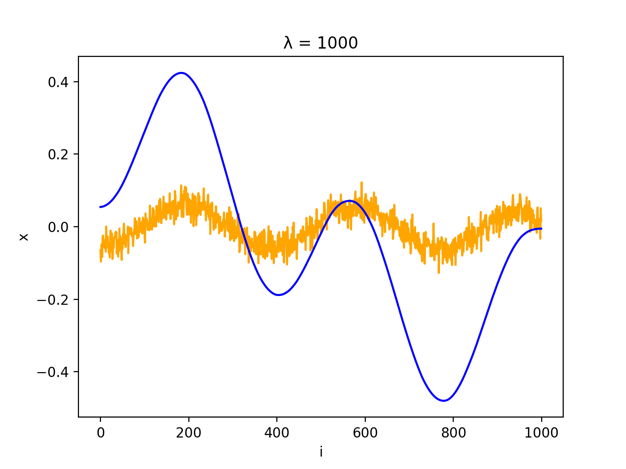
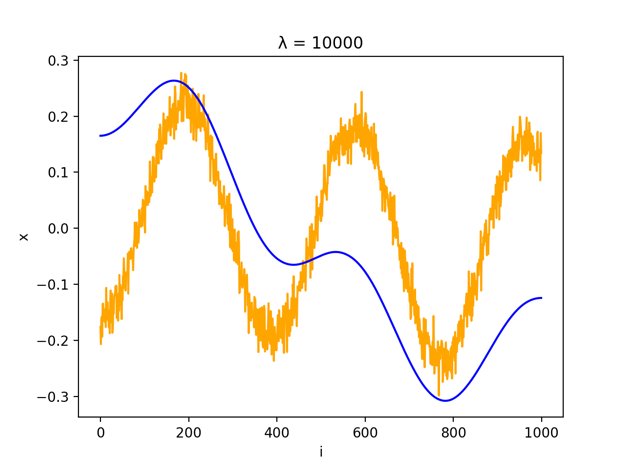
lamda = 100

X = np.linalg.inv(I + lamda \* D.T.dot(D)).dot(xcor)

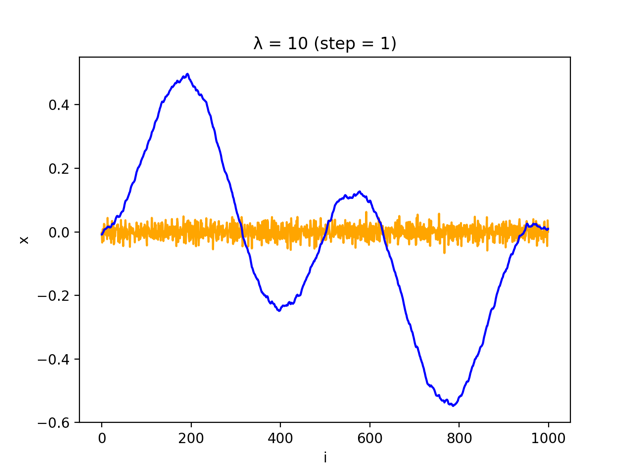
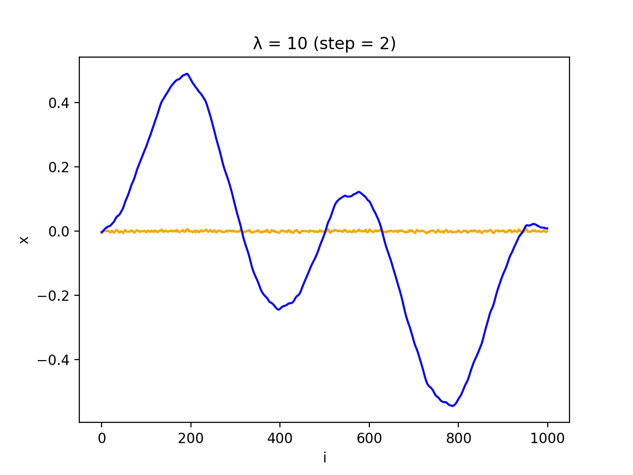
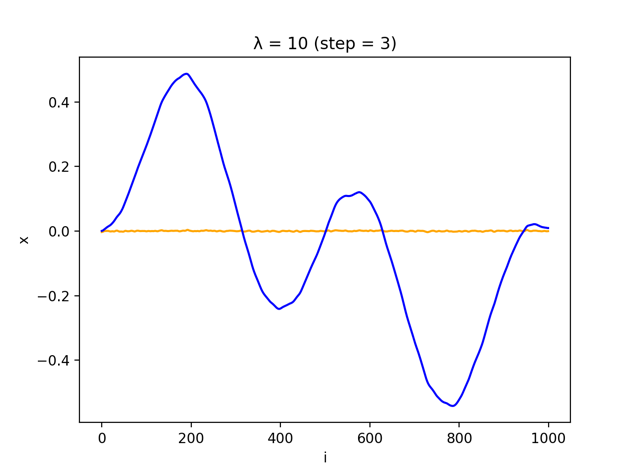
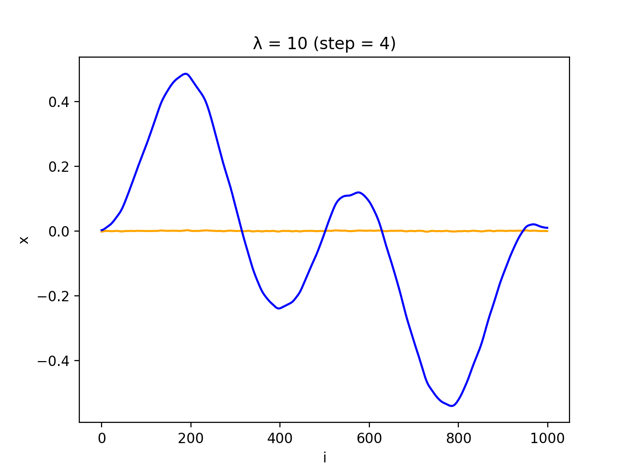
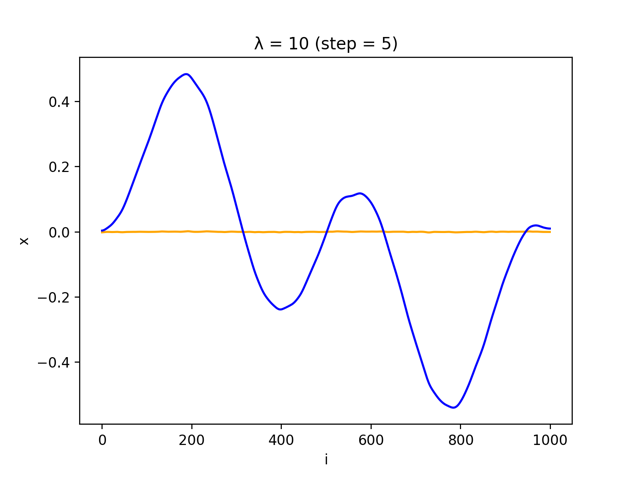
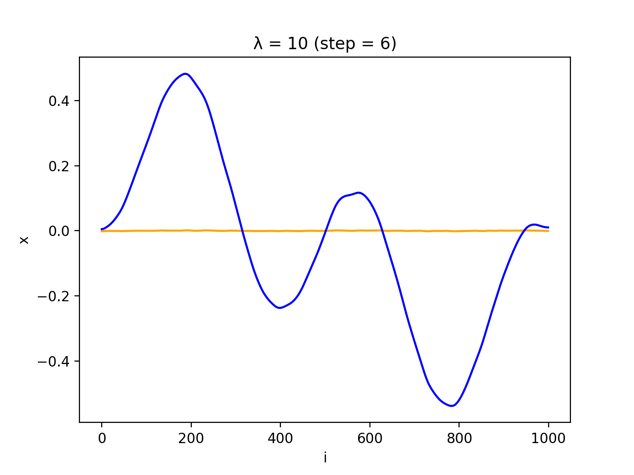
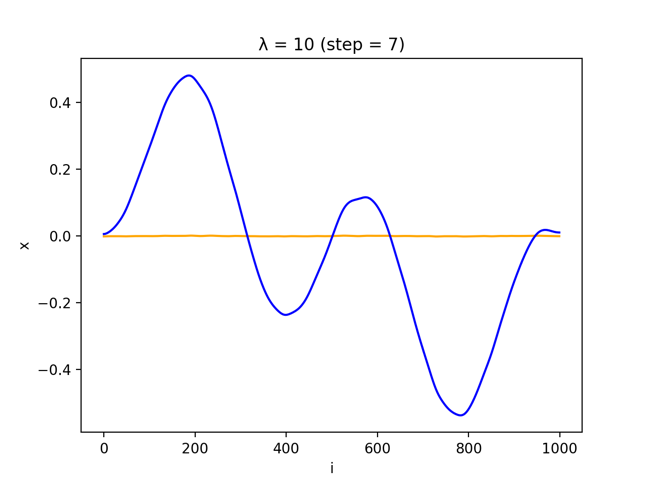
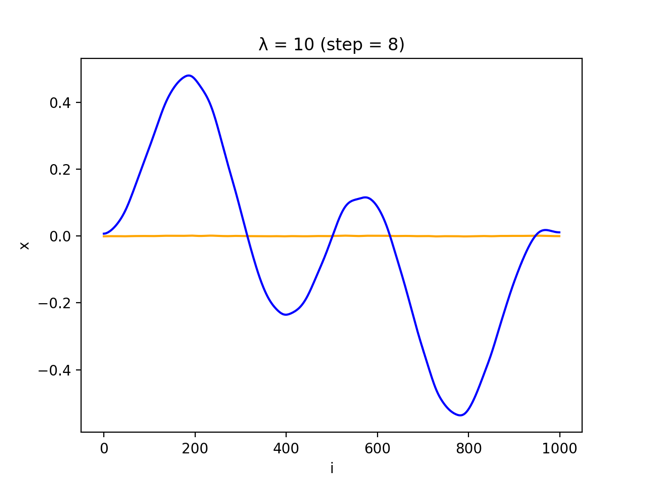
plt.plot(x\_axis, X)

plt.show()

* v = xcov – x^

1. 結果
2. 分析

由上述結果可以發現，當λ越小時，可以找出的noise越小，但當λ過大時，可能會連原本的訊號一起被濾掉。若想避免λ設定的問題，我嘗試了將λ固定在較小的值，將還原出來的訊號重複做least squares，但也要避免訓練過多次，結果如下。



*import* numpy *as* np

*from* matplotlib *import* pyplot *as* plt

*# read the data*

xcor = np.load('denoise.npy')

*# n = np.random.rand(1000)/-100+np.random.rand(1000)/-10+np.random.rand(1000)/10*

x\_axis = np.linspace(0, len(xcor)-1, len(xcor))

plt.plot(x\_axis, xcor)

plt.show()

*# set matrix D*

D = np.zeros((999, 1000))

*for* i in range(D.shape[0]):

D[i][i] = -1

D[i][i+1] = 1

*# calculate X*

I = np.identity((1000))

lamda = [0.001, 0.1, 1, 10, 100, 500, 1000, 10000]

*for* l in lamda:

X = np.linalg.inv(I + l \* D.T.dot(D)).dot(xcor)

noise = xcor - X

plt.plot(x\_axis, noise, *c*='orange')

plt.plot(x\_axis, X, *c*='blue')

plt.title("λ = {}".format(l))

plt.xlabel('i')

plt.ylabel('x')

plt.show()

*# sequential learning*

X = xcor

*for* step in range(0,8):

XX = np.linalg.inv(I + 10 \* D.T.dot(D)).dot(X)

noise = X - XX

plt.plot(x\_axis, noise, *c*='orange')

plt.plot(x\_axis, XX, *c*='blue')

plt.title("λ = 10 (step = {})".format(step + 1))

plt.xlabel('i')

plt.ylabel('x')

plt.show()

X = XX