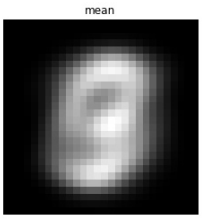
dsp2021_phw1

R10922129 劉旭庭

Q1:

把 70000 張 image 累加在一起然後除以 70000



Q2:

使用到手刻 cpca(centered pca) 函式，傳入要作 pca 的資料 x，會先算一次 mean，把 x 的每一筆資料都減掉 mean，然後找出 eigenvalue, eigenvector，隨後把 eigenvalue 轉正(對應的 eigenvector 也會隨之作 sign 的處理)

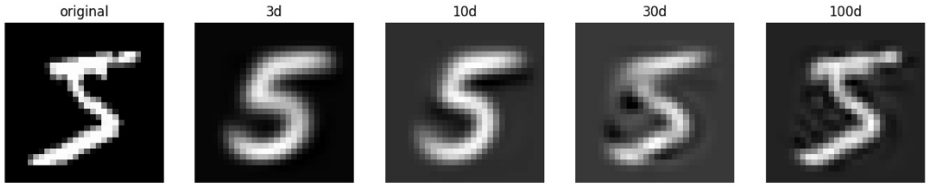
另外會對 eigenvalue 作 argsort 輸出成 vector m，越大的 eigenvalue index 會排越前面。

而 x 在處理完後會變成減去 mean 的形式。

以下是找出來 eigenvalue 最大的三個 eigenvector

Q3:

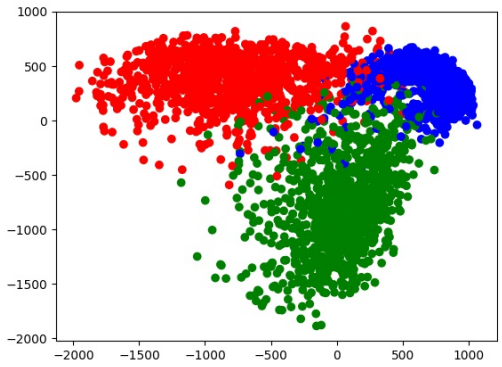
類似 Q2，在重建時會利用 vector m，找出前 N 大的 eigenvector，並且把要重建的 target image 投影到 eigenvector space 上，最後再把 mean 加回去。



Q4:

把 1,3,6 的 data 搜集起來後，作 cpca，找出前 2 大的 eigenvalue 對應的 eigenvector，分別計算每個資料點對應到的係數，隨後畫成圖。

1:blue, 3:green, 6:red



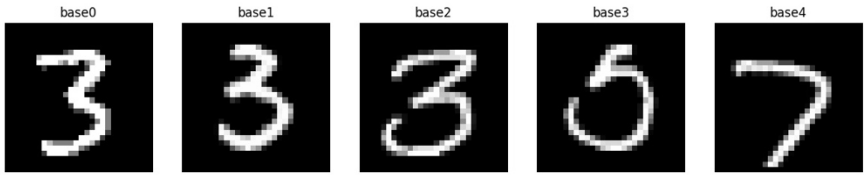
Q5:

使用手刻 omp 函式，會先對 training data 作歸一化，建立 use vector，表示 data 是否已經被選作 basis，已經被選到的 use 會是 False

依照內積絕對值的大小找出 k 個 basis 會依序 append 到 b，用來算每一輪的 r

r=x-(projection x on b)。

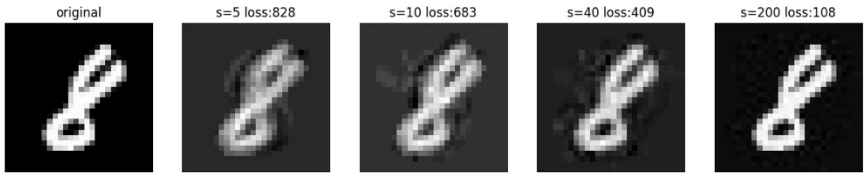
最後函式只會 return basis 的 index。 在函式外透過 index 找 basis



Q6:

原理同 Q5，只是重建時將 target image 投影到 basis 的 space 上

s 為 sparsity 縮寫，loss 為 L-2 norm



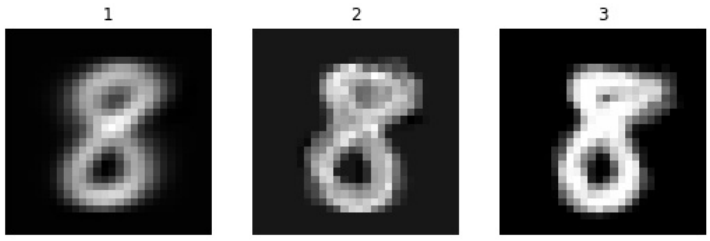
Q7:

1. 作法如 Q3
2. 做法如 Q6
3. Call sklearn 的 inear_model.Lasso
   參數: alpha=2.3, normalize=True, copy_X=True

   fit 完之後，對 coef_作 argsort，依據係數大到小對 index 作排列，利用排列後的 index 重建 target image 進行投影。
4. 對於參數的實驗，隨著 alpha 值越大，regularization 的 penalty 越大，非零細數也越來越少

   Alpha :1, 2, 2.3
   非零係數:18, 8, 5

   下圖分別對應 Q7-1,2,3 的重建結果



Bonus:

code fragment:

```
def soft(a,c,l):
    if c < -l:
        return (c+l)/a
    elif c>l:
        return (c-l)/a
    else:
        return 0
def lasso(orix,oriy,l):
    x=orix*1
    y=oriy*1
    ii,jj=x.shape
    for j in range(jj):
        x[:, j] /= (np.inner(x[:, j], x[:, j])**0.5)
    y/=(np.inner(y,y)**0.5)
    tempw=np.zeros(jj)
    a = np.zeros(jj)
    c = np.zeros(jj)
    s=x.T@x
    invs=np.linalg.pinv(s)
    w=invs@x.T@y
    while np.all(tempw!=w):
        tempw=w*1
        for j in range(jj):
            a[j] = 2*np.inner(x[:,j], x[:,j])
            c[j] = (2)*np.sum(x[:, j]@(y-(x@w)+(x[:, j]*w[j])))
            w[j]=soft(a[j],c[j],l)
    return w

z = lasso(vec8.T, last8, 1.5)
idx = np.argsort(-abs(z))
space = []
for i in range(5):
    space.append(vec8[idx[i]])
space = np.array(space)
draw(project(last8, space.T),'handcraft')
```

Implementation explaining:
soft 函式講義有提到，單純實作出來而已

lasso function 裡面會先對資料以及 target image 作歸一化，並且對係數 vector w 初始，這邊初始為 target image 投影到 training data 的係數，初始化其實蠻影響結果的。

隨後依照用講義補充的內容去更新，在 w 沒有更動後認定為收斂，return w

重建的部分也是對 w 作 argsort 找到係數前五大的 index，然後將 target image 投影上去做重建。