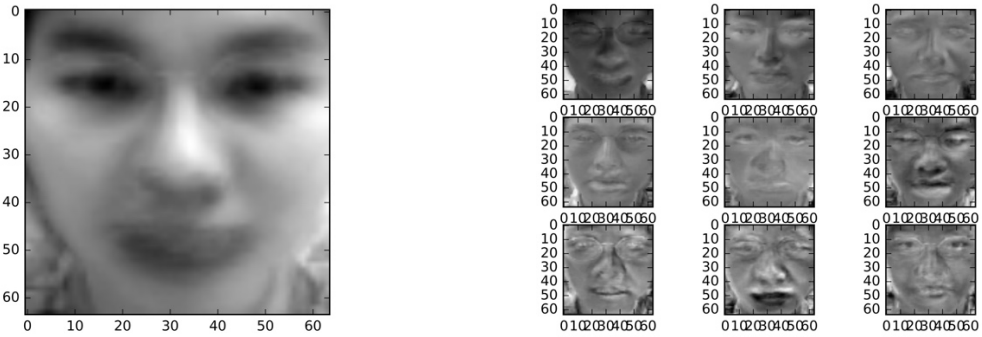
1.1. Dataset 中前 10 個人的前 10 張照片的平均臉和 PCA 得到的前 9 個 eigenfaces:
答： (左圖平均臉，右圖為 3x3 格狀 eigenfaces, 順序為 左到右再上到下)



1.2. Dataset 中前 10 個人的前 10 張照片的原始圖片和 reconstruct 圖 (用前 5 個 eigenfaces):
答：(左右各為 10x10 格狀的圖, 順序一樣是左到右再上到下)

1.3. Dataset 中前 10 個人的前 10 張照片投影到 top k eigenfaces 時就可以達到 < 1% 的 reconstruction error.
答： (回答 k 是多少)

2.1. 使用 word2vec toolkit 的各個參數的值與其意義:
答：

word2vec.word2phrase('./Book5TheOrderOfThePhoenix/all.txt',
'./Book5TheOrderOfThePhoenix/all-phrases', verbose=True)

word2phrase: 將 text file 轉成詞彙 (phrase) 檔，形成更好的 input 給 word2vec
第一個參數（train）：要被訓練的 text file 位址
第二個參數 （output）：要儲存的詞彙 (phrase) 檔位址
第三個參數 （verbose）: 程式執行時是否顯示資訊

word2vec.word2vec('./Book5TheOrderOfThePhoenix/all-phrases',
'./Book5TheOrderOfThePhoenix/all.bin', size=w2v_size, verbose=True)

word2vec: 訓練 text file / 詞彙 (phrase) 檔，得到 model
第一個參數（train）：要被訓練的 text file / 詞彙 (phrase) 位址
第二個參數 （output）：要儲存的 word vectors binary 檔位址
第三個參數 （size）: 要形成的 word vector 的長度
第四個參數 （verbose）: 程式執行時是否顯示資訊

w2v_model = word2vec.load('./Book5TheOrderOfThePhoenix/all.bin')

word2vec.load: 載入 word to vectors 的 model
第一個參數：word2vec model binary 檔位置

w2v_model.vocab

vocab: array，儲存各個單詞彙，對應到 w2v_model.vectors

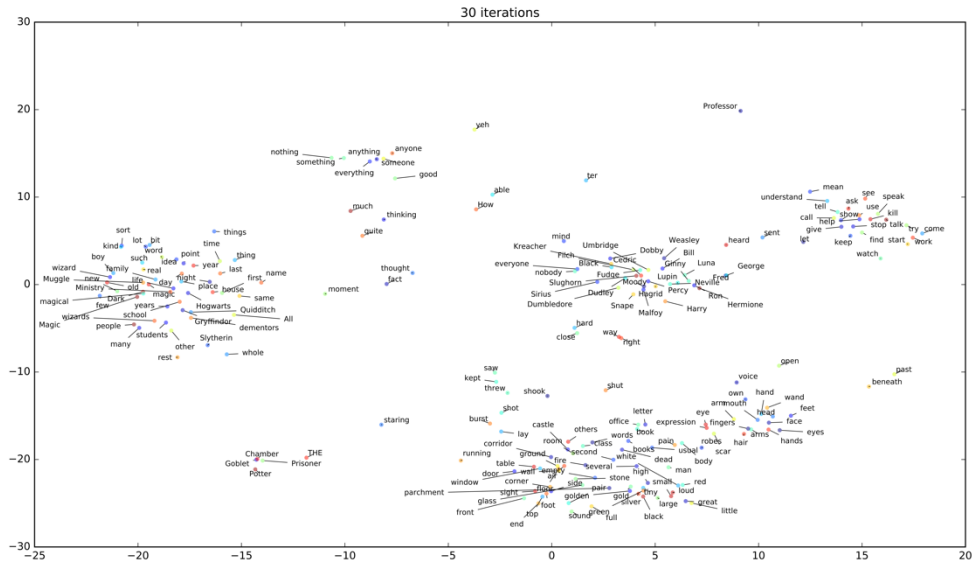
```
array(['</s>', 'the', 'and', ..., 'thing:', 'neared', 'appear,'],
      dtype='<U78')
```

w2v_model.vectors

vector: array，儲存各個單詞彙的 word vectors，對應到 w2v_model.vocab

```
array([[ 0.04332093,  0.04783104, -0.04145478, ...,  0.0383517 ,
         0.00421121,  0.02781213],
       ...,
       [ 0.00882514, -0.04131756,  0.00608916, ...,  0.04523514,
        -0.02729767, -0.03263131]])
```

2.2. 將 word2vec 的結果投影到 2 維的圖:
答： (圖)



2.3. 從上題視覺化的圖中觀察到了什麼？
答：
可以看到有很明顯的分群。

最右邊那一群來看，可以看出是動詞類的被分再一起了。

中間稍微偏右的那一群則是由人名、主詞所構成的一群。

最下方那一群可以分三小群來看。
以 x 軸來分，
[-5,0]: 房間、屋子、構造物為主的一群
[0,10]: 形容詞
[10,15]: 人體部位和特徵

最左邊那一群稍微雜亂，但還是看得出一些端倪，像是 day 和 night 靠很近，new 和 old 也靠很近。

3.1. 請詳加解釋你估計原始維度的原理、合理性，這方法的通用性如何？
答：

knbrs = NearestNeighbors(n_neighbors=200, algorithm='ball_tree').fit(data)

運用了 NearestNeighbors 來做，將 n_nighbor 設 200，algorithm 使用 ball_tree，將原本的資料餵進去。

因為原本的維度是 100 ，比較大而不適合用 kd_tree，所以用 ball_tree

之後從中取 3 點，用 svd 算出他們的 singular eigenvalues，再將其作為 input data 餵給 SVR。

svr = SVR(C=1)
svr.fit(X, y)

其中 y 是取 log(dim)，這樣直接 train 出來會有比較好的結果。

3.2. 將你的方法做在 hand rotation sequence datatset 上得到什麼結果？合理嗎？請討論之。
答：