Robert Heeter
ELEC 378
02/3/2023

Problem Set #3

1. a. See code below. It appears that the intercepted transmission is likely binary communication as when the data is projected on to the first two principal components, two distinct clusters centered on two different values on PC1 are obvious (see scatterplot).

b. See code below and colored scatterplot - the clusters are now clearly colored.

c. See decoded message and code below.

2. See code below and digit outputs. A noise level of 3 was used with the first 5 eigenvectors used for de-noising.

3. Next page.

3. a. See code below. The plot shows the cancer patients plotted on a graph with principal component 1 and 2 as the axes. From the plot, it is clear that principal component 2 ("PC2") is most informative in determining a patient's melanoma diagnosis. When the patient data is projected on to PC1, the melanoma patients mix with the other non-melanoma patients (i.e. they are all clustered together). However, when the patient data is projected on to PC2, two clusters are formed with one being primarily non-melanoma patients and the other being melanoma patients.

b. Coordinate of i^{th} data point projected on to j^{th} principal direction:

$$\langle x_i, v_j \rangle = \sum_{k=1}^{p} x_i[k] \, v_j[k]$$

k^{th} gene of x_i ; v_j is j^{th} eigenvector of $X^T X$

If v_j is the most informative principal direction for some diagnosis, then the genes $x_i[k]$ for which $v_j[k]$ is large must be the most important genes in causing or contributing to that diagnosis.

For example, for melanoma, the genes $x_i[k]$ for which $v_2[k]$ is large (second principal component is most informative principal direction for diagnosis of melanoma, determined in part A) are the most important genes in causing or contributing to melanoma.

c. Given that large $v_j[k]$ entries correspond to genes $x_i[k]$ that are responsible for a given diagnosis, given v_j is the most informative principal component for diagnosis), sorting the genes $x_i[k]$ by their corresponding term/element in $v_j[k]$ in magnitude (largest to smallest), gives a sorted list of genes that are most to least impactful for the diagnosis.

Looking at the heat map produced (see code below), the patients with melanoma and their corresponding melanoma genes can be seen as a yellow-green band at the bottom of the heatmap. The most yellow genes are most important for diagnosis (large $v_2[k]$)

4. a. $-x_1 + 2x_2 + 3x_3 = 16 \rightarrow -4x_1 + 8x_2 + 12x_3 = 64$

$4x_1 - 2x_2 - x_3 = -2 \rightarrow 4x_1 - 2x_2 - x_3 = -2$

$x_2 + x_3 = 7$ $\qquad 6x_2 + 11x_3 = 62$

$x_2 = 7 - x_3$

$4(7 - x_3) + 11x_3 = 62 \rightarrow 5x_3 = 20$

$x_3 = 4$

$\therefore x_2 = 7 - x_3 = 7 - 4 = 3$

$\therefore x_1 = 2x_2 + 3x_3 - 16$

$= 2(3) + 3(4) - 16$

$x_1 = 2$

$x_1 = 2$
$x_2 = 3$
$x_3 = 4$

↳ critical case b/w over and underdetermined where # of equations equals # of free variables

b. $x_1 - x_2 - 2x_3 = -2$
$2x_1 + 2x_2 + x_3 = 5$
$3x_1 - 3x_2 - 6x_3 = -6$

$$= \left[\begin{array}{ccc|c} 1 & -1 & -2 & -2 \\ 2 & 2 & 1 & 5 \\ 3 & -3 & -6 & -6 \end{array}\right]$$

Add $-2 \times R_1$ to R_2

Add $-3 \times R_1$ to R_3

$$= \left[\begin{array}{ccc|c} 1 & -1 & -2 & -2 \\ 0 & 4 & 5 & 9 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Add $\frac{1}{4} R_2$ to R_1

Multiply R_2 by $\frac{1}{4}$

$$= \left[\begin{array}{ccc|c} 1 & 0 & -\frac{3}{4} & \frac{1}{4} \\ 0 & 1 & \frac{5}{4} & \frac{9}{4} \\ 0 & 0 & 0 & 0 \end{array}\right]$$

$x_1 \quad x_2 \quad x_3 \quad b$

Underdetermined system:

$x_1 = \frac{3}{4} x_3 + \frac{1}{4}$

$x_2 = -\frac{5}{4} x_3 + \frac{9}{4}$

x_3 = free variable (can be any arbitrary value)

c. Next page

(4). c. $4x_1 + 3x_2 + 2x_3 = 17$
$x_1 + 2x_2 + 3x_3 = 8$

$$= \left[\begin{array}{ccc|c} 4 & 3 & 2 & 17 \\ 1 & 2 & 3 & 8 \end{array}\right]$$ Add $-\frac{1}{4}R_1$ to R_2, Multiply R_1 by $\frac{1}{4}$

$$= \left[\begin{array}{ccc|c} 1 & \frac{3}{4} & \frac{1}{2} & \frac{17}{4} \\ 0 & \frac{5}{4} & \frac{5}{2} & \frac{15}{4} \end{array}\right]$$ Add $-\frac{3}{5}R_2$ to R_1, Multiply R_2 by $\frac{4}{5}$

$$= \left[\begin{array}{ccc|c} 1 & 0 & -1 & 2 \\ 0 & 1 & 2 & 3 \end{array}\right]$$
$x_1 \quad x_2 \quad x_3 \quad b$

Underdetermined system:
$x_1 = x_3 + 2$
$x_2 = -2x_3 + 3$
x_3 = free variable (can be any arbitrary value)

d. $2x_1 + x_2 = 4 \longrightarrow -2x_1 - x_2 = -4$
$x_1 + 3x_2 = 4 \longrightarrow 2x_1 + 6x_2 = 8$
$-x_1 + 2x_2 = 3$

$5x_2 = 4 \rightarrow x_2 = \frac{4}{5}$

$x_1 = 4 - 3(x_2) = 4 - 3\left(\frac{4}{5}\right)$

$x_1 = \frac{8}{5}$

plug into eqn. 3

$-\left(\frac{8}{5}\right) + 2\left(\frac{4}{5}\right) \stackrel{?}{=} 3$

$-\frac{8}{5} + \frac{8}{5} = 0 \neq 3$

$x_1 = \frac{8}{5}$, $x_2 = \frac{4}{5}$ } satisfies eqns #1 and #2 but not eqn #3

System of linear equations is inconsistent (no set of x_1, x_2, x_3 will satisfy all 3 equations)

ELEC378-HW3

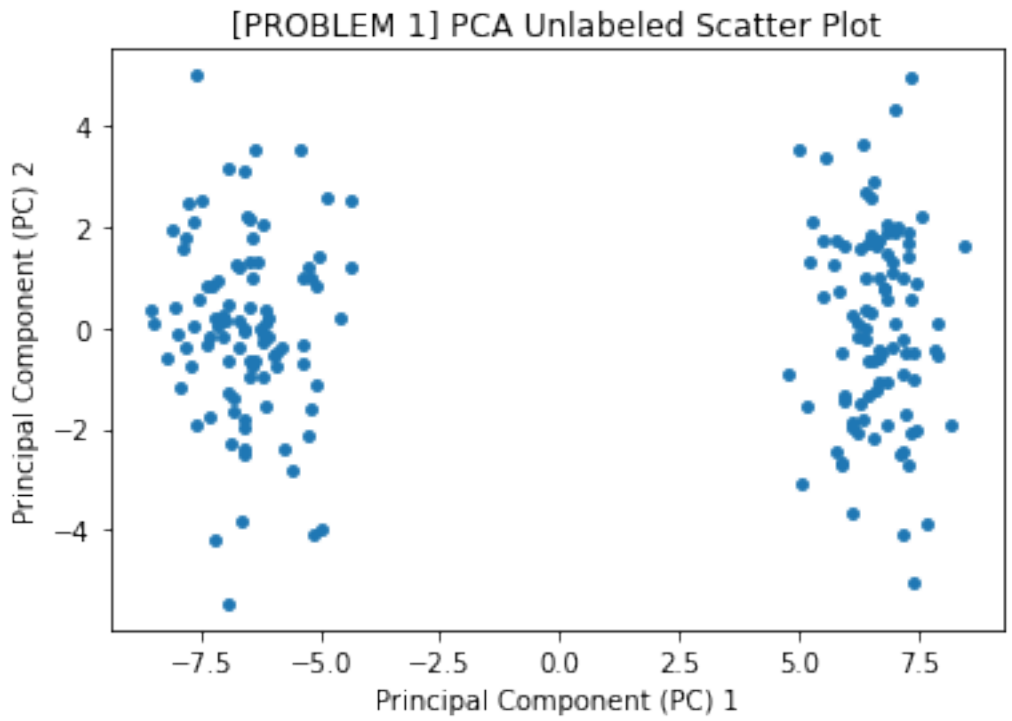
February 3, 2023

[1]:
```
# ROBERT HEETER
# ELEC 378 Machine Learning
# 3 February 2023

# PROBLEM SET 3
```

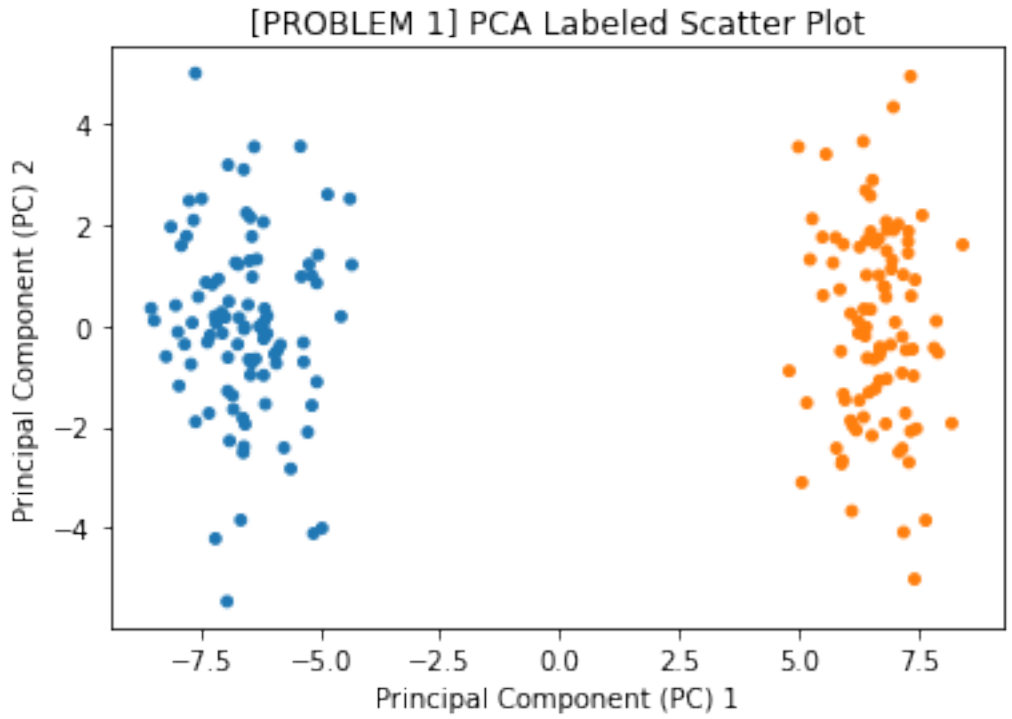
[2]:
```
import scipy.io as sc
import numpy as np
import matplotlib.pyplot as plt
```

[3]:
```
# PROBLEM 1

# import file with vectors on each row
eaves_file = sc.loadmat('eavesdropping.mat')
eaves = eaves_file["Y"]
```

[4]:
```
# PART A

# center the data by subtracting column (dimension) mean from each column
eaves_c = eaves-np.mean(eaves,axis=0)

# take SVD
(u,s,vh) = np.linalg.svd(eaves_c)

# pca transform matrix to reduce dimensionality to 2
PCA = np.dot(u,np.diag(s))[:,0:2]

# plot approximation
plt.scatter(PCA[:,0],PCA[:,1],s=15)
plt.title("[PROBLEM 1] PCA Unlabeled Scatter Plot")
plt.xlabel('Principal Component (PC) 1')
plt.ylabel('Principal Component (PC) 2')
plt.show()
```



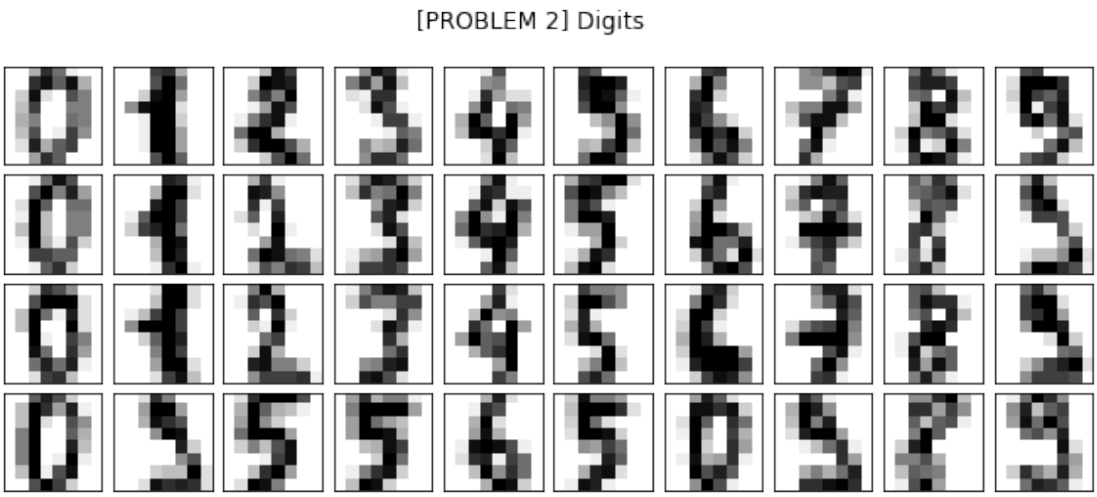
[5]:
```
# PART B

# create a vector of binary labels indicating which cluster each data point␣
 ↪lives in
assignment = np.where(PCA[:,0]>0,1,0)

# color and plot data
colors = np.choose(assignment,['#1f77b4','#ff7f0e'])
plt.scatter(PCA[:,0],PCA[:,1],s=15,c=colors)
plt.title("[PROBLEM 1] PCA Labeled Scatter Plot")
plt.xlabel('Principal Component (PC) 1')
plt.ylabel('Principal Component (PC) 2')
plt.show()
```



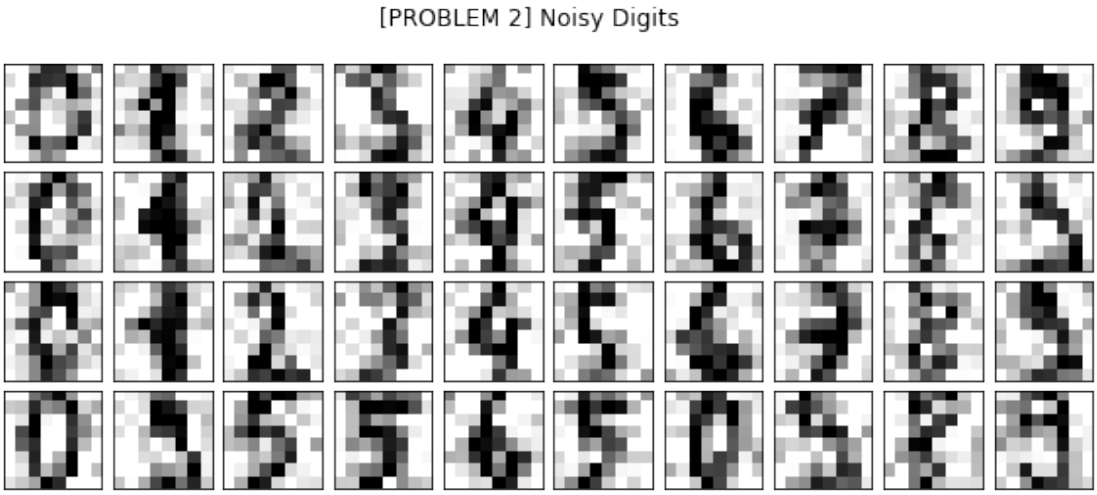
[6]:
```
# PART C

# choose a value for each bit
choices = np.choose(assignment,[0,1])

# reshape to 7-bit ASCII
reshaped = np.reshape(choices,(27,7))

# convert binary to ASCII to decode
message = ""
for i in reshaped:
    temp_string = ""
    for num in i:
        temp_string = temp_string + str(num)
    char = int(temp_string, 2)
    message = message + chr(char)

print('[PROBLEM 1] DECODED MESSAGE:')
print(message)
```

```
[PROBLEM 1] DECODED MESSAGE:
richb is love richb is life
```

[7]:
```
# PROBLEM 2

# load the data matrix containing as each row an unwrapped
# array representation of an image of a handwritten digit
digits = np.load('digits.npy')

# show digits
fig, axes = plt.subplots(4, 10, figsize=(10, 4),subplot_kw={'xticks':[],␣
↪'yticks':[]},gridspec_kw=dict(hspace=0.1, wspace=0.1))
fig.suptitle('[PROBLEM 2] Digits')
for i, ax in enumerate(axes.flat):
    ax.imshow(digits[i].reshape(8, 8),cmap='binary',␣
↪interpolation='nearest',clim=(0, 16))
```

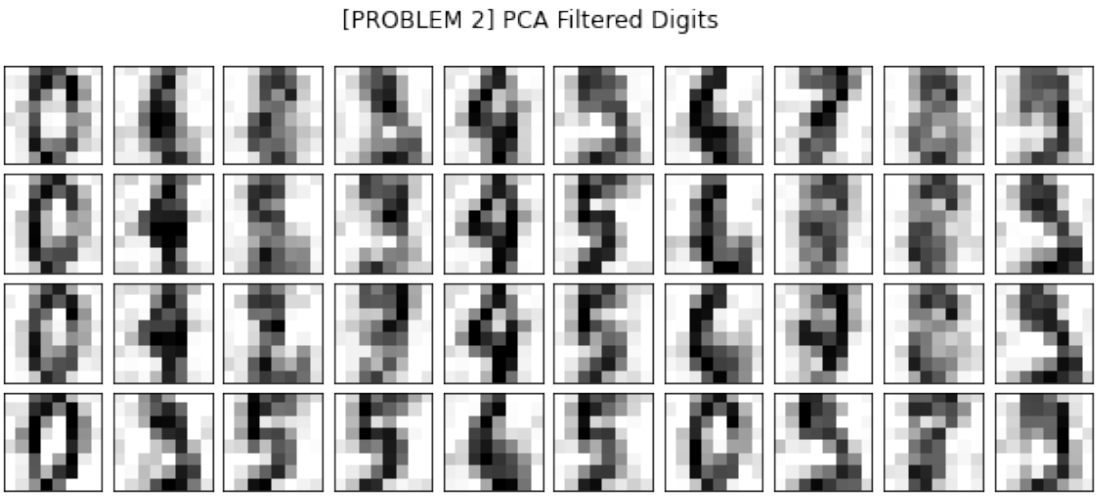


[8]:
```
# add noise and show noisy digits
noise_amount = 3
noisy_digits = np.random.normal(digits.data, noise_amount)

fig, axes = plt.subplots(4, 10, figsize=(10, 4),subplot_kw={'xticks':[],␣
↪'yticks':[]},gridspec_kw=dict(hspace=0.1, wspace=0.1))
fig.suptitle('[PROBLEM 2] Noisy Digits')
for i, ax in enumerate(axes.flat):
    ax.imshow(noisy_digits[i].reshape(8, 8),cmap='binary',␣
↪interpolation='nearest',clim=(0, 16))
```



[9]:
```
# center the data by subtracting column (dimension) mean from each column
noisy_digits_mean = np.mean(noisy_digits,axis=0)
noisy_digits_c = noisy_digits-noisy_digits_mean

# take SVD
[u,s,vh] = np.linalg.svd(noisy_digits_c)

# keep only the first q eigenvectors as V_q
# compute P_q * V_q^H = X_noisy * V_q * V_q^H
# view the results

# pca transform matrix
PCA = np.dot(u,np.diag(s))

# keep only the first q eigenvectors as v_q
q = 5
v_q_T = vh[0:q,:]

# compute noisy_digits_filtered = PCA*v_q^H = P_q*V_q^H
noisy_digits_filtered = np.matmul(PCA[:,0:q],v_q_T)

# add back column means to filtered digits
noisy_digits_filtered = noisy_digits_filtered + noisy_digits_mean

# show filtered digits
fig, axes = plt.subplots(4, 10, figsize=(10, 4),subplot_kw={'xticks':[],␣
↪'yticks':[]},gridspec_kw=dict(hspace=0.1, wspace=0.1))
fig.suptitle('[PROBLEM 2] PCA Filtered Digits')
for i, ax in enumerate(axes.flat):
```

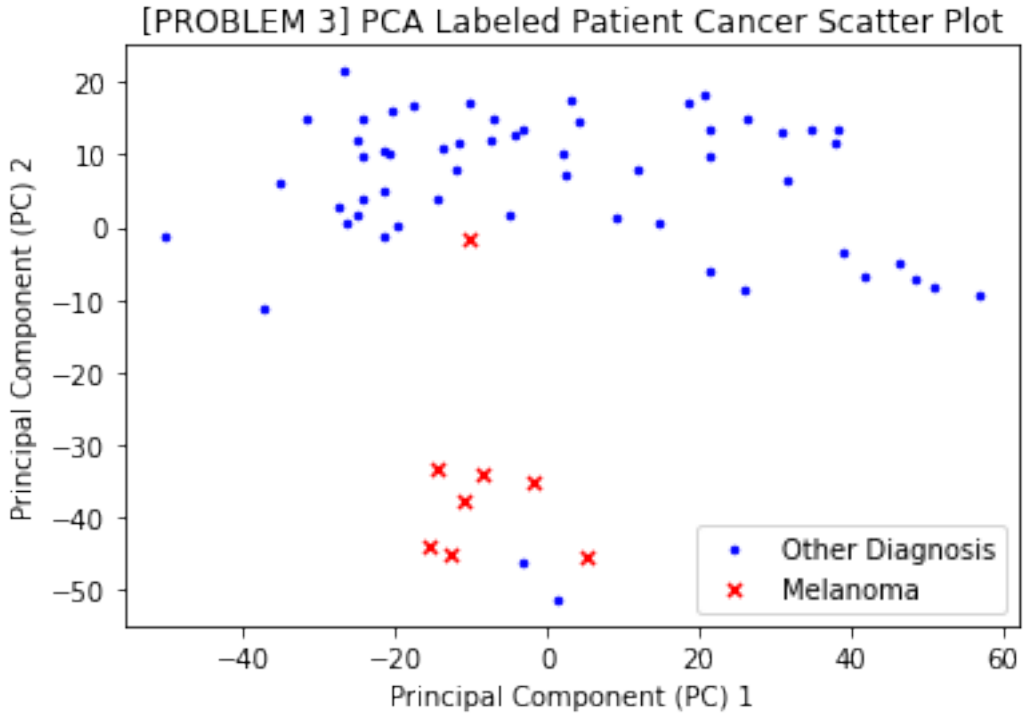
```
    ax.imshow(noisy_digits_filtered[i].reshape(8, 8),cmap='binary',␣
↪interpolation='nearest',clim=(0, 16))
```



[10]:
```
# PROBLEM 3

cancer = sc.loadmat('cancer.mat')
X = np.array(cancer['X'])
Y = ([y[0][:] for y in np.concatenate(cancer['Y'][:])])
```

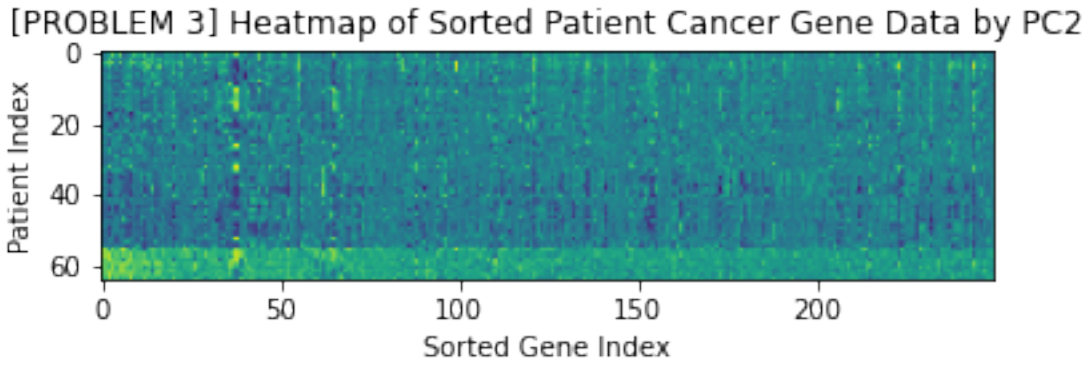
[11]:
```
# PART A

# use PCA to reduce the dimensionality of X to 2
# display a scatterplot of the 2D representation
# including the corresponding labels in Y

# indices of patients with 'MELANOMA'
I_mel = np.char.startswith(Y,'MELANOMA')

# Xmel_n = preprocessing.scale(Xmel,axis=0,with_mean=True)
# Xmel_n = preprocessing.normalize(Xmel_n,axis=0)

# center the data by subtracting column (dimension) mean from each column
X_c = X-np.mean(X,axis=0)

# take SVD
[X_u,X_s,X_vh] = np.linalg.svd(X_c)

# pca transform matrix to reduce dimensionality to 2
```

```
PCA = np.dot(X_u,np.diag(X_s))[:,0:2]

# color and plot data
plt.scatter(PCA[~I_mel,0],PCA[~I_mel,1],color='blue',s=25,marker='.
 ↪',label='Other Diagnosis')
plt.
 ↪scatter(PCA[I_mel,0],PCA[I_mel,1],color='red',s=25,marker='x',label='Melanoma')
plt.xlabel('Principal Component (PC) 1')
plt.ylabel('Principal Component (PC) 2')
plt.legend(loc="lower right")
plt.title('[PROBLEM 3] PCA Labeled Patient Cancer Scatter Plot')
plt.show()
```



[12]:
```
# PART C

# sort columns of X by v_j[k]
X_c_sorted = X_c[:,(X_vh[1,:]).argsort()]

# produce heatmap of sorted X
plt.imshow(X_c_sorted[:,0:250])
plt.xlabel('Sorted Gene Index')
plt.ylabel('Patient Index')
plt.title('[PROBLEM 3] Heatmap of Sorted Patient Cancer Gene Data by PC2')
```

```
plt.show()
```


[PROBLEM 3] Heatmap of Sorted Patient Cancer Gene Data by PC2
Patient Index
0
20
40
60
0
50
100
150
200
Sorted Gene Index

```
[ ]:
```