Homework 10 Solutions:: MATH 504

Your homework submission must be a single pdf called "LASTNAME-hw10.pdf" with your solutions to all theory problem to receive full credit. All answers must be typed in Latex.

1. Consider "Rosenbrock" function

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

With a starting point $[0, 0]^\top$, apply two iterations of Newton's method to minimize Rosenbrock function. Hint:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Solution.

Newton's method iterates $x^{(k+1)} = x^{(k)} - F(x^{(k)})^{-1} g^{(k)}$ where $F(x^{(k)} = \nabla^2 f(x^{(k)})$ and $g^{(k)} = \nabla f(x^{(k)})$.

Here, $\nabla f(x_1, x_2) = \begin{bmatrix} -400x_1(x_2 - x_1) - 2(1 - x_1) \\ 200(x_2 - x_1) \end{bmatrix}$, and

$\nabla^2 f(x_1, x_2) = \begin{bmatrix} 1200x_1^2 - 400x_2 + 2 & -400x_1 \\ -400x_1 & 200 \end{bmatrix}$

Iteration 1:

$x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $F(x^{(0)})^{-1} = \begin{bmatrix} 2 & 0 \\ 0 & 200 \end{bmatrix}^{-1} = \frac{1}{400} \begin{bmatrix} 200 & 0 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{200} \end{bmatrix}$, $g^{(0)} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$.

Therefore, $x^{(1)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{200} \end{bmatrix} \begin{bmatrix} -2 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Iteration 2:

$x^{(2)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 1202 & -400 \\ -400 & 200 \end{bmatrix}^{-1} \begin{bmatrix} 400 \\ -200 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \begin{bmatrix} \frac{1}{402} & \frac{1}{201} \\ \frac{1}{201} & \frac{601}{40200} \end{bmatrix} \begin{bmatrix} 400 \\ -200 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} = x^*$

2. Let $S = \text{span}\{x_1, x_2, x_3\}$, where

$$x_1 = \begin{bmatrix} 2 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \qquad x_2 = \begin{bmatrix} 2 \\ 2 \\ 0 \\ -3 \end{bmatrix}, \qquad x_3 = \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix}.$$

Find an orthonormal basis for S, using Gram-Schmidt algorithm.

Solution.

$$\begin{aligned}
\tilde{q}_1 &= x_1 \\
q_1 &= \frac{\tilde{q}_1}{||\tilde{q}_1||} \\
||\tilde{q}_1|| &= \sqrt{4+1} \\
q_1 &= \begin{bmatrix} \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \\ 0 \\ 0 \end{bmatrix} \\
\tilde{q}_2 &= x_2 \; - \; (q_1^T x_2) q_1 \\
&= \begin{bmatrix} 2 \\ 2 \\ 0 \\ -3 \end{bmatrix} - (\begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \\ 0 \\ -3 \end{bmatrix}) \begin{bmatrix} \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \\ 0 \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} 2 \\ 2 \\ 0 \\ -3 \end{bmatrix} - \begin{bmatrix} \frac{12}{5} \\ \frac{6}{5} \\ 0 \\ 0 \end{bmatrix} \\
\tilde{q}_2 &= \begin{bmatrix} -\frac{2}{5} \\ \frac{4}{5} \\ 0 \\ -3 \end{bmatrix} \\
||\tilde{q}_2|| &= \sqrt{\frac{49}{5}} \\
q_2 &= \begin{bmatrix} \frac{-2}{5\sqrt{\frac{49}{5}}} \\ \frac{4}{5\sqrt{\frac{49}{5}}} \\ 0 \\ -\frac{3}{\sqrt{\frac{49}{5}}} \end{bmatrix}
\end{aligned}$$

$$\tilde{q}_3 = x_3 \; - \; (q_1^T x_3)q_1 \; - \; (q_2^T x_3)q_2$$

$$= \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix} - (\begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix}) \begin{bmatrix} \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \\ 0 \\ 0 \end{bmatrix} - (\begin{bmatrix} \frac{-2}{5\sqrt{\frac{49}{5}}} & \frac{4}{5\sqrt{\frac{49}{5}}} & 0 & -\frac{3}{\sqrt{\frac{49}{5}}} \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix}) \begin{bmatrix} \frac{-2}{5\sqrt{\frac{49}{5}}} \\ \frac{4}{5\sqrt{\frac{49}{5}}} \\ 0 \\ -\frac{3}{\sqrt{\frac{49}{5}}} \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ -1 \\ 1 \\ 0 \end{bmatrix} + (\frac{1}{\sqrt{5}}) \begin{bmatrix} \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \\ 0 \\ 0 \end{bmatrix} - (-\frac{4}{5\sqrt{\frac{49}{5}}}) \begin{bmatrix} \frac{-2}{5\sqrt{\frac{49}{5}}} \\ \frac{4}{5\sqrt{\frac{49}{5}}} \\ 0 \\ -\frac{3}{\sqrt{\frac{49}{5}}} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{2}{5} \\ -\frac{4}{5} \\ 1 \\ 0 \end{bmatrix} - \begin{bmatrix} \frac{8}{5(49)} \\ -\frac{16}{5(49)} \\ 0 \\ \frac{12}{49} \end{bmatrix}$$

$$\tilde{q}_3 = \begin{bmatrix} \frac{18}{49} \\ -\frac{36}{49} \\ 1 \\ -\frac{12}{49} \end{bmatrix}$$

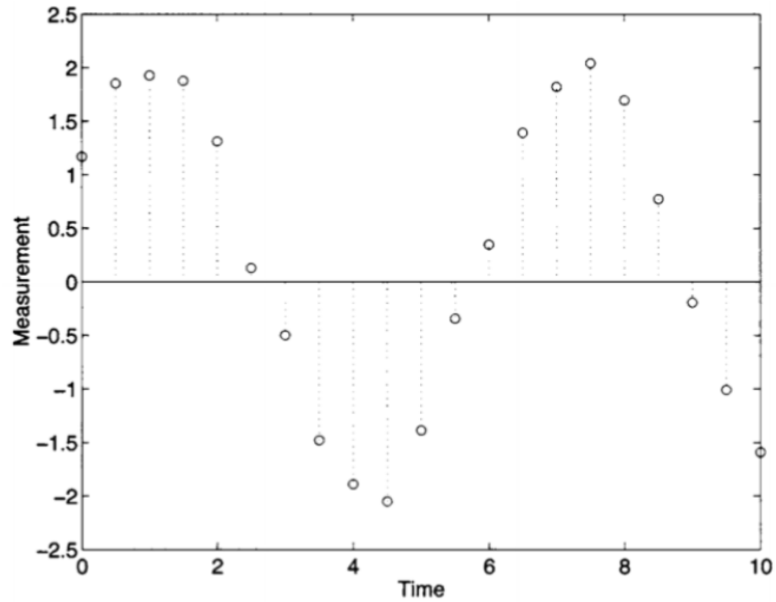
$$||\tilde{q}_3|| = \sqrt{\frac{85}{49}}$$

$$q_3 = \begin{bmatrix} \frac{18}{49\sqrt{\frac{85}{49}}} \\ -\frac{36}{49\sqrt{\frac{85}{49}}} \\ \frac{1}{\sqrt{\frac{85}{49}}} \\ -\frac{12}{49\sqrt{\frac{85}{49}}} \end{bmatrix}$$

3. Write a code and implement the Gauss-Newton Method on the last example given in the lecture, to find A, ω, and ϕ such that the resulting sinusoid

$$y = A\sin(\omega t + \phi)$$

best fits $(t_i, y_i), i = 1, 2, \ldots, 21$, with $t_1 = 0$ and $t_{21} = 10$ and y_i given roughly below.



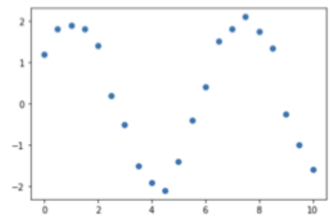
Solution.

The code and output for question 3 can be found below. I had some trouble with latex packages so I had to screenshot the code and put it into another pdf.

```
import numpy as np
import matplotlib.pyplot as plt
```

```
data=np.array([[0,1.2],[.5,1.8],[1,1.9],[1.5,1.8],[2,1.4],[2.5,.2],[3,-.5],[3.5,-1.5],[4,-1.9],[4.5,-2.1],[5,-1.4],[5.5,-.4],[6,.4],[6.5,1.5],[7,1.8],[7.5,2
```

```
#Visualizing data
plt.scatter(data[:,0],data[:,1])
plt.show()
```




```
def r_vec(A,w,phi,data):
  return data[:,1].reshape((len(data),1))-A*np.sin(w*data[:,0].reshape(len(data),1)+phi)

#Note: Xvec=[A,w,phi]^T
def J(xvec):
  drdA=-1*np.sin(data[:,0]*xvec[1][0]+xvec[2][0]).reshape((len(data),1))
  drdw=-1*np.multiply(xvec[0][0]*np.cos(data[:,0]*xvec[1][0]+xvec[2][0]),data[:,0]).reshape((len(data),1))
  drdphi=-1*xvec[0][0]*np.cos(data[:,0]*xvec[1][0]+xvec[2][0]).reshape((len(data),1))

  return np.concatenate((drdA,drdw,drdphi),axis=1)
```

```
#Note: x0 is a vector [A,w,phi]^T
def Gauss_Newton(r_vec,J,x0):
  xk=x0
  diff=.02
  while np.linalg.norm(diff)>.001:
    #print(xk,xk[0][0],xk[1][0],xk[2][0])
    xkp1=xk-np.matmul(np.matmul(np.linalg.inv(np.matmul(np.transpose(J(xk)),J(xk))),np.transpose(J(xk))),r_vec(xk[0][0],xk[1][0],xk[2][0],data=data))
    diff=xkp1-xk
    xk=xkp1

  return xkp1
```

```
b=Gauss_Newton(r_vec,J,x0=np.array([[4],[1],[1]]))
```

```
print(b)
```

```
[[2.05540401]
 [0.98675651]
 [0.54589851]]
```

```
#Plotting over scatterplot
plt.scatter(data[:,0],data[:,1])
plt.plot(data[:,0],b[0][0]*np.sin(b[1][0]*data[:,0]+b[2][0]),c='red')
plt.show()
```