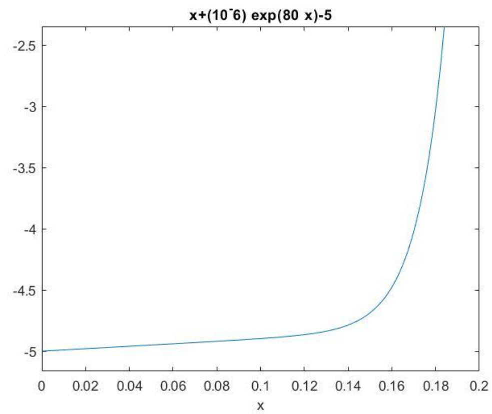
12.12.2018
Student: Orçun KÖK

HOMEWORK 3

Problem 1.

a. I have provided the MatLab script as *P1a.m* . Before any calculations I wanted to check the convergence which I had verified from the plot below,



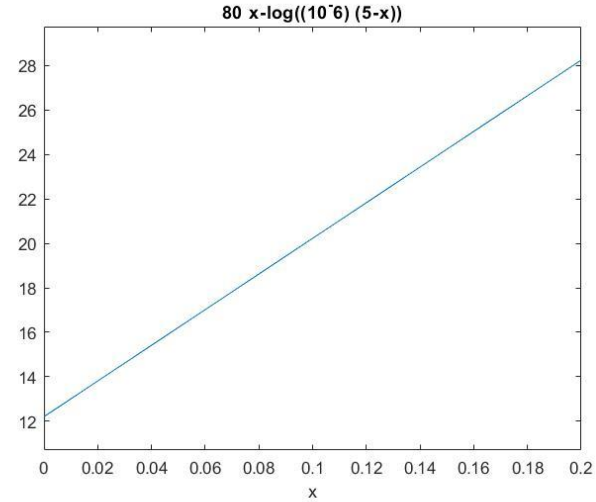
➔Once the Program is ran total number of iterations were ***390***.
My solution was exactly *0.192321560909437.* Putting this value to the function resulted in 9.226853617949615e-05 which is very close to 0 since I was expecting *F(x)=0*. After that I can determine if I am within the 10^{-6} of the exact solution with the absolute value of $(x_n - x_{n+1})$.

b. I have provided the MatLab script as *P1a.m* . It is easy to iterate the function for different values of I and initializing only. Below is the number of iteration for the each step of i.

i =	1	2	3	4	5	6	7	8	9	10
Iteration#	32	5	4	4	4	4	4	3	3	3

Total number of iterations are 66. Which is significantly less than the total number of iterations in part a.

c. I have provided the MatLab script as *P1c.m* . Also for this part I first checked the convergence with the below graph too.



➔It can be clearly seen that the problem is linear. Because of the linearity I can state that this problem is way easier to solve than the P1.a and P1.b. Since I know that Newton's method needs exactly 1 iteration for linear problems. My MatLab script result was *x=0.192331028309207* which holds with P1.a and P1.b.

Problem 2.

a. Because the calculations are relatively complex to show on a PDF file, I made the calculations by hand. Below are the hand-made calculations with the explanations.

2)

a) In matrix form, nodal equations are;

$$\underbrace{\begin{bmatrix}2&-1&&&\\-1&2&-1&&\\&&\ddots&&\\&&-1&2&-1\\&&&-1&2\end{bmatrix}}_{G}\underbrace{\begin{bmatrix}\psi_1\\ \vdots\\ \psi_N\end{bmatrix}}_{x} + \Delta x^2 \underbrace{\begin{bmatrix}e^{\psi_1}-e^{-\psi_1}\\ \vdots \\ e^{\psi_N}-e^{-\psi_N}\end{bmatrix}}_{H} = \underbrace{\begin{bmatrix}-V\\0\\0\\ \vdots\\0\\0\\V\end{bmatrix}}_{b}$$

- Let $F(x) = G(x) + H(x) - b$ (1)
- To show non-singularity, Taylor series expansion around x^0 is,

$$F(x') = G(x^0) + H(x^0) - b + J(x^0)(x' - x_0) \quad (2)$$

- Where Jacobian evaluated at x^0 is,

$$J(x^0) = G + \left[\frac{dH_i}{d\psi_j}\right]_{ij}\Bigg|_{\psi_i = \psi_j^0} \quad (3)$$

Keeping in mind $2\sinh x = e^x - e^{-x}$, writing $H(x)$,

$$H = 2(\Delta x)^2 \left[\sin \psi_i\right]_i \quad (4)$$

- Combine eq. 4 and eq. 3,

$$J_H(x^0) = \left[\frac{dH_i}{d\psi_j}\right]_{ij}\Bigg|_{\psi_i = \psi_j^0} = 2(\Delta x)^2 \left[\cosh \psi_i \, \delta_{ij}\right]_{ij}$$

- Hence, Jacobian is a sum of a constant matrix and a matrix with hyperbolic cosines along the diagonal,

$$J(x^0) = G + J_H(x^0) \qquad (5)$$

- Since it can be seen that matrix is strictly diagonaly dominant, every eigenvalue is real. For SDD, every eigenvalue of matrix $A_{N \times N}$ should satisfy,

$$|\lambda_i - A_{ii}| \leq \sum_{i \neq j} |A_{ij}| \qquad (6)$$

- For the first & last rows,

$$|\lambda_i - (2 + 2(\Delta x)^2 \cosh(\varphi_i)| < 2 \qquad (7)$$

$$2(\Delta x)^2 \cos(\varphi_i) < \lambda_i < 4 + 2(\Delta x)^2 \cos(\varphi_i) \qquad (7a)$$

- For the rest of the rows,

$$|\lambda_i - (2 + 2(\Delta x)^2 \cosh(\varphi_i)| < 1 \qquad (8)$$

$$1 + 2(\Delta x)^2 \cos(\varphi_i) < \lambda_i < 3 + 2(\Delta x)^2 \cos(\varphi_i) \qquad (8a)$$

- Since we know that $\cosh x \geq 1$ and $\Delta x \geq 0$, we can state that Jacobian is SDD for and λ_i. It is obvious that SDD matrices are non-singular. Therefore, damped Newton method should not get stuck around local minima. What is more that, from eq. 6 we can see that lower bound for any eigenvalue is $2(\Delta x)^2$ and we know that $\Delta x = 1/(N+1)$, So while not strictly singular anywhere, it may can be potentially near singular. And increasing the iteration may cause oscillatory behaviour.

b. For this part the MatLab code is provided as *P2.m* and *RHS.m* and comment for the necessary parts. Below are the MatLab output,

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
>> P2
Initial F = 1.41421
Iteration #1: ||dx|| = 5.52994, ||F|| = 0.000117637, a= 1
Iteration #2: ||dx|| = 0.0135486, ||F|| = 1.50786e-09, a= 1
Iteration #3: ||dx|| = 2.96714e-07, ||F|| = 6.52048e-16, a= 1
For V = 1:  # Iterations: 3, Residual evaluations: 4, # Lin sys solves: 3
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
>> P2
Initial F = 28.2843
Iteration #1: ||dx|| = 110.599, ||F|| = 55168.2, a= 1
Iteration #2: ||dx|| = 8.62595, ||F|| = 20294.6, a= 1
Iteration #3: ||dx|| = 8.21118, ||F|| = 7464.97, a= 1
Iteration #4: ||dx|| = 7.83169, ||F|| = 2744.94, a= 1
Iteration #5: ||dx|| = 7.45207, ||F|| = 1008.26, a= 1
Iteration #6: ||dx|| = 7.06208, ||F|| = 369.097, a= 1
Iteration #7: ||dx|| = 6.65442, ||F|| = 133.737, a= 1
Iteration #8: ||dx|| = 6.217, ||F|| = 47.0612, a= 1
Iteration #9: ||dx|| = 5.72633, ||F|| = 15.4108, a= 1
Iteration #10: ||dx|| = 5.14775, ||F|| = 4.44848, a= 1
Iteration #11: ||dx|| = 4.3901, ||F|| = 1.0919, a= 1
Iteration #12: ||dx|| = 3.05901, ||F|| = 0.17631, a= 1
Iteration #13: ||dx|| = 1.09178, ||F|| = 0.00880268, a= 1
Iteration #14: ||dx|| = 0.0895884, ||F|| = 3.15307e-05, a= 1
Iteration #15: ||dx|| = 0.000462205, ||F|| = 5.32103e-10, a= 1
Iteration #16: ||dx|| = 1.07687e-08, ||F|| = 5.07912e-15, a= 1
For V = 20:  # Iterations: 16, Residual evaluations: 17, # Lin sys solves: 16
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
>> P2
Initial F = 141.421
Iteration #1: ||dx|| = 552.994, ||F|| = 3.79415e+38, a= 1
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND
=  4.279881e-39.
> In P2 (line 20)

Iteration #2: ||dx|| = 9.6352, ||F|| = 1.39579e+38, a= 1
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND
=  1.152203e-38.
> In P2 (line 20)
```

….

Iteration #53: ||dx|| = 6.05672, ||F|| = 9.9038e+15, a= 1
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 1.482673e-16.
> In P2 (line 20)

Iteration #54: ||dx|| = 5.97232, ||F|| = 3.64341e+15, a= 1
…
Iteration #90: ||dx|| = 2.06447e-05, ||F|| = 7.50139e-11, a= 1
For V = 100: # Iterations: 90, Residual evaluations: 91, # Lin sys solves: 90

~~~~~~

➔It is very clear from inspecting the **||dx||** (norm delta fi) and **||F||** (norm residual) is that quadratic convergence can be seen at near solution. Obviously it can be seen that for V=20, it takes longer than V=1. On the other hand when V=100, in the first iteration of Newton Jacobian is singular. However MatLab has overcome with that.

c. For this part I modified the MatLab code for part *2.a* slightly to induce the boundaries for the damping method. I will be providing the same code where the damping parts are commented.

~~~~~~
>> P2
Initial F = 141.421
Iteration #1: ||dx|| = 552.994, ||F|| = 101.067, a= 0.125
Iteration #2: ||dx|| = 8.89622, ||F|| = 22.08, a= 0.5
Iteration #3: ||dx|| = 7.44634, ||F|| = 6.30691, a= 1
Iteration #4: ||dx|| = 6.60395, ||F|| = 1.84926, a= 1
Iteration #5: ||dx|| = 5.5548, ||F|| = 0.451334, a= 1
Iteration #6: ||dx|| = 3.71235, ||F|| = 0.0691419, a= 1
Iteration #7: ||dx|| = 1.1737, ||F|| = 0.00305243, a= 1
Iteration #8: ||dx|| = 0.0756883, ||F|| = 8.14604e-06, a= 1
Iteration #9: ||dx|| = 0.000252867, ||F|| = 7.02202e-11, a= 1
Iteration #10: ||dx|| = 2.5866e-09, ||F|| = 8.05199e-14, a= 1
For V = 100: # Iterations: 10, Residual evaluations: 15, # Lin sys solves: 10
~~~~~~
~~~~~~

~~~~~~

```
>> P2
Initial F = 5.65685e+15
Iteration #1: ||dx|| = 2.21197e+16, ||F|| = 5.65685e+15, a= 7.10543e-15
Iteration #2: ||dx|| = 5.01752e+07, ||F|| = 2.1094e+15, a= 4.76837e-07
….
Iteration #22: ||dx|| = 0.028546, ||F|| = 3.53553, a= 1
Iteration #23: ||dx|| = 5.44276e-05, ||F|| = 3.53553, a= 1
For V = 4e+15:  # Iterations: 23, Residual evaluations: 92, # Lin sys solves: 23
```

~~~~~~

➔With the help of damping method we eliminated the singularity problem that we received in the part *2.b*. Not only staying with that we reduced the residual evolutions significantly. Moreover damped Newton scheme also helped us to receive convergence for less than 200 residual evaluations for a very large $V=4*10^{15}$. It should be

Problem 3.

a. I have been able to track two bugs in the MatLab scripts. First and the most important one is that the some Jacobians were missing. It can also be seen with the test files that the convergence was not quadratic. Modified files are *force.m, newton.m and loadNewton.m.* Modified files are provided with the comments for the changes. Second problem occurred while using the test files. Apparently Jacobian created in the first iteration is a singular matrix. Warning was related to line 16 which was,

```
dx = (Matrix \ (-RHS' ))'
```

I used the pinv() function to solve the problem.

b. After implementing the changes I was able to see the convergence on the first two examples and a linear convergence on the test3. Because test3 took only one iteration and we know that the Newton's method solves linear problems in exactly one iteration.

Problem 4.

a. Because the calculations are relatively complex to show on a PDF file, I made the calculations by hand. Below are the hand-made calculations with the explanations.

3)

a. Let $F_1(x) = Av - \lambda v$ and $F_2(x) = v^T v - 1$ where $F(v, \lambda) = \begin{bmatrix} F_1(x) \\ F_2(x) \end{bmatrix}$

Then start implementing Jacobian with $F_1(x)$. Let $Av = b$ and $\lambda v = c$,

- $$A * v = b = \begin{bmatrix} A_{11}v_1 + A_{12}v_2 + \dots + A_{1N}v_N \\ A_{21}v_1 + A_{22}v_2 + \dots + A_{2N}v_N \\ \vdots \\ A_{N1}v_1 + A_{M2}v_2 + \dots + A_{MN}v_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix} \rightarrow b_i = A_{i1}v_1 + \dots + A_{iN}v_N$$

$$b_i = \sum_{j=1}^{N} A_{ij} v_j \quad (1)$$

- Since $Av - \lambda v = 0$, $(A - \lambda I)v = 0$; Let λv

$$\lambda * v = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & & \ddots & \\ 0 & \dots & & \lambda_M \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_M \end{bmatrix} = \begin{bmatrix} \lambda_1 v_1 \\ \lambda_2 v_2 \\ \vdots \\ \lambda_M v_N \end{bmatrix} \rightarrow c_j = \lambda_j v_j \quad (2)$$

- We know that the Jacobian matrix is in the form of,

$$J_F(x) = \begin{bmatrix} \frac{dF_{1_1}(x)}{dx_1} & \dots & \frac{dF_{1_M}(x)}{dx_M} \\ \vdots & & \vdots \\ \frac{dF_{N_1}(x)}{dx_1} & \dots & \frac{dF_{N_M}(x)}{dx_N} \end{bmatrix} \quad (3)$$

-1-

- Now looking at $F_2(v)$,

$$v^T v - 1 = v_1^2 + v_2^2 + \cdots + v_M^2 - 1 \qquad (4)$$

- Let's make $\lambda = v_{n+1}$, Jacobian of this vector with respect to expanded $+$ vector is,

$$J(F, v) = \begin{bmatrix} a_{11} - \lambda & a_{12} & \cdots & a_{1N} & -v_1 \\ \vdots & & & & \\ a_{N1} & a_{N2} & \cdots & a_{NN} - \lambda & -v_N \\ 2v_1 & 2v_2 & \cdots & 2v_N & 0 \end{bmatrix} \qquad (5)$$

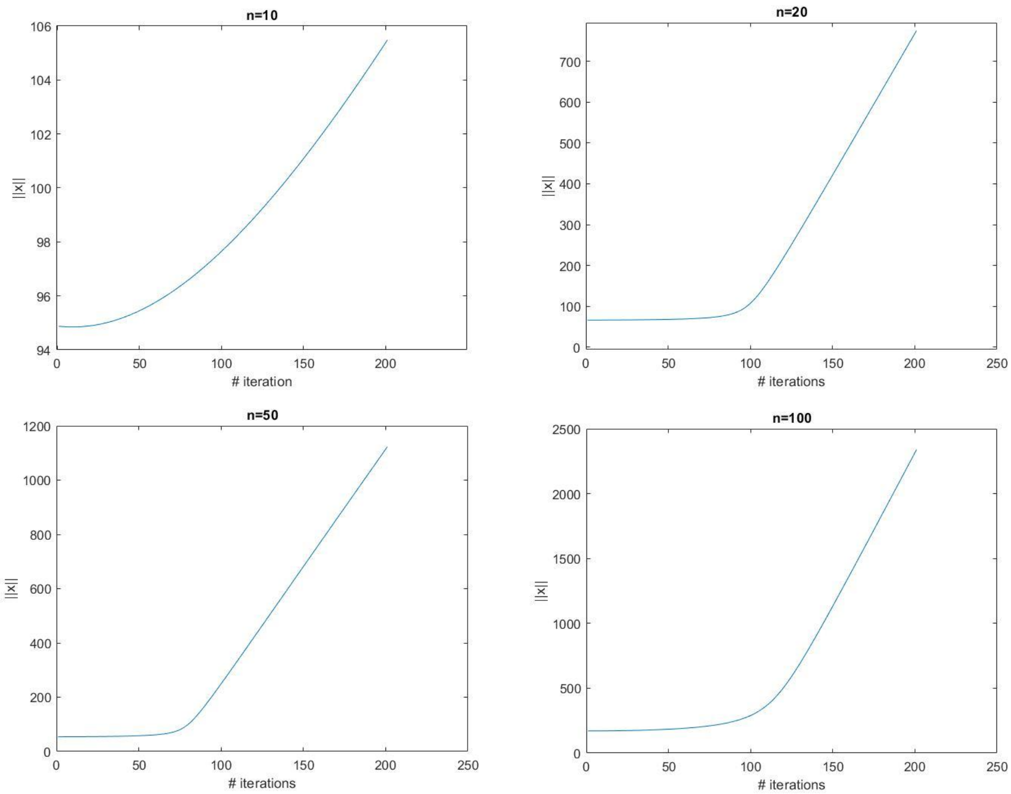
b) To implement standart Newton's method,

$$F = \begin{bmatrix} a_{11} v_1 + a_{12} v_2 + \cdots + a_{1N} v_M - \lambda v_1 \\ \vdots \\ a_{N1} v_1 + a_{N2} v_2 + \cdots + a_{NN} v_N - \lambda v_N \\ v_1^2 + v_2^2 + \cdots + v_N^2 - 1 \end{bmatrix} \qquad (6)$$

- Now solving below equation in Matlab. Matlab file is provided as P4.m,

$$v_{n+1} = v_n - J(F, v)^{-1} F \qquad (7)$$

b. MatLab script is provided as P4.m and P4a.m with necessary comments. Below are the plots for nxn size matrix with n=10, 20, 50,100. Every other condition are same among them.



These plots are the # of iterations vs norm of the eigenvector in the corresponding iteration. It can be clearly seen that, increasing the size of matrix results in a more aggressive convergence. Furthermore it also increases the norm of the vector. We can conclude that It gets harder and results in more error working with larger matrices.

Problem 5.

MatLab script is provided as P5 with necessary comments. This problem has not been finished so there are no analyzing.