Class: MATH 466 Numerical Methods
Professor: Eric Olsen
Team 1:

Michael Dorado
Maverick Marsala
Matthew Neill
Adrian Samberg

Date: 12/7/21
Assignment: Project 2

Problem 1:

a) This section used an input matrix A and an iteration defined as follows, respectively:

$$X_{k+1} = \frac{1}{2}\left(X_k + \left(X_k^{-1}\right)^T\right) \qquad \text{where} \qquad X_0 = A.$$

$$A = \begin{bmatrix} -0.49 & -0.21 & -0.40 & -0.21 \\ 0.36 & 0.10 & 0.29 & -0.04 \\ 0.12 & -0.01 & 0.48 & -0.47 \\ 0.09 & 0.09 & -0.41 & 0.22 \end{bmatrix}.$$

By making X0 equal to matrix and solving X1 for the first iteration, we can compute the Frobenius norm is 16.3705, confirming the matrix is correctly entered and the iteration is being calculated correctly.

Code:

```
#***Problem 1a***

println("Problem 1a: ")
A = [-0.49 -0.21 -0.40 -0.21
    0.36 0.10 0.29 -0.04
    0.12 -0.01 0.48 -0.47
    0.09 0.09 -0.41 0.22]

X0 = copy(A)

X1 = 0.5*(X0+transpose(inv(X0)))

print("Frobenius Norm: ")
println(norm(X0-X1)) #Want this to be 16.37054203598731
print("\n")
```

Output:

```
julia> include("Math466_Proj2.jl")
Problem 1a:
Frobenius Norm: 16.37054203598732
```

b) In this section we perform ten iterations (zero through nine) of the matrix operation defined in part a. We then create the vector delta to track the Frobenius norms for each iteration as well. We can see that these Frobenius norms approach zero as it iterates through the operation.

Code:

```
#***Problem 1b***
println("Problem 1b: ")
#Now Make a vector v, each entry is a matrix with the (n-1)th iteration.
#v[1] = X0 etc.
v = Matrix{Float64}[]
push!(v, X0)
v[1] # X0

for i=2:11
    temp = 0.5*(v[i-1]+transpose(inv(v[i-1])))
    push!(v, temp)
    #display(i-1)
    #display(v[i])
end

delta = zeros(10) #values of the delta k's (delta[1] = delta 0 )etc.

for i=1:10


    delta[i] = norm(v[i+1]-v[i])
    print("Delta, index ")
    print(i-1)
    print(", is = ")
    display(delta[i])
end
print("\n")
```

Output:

```
Problem 1b:
Delta, index 0, is = 16.37054203598732
Delta, index 1, is = 8.154659273555275
Delta, index 2, is = 4.029661257669982
Delta, index 3, is = 1.9461321661628397
Delta, index 4, is = 0.8646228972076174
Delta, index 5, is = 0.28274913277050345
Delta, index 6, is = 0.038465495535442644
Delta, index 7, is = 0.0007392504801242873
Delta, index 8, is = 2.7324556151232616e-7
Delta, index 9, is = 3.734666635696956e-14
```

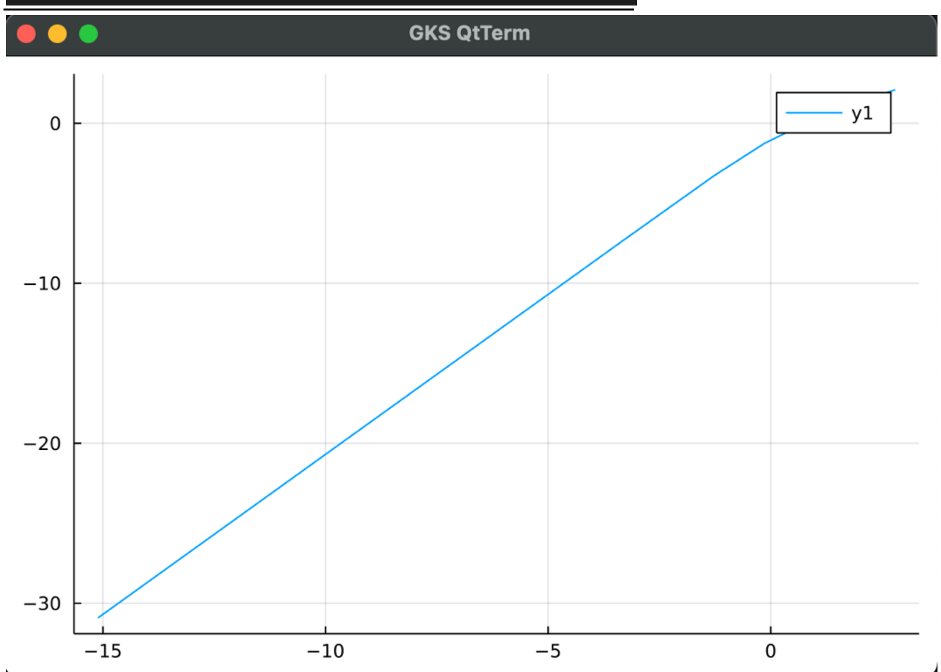
c) In this section we calculate plot, on a logarithmic scale, sequential delta values. If the graph is linear, it shows that we have a quadratic order convergence. We verify this by computing alpha, which is the slope between the last two points on the graph. We see that alpha is 2, so we confirm quadratic convergence that we see in the plot.

<u>Code:</u>

```
#***Problem 1c***
# need to plot (delta[i],delta[i+1]) for i=2:9 (which is k=1:8)

println("Problem 1c: ")
alpha = (log(delta[10])-log(delta[9]))/(log(delta[9])-log(delta[8]))
deltaX = zeros(9)
deltaY = zeros(9)
for i=1:9
    global deltaX, delta
    deltaX[i] = delta[i]
end
#println(deltaX)

for i=1:9
    global deltaY, delta
    deltaY[i] = delta[i+1]
end
#println(deltaY)

display(plot(log.(deltaX),log.(deltaY)))

print("Alpha is = ")
println(alpha)
print("\n")
```

<u>Output:</u>

```
Problem 1c:
Alpha is = 1.9999488019800933
```



d) This section confirms that W is orthogonal because $X^T * X$ converges towards the identity matrix.

Code:

```
#***Problem 1d***

println("Problem 1d: ")
println("K = 8: ")
display(v[9]'*v[9])
println("K = 9: ")
display(v[10]'*v[10])
println("K = 10: ")
display(v[11]'*v[11])
print("\n")
```

Output:

```
Problem 1d:
K = 8:
4×4 Matrix{Float64}:
  1.0          -1.88371e-7  -1.04753e-8   1.80024e-8
 -1.88371e-7   1.0           2.58256e-8  -4.43827e-8
 -1.04753e-8   2.58256e-8   1.0          -2.46812e-9
  1.80024e-8  -4.43827e-8  -2.46812e-9   1.0
K = 9:
4×4 Matrix{Float64}:
  1.0           -2.57131e-14  -1.41068e-15   2.46364e-15
 -2.57131e-14   1.0            3.54173e-15  -6.04376e-15
 -1.41068e-15   3.54173e-15   1.0           -3.81e-16
  2.46364e-15  -6.04376e-15  -3.81e-16      1.0
K = 10:
4×4 Matrix{Float64}:
 1.0           1.34121e-16   3.76323e-17   9.75894e-18
 1.34121e-16   1.0          -3.37948e-17  -4.60842e-17
 3.76323e-17  -3.37948e-17   1.0           2.65522e-17
 9.75894e-18  -4.60842e-17   2.65522e-17   1.0
```

e) In this section we found that some of the eigenvalues of A are negative and some of them are imaginary. In the matrix P we see that all the eigenvalues are positive and real.

Code:

```
#***Problem 1e***
println("Problem 1e: ")
W = v[11]

Winv  = inv(W)

P = Winv * A
println("Eigenvalues of A are: ")
display(eigvals(A))
println("\nEigenvalues of P are: ")
display(eigvals(P))
```

Output:

```
Problem 1e:
Eigenvalues of A are:
4-element Vector{ComplexF64}:
 -0.20699017509306217 - 0.2067450022872031im
 -0.20699017509306217 + 0.2067450022872031im
 -0.05873738107811822 + 0.0im
   0.7827177312642422 + 0.0im

Eigenvalues of P are:
4-element Vector{Float64}:
 0.030861372732592457
 0.1989032921153495
 0.664894706530955
 0.9641058475529806
```