**Homework 3**

Due: February 1, 2021

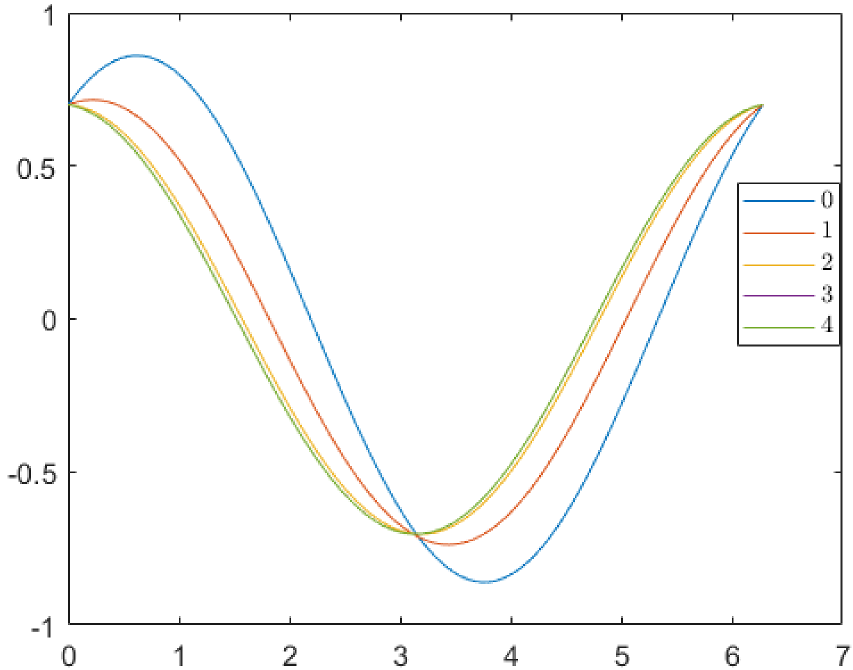
1. (Nonlinear pendulum)

   (a) Write a program to solve the boundary value problem for the nonlinear pendulum as discussed in the text. See if you can find yet another solution for the boundary conditions illustrated in Figures 2.4 and 2.5.

   (b) Find a numerical solution to this BVP with the same general behavior as seen in Figure 2.5 for the case of a longer time interval, say, $T = 20$, again with $\alpha = \beta = 0.7$. Try larger values of $T$. What does $\max_i \theta_i$ approach as $T$ is increased? Note that for large $T$ this solution exhibits "boundary layers".

**Solution.**

```
clear all;close all;
T=2*pi;
m=500;
h=T/(m+1);
t=0:h:T;
t=t';
theta=0.7*cos(t)+0.5*sin(t); % initial condition
J=zeros(m,m);
g=zeros(m+2,1);
plot(t,theta);
hold on
for k=1:4
    J(1,2)=1;
    J(m,m-1)=1;
    for j=1:m
        J(j,j)=-2+h^2*cos(theta(j+1));
    end
    for j=2:m-1
        J(j,j-1)=1;
        J(j,j+1)=1;
    end
    J=J/h^2;
    for i=2:m+1
        g(i)=(theta(i-1)-2*theta(i)+theta(i+1))/h^2+sin(theta(i));
    end
    G=g(2:m+1);
    Δ=J\(-G);
    theta(2:m+1)=theta(2:m+1)+Δ;
    plot(t,theta);
end
set(gca,'Fontsize',[12])
set(gca,'TickLabelInterpreter', 'tex');
legend('$0$','$1$','$2$','$3$','$4$','interpreter','latex')
```



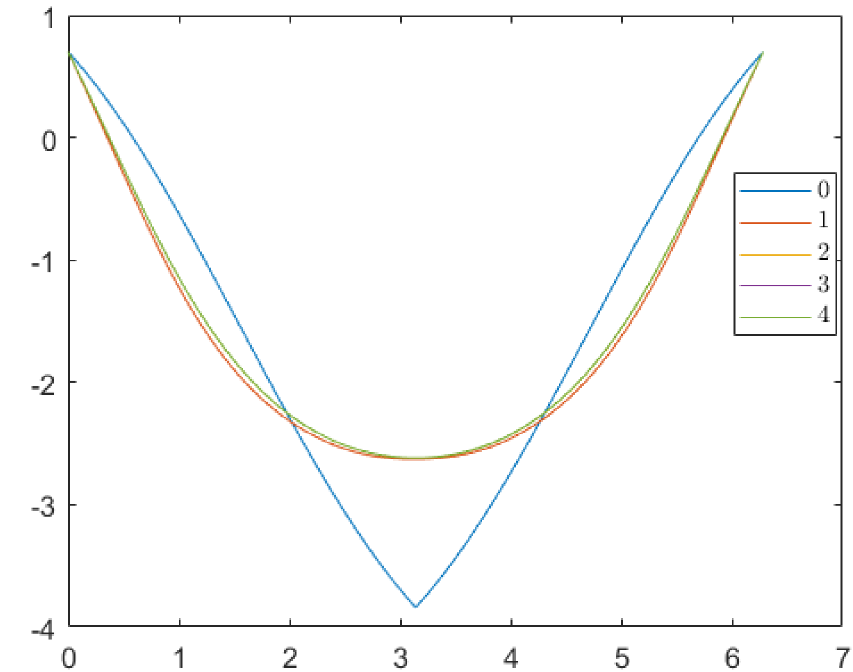




Figure 1: Solution to the pendulum problem. Top: $\theta_i^{[0]} = 0.7\cos(t_i) + 0.5\sin(t_i)$. Bottom: $\theta_i^{[0]} = 0.7\cos(t_i) + |t_i - \pi| - \pi$

.

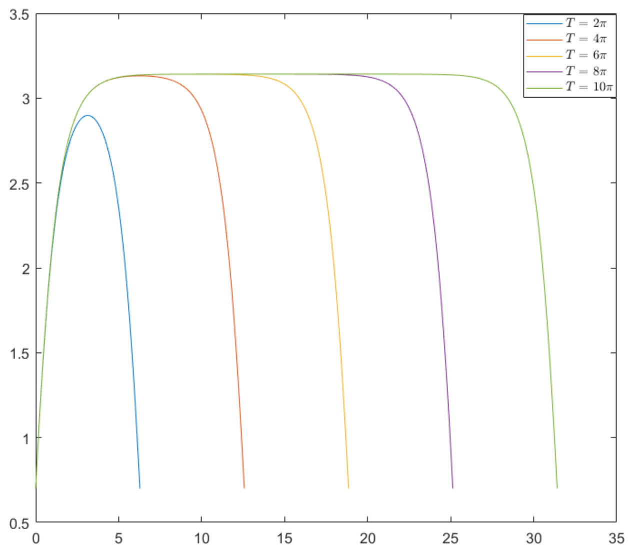



Figure 2: Solution to the pendulum problem with $\theta_i^{[0]} = 0.7 + \sin(t_i/2)$ for $T = 2\pi$. For $T > 2\pi$, we use the solution at the previous $T$ as our initial condition. I notice that if you simply use $\theta_i^{[0]} = 0.7 + \sin(t_i/2)$ for all $T$, the solution might not converge.

.

| $T$ | $\max_i \theta_i$ |
|---|---|
| $2\pi$ | 2.897573207012707 |
| $4\pi$ | 3.131139869459985 |
| $6\pi$ | 3.141140773005563 |
| $8\pi$ | 3.141573110263258 |
| $10\pi$ | 3.141591807934075 |
| $12\pi$ | 3.141592616974510 |
| $14\pi$ | 3.141592652003227 |
| $16\pi$ | 3.141592653520986 |

Table 1: It is evident that $\max_i \theta_i$ approaches to $\pi$ as $T$ is increased. This means that the pendulum has to stay vertical before it falls back to $\theta = 0.7$ at $T$.

2. (Gerschgorin's theorem and stability) Consider the boundary value problem

$$-u_{xx} + (1+x^2)u = f, \quad 0 \le x \le 1,$$

$$u(0) = 0, \quad u(1) = 0.$$

On a uniform grid with spacing $h = 1/(m+1)$, the following set of difference equations has local truncation error $O(h^2)$:

$$\frac{2u_i - u_{i+1} - u_{i-1}}{h^2} + (1+x_i^2)u_i = f(x_i), \quad i = 1, \ldots, m.$$

   (a) Use Gerschgorin's theorem to determine upper and lower bounds on the eigenvalues of the coefficient matrix for this set of difference equations.

   (b) Show that the $L_2$-norm of the *global error* is of the same order as the local truncation error.

**Solution.**

(a) By Gerschgorin's theorem, $\lambda \in \cup_i R_i$ where $R_i = \{z : |z - a_{ii}| \le \sum_j a_{ij}\}$. Since $A$ is a symmetric matrix, all eigenvalues are real. We can write

$$R_i = [a_{ii} - \sum_j |a_{ij}|, a_{ii} + \sum_j |a_{ij}|]$$
$$a_{ii} = \frac{2}{h^2} + 1 + x_i^2 \quad i = 1, \ldots, m$$
$$a_{i,i-1} = a_{i,i+1} = -\frac{1}{h^2} \quad i = 2, \ldots, m-1$$
$$a_{12} = a_{m,m-1} = -\frac{1}{h^2}$$

Then for $i = 2, \ldots, m-1$

$$R_i = [1 + x_i^2, \frac{4}{h^2} + 1 + x_i^2]$$
$$= [1 + (ih)^2, \frac{4}{h^2} + 1 + (ih)^2]$$

In particular

$$R_2 = [1 + (2h)^2, \frac{4}{h^2} + 1 + (2h)^2]$$
$$R_{m-1} = [1 + ((m-1)h)^2, \frac{4}{h^2} + 1 + ((m-1)h)^2]$$

For $i = 1, m$

$$R_1 = [\frac{1}{h^2} + 1 + h^2, \frac{3}{h^2} + 1 + h^2]$$
$$R_m = [\frac{1}{h^2} + 1 + (mh)^2, \frac{3}{h^2} + 1 + (mh)^2]$$

Since $\lambda \in \cup_i R_i$

$$\lambda \in [1 + (2h)^2, \frac{4}{h^2} + 1 + ((m-1)h)^2]$$

(b) We know that the global error is related to the local truncation error by

$$\|e\|_2 \leq \|A^{-1}\|_2 \|\tau\|_2$$

where $\|A^{-1}\|_2\| = \max_{1\leq p\leq m} |\lambda_p^{-1}|$. From (a) we know that $\lambda > 1$, then $\lambda^-1 < 1$. Thus $\|A^{-1}\|_2 < 1$ and global error is of the same order as the local truncation error.

3. (Richardson extrapolation) Use your code from problem 6 in assignment 1, or download the code from the course web page to do the following exercise. Run the code with $h = .1$ (10 subintervals) and with $h = .05$ (20 subintervals) and apply Richardson extrapolation to obtain more accurate solution values on the coarser grid. Record the $L_2$-norm or the $\infty$-norm of the error in the approximation obtained with each $h$ value and in that obtained with extrapolation.

   Suppose you assume that the coarse grid approximation is piecewise linear, so that the approximation at the midpoint of each subinterval is the average of the values at the two endpoints. Can one use Richardson extrapolation with the fine grid approximation and these interpolated values on the coarse grid to obtain a more accurate approximation at these points? Explain why or why not?

   **Solution.**

| $h$ | Error |
|---|---|
| 0.1 | 1.7564e-03 |
| 0.05 | 4.2625e-04 |
| Richardson | 5.4607e-06 |

Suppose we have a finite difference scheme $\tilde{u}(h)$ to $u(x)$ with $n$-th order accuracy

$$u(x) = \tilde{u}(h) + Ch^n + O(h^{n+1}) \tag{1}$$

Reducing $h$ to half, we have

$$u(x) = \tilde{u}(h/2) + C(h/2)^n + O(h^{n+1}) \tag{2}$$

The standard way of doing Richardson extrapolation is to multiply (2) by $2^n$, subtract (1) and divide by $2^n - 1$

$$u(x) = \frac{2^n\tilde{u}(h/2) - \tilde{u}(h)}{2^n - 1} + O(h^n)$$

Thus the order of accuracy is increase by 1. However, in general $C$ is a function of $x$. If we were to do the linear interpolation with

$$\begin{aligned} u(x) &= \frac{\tilde{u}(x+\frac{h}{2}) + \tilde{u}(x-\frac{h}{2})}{2} \\ &= \frac{2u(x) + C(x+\frac{h}{2})h^n + C(x-\frac{h}{2})h^n + O(h^{n+1})}{2} \end{aligned}$$

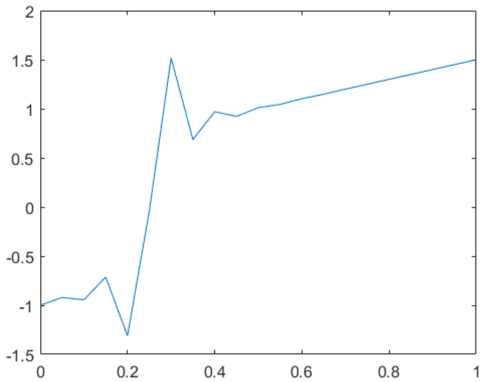
Then we can only do Richard extrapolation if $\frac{C(x+\frac{h}{2})+C(x-\frac{h}{2})}{2}$ is equal to $C$ in the fine grid approximation.

4. Write down the Jacobian matrix associated with Example 2.2 and the nonlinear difference equations (2.106) on p. 49. Write a code to solve these difference equations when $a = 0$, $b = 1$, $\alpha = -1$, $\beta = 1.5$, and $\epsilon = 0.01$. Use an initial guess of the sort suggested in the text. Try, say, $h = 1/20$, $h = 1/40$, $h = 1/80$, and $h = 1/160$, and turn in a plot of your results.

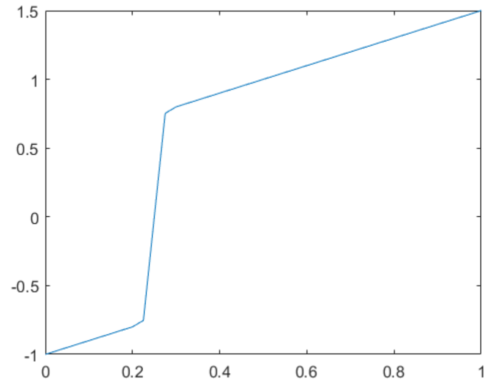
**Solution.**

$$J_{ij}(U) = \begin{cases} -\frac{2\epsilon}{h^2} + \frac{U_{i+1}-U_{i-1}}{2h} - 1 & j = i \\ \frac{\epsilon}{h^2} - \frac{U_i}{2h} & j = i-1 \\ \frac{\epsilon}{h^2} + \frac{U_i}{2h} & j = i+1 \\ 0 & \text{otherwise} \end{cases}$$

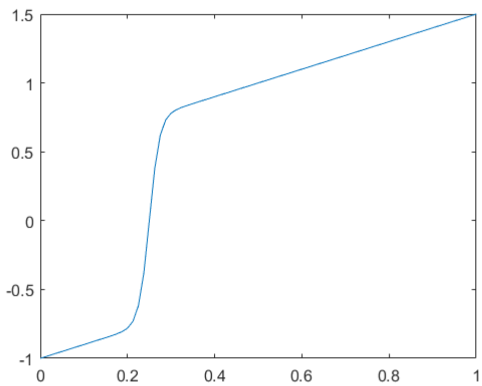
```
clear all;close all;
a=0;b=1;alpha=-1;beta=1.5;epsilon=0.01;
xbar=(a+b-alpha-beta)/2;
w0=(a-b+beta-alpha)/2;
h=1/20;
m=(b-a)/h-1;
x=0:h:1;
x=x';
u=x-xbar+w0*tanh(w0*(x-xbar)/(2*epsilon)); % initial condition
j=zeros(m+2,m+2);
g=zeros(m+2,1);
Δ=ones(m,1);
while max(abs(Δ))>1e-8 % stop loop at tolerance
    for i=2:m+1
        j(i,i)=-2*epsilon/h^2+(u(i+1)-u(i-1))/(2*h)-1;
        j(i,i-1)=epsilon/h^2-u(i)/(2*h);
        j(i,i+1)=epsilon/h^2+u(i)/(2*h);
    end
    J=j(2:m+1,2:m+1);
    for i=2:m+1
        g(i)=epsilon*(u(i-1)+u(i+1)-2*u(i))/h^2+u(i)*...
        ((u(i+1)-u(i-1))/(2*h)-1);
    end
    G=g(2:m+1);
    Δ=J\(-G);
    u(2:m+1)=u(2:m+1)+Δ;
end
plot(x,u)
set(gca,'Fontsize',[12])
set(gca,'TickLabelInterpreter', 'tex');
```



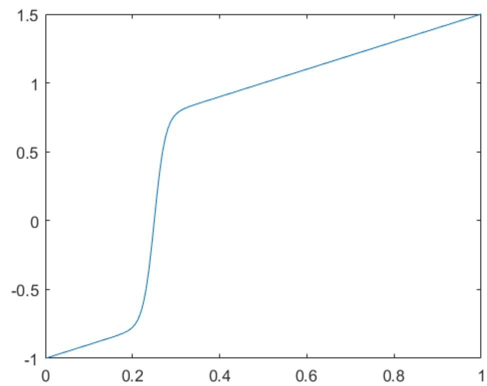

(a) $h = 1/20$




(b) $h = 1/40$




(c) $h = 1/80$




(d) $h = 1/160$

Figure 3: Solution to Example 2.2 with different mesh width.