MAT 425: Homework 7 (03/23)

Ex 1. [4pts]

a) The general solution of the ODE: $y'' + 5y = 0$ is given by:

$$y(x) = A\cos\sqrt{5}x + B\sin\sqrt{5}x.$$ .5pt

Since $y(0) = 0$, we have $A = 0$. Therefore, $y(x) = B\sin\sqrt{5}x$. Thus, we have: .5pt

i) no solution: if the right boundary condition is for example $y(\pi/\sqrt{5}) = 1$. .5pt

ii) one solution: if $y(\pi/2\sqrt{5}) = 1$ (that gives $B = 1$). .5pt

iii) infinitely many solutions: if $y(\pi/\sqrt{5}) = 0$ (B can be anything). .5pt

b) Solving the ODE $y'' - 5y = 0$ yields:

$$y(x) = Ae^{\sqrt{5}x} + Be^{-\sqrt{5}x}.$$ .5pt

The boundary condition leads to the linear system:

$$\begin{cases} A + B = 0 \\ Ae^{\sqrt{5}b} + Be^{-\sqrt{5}b} = \beta. \end{cases}$$

There always exists a unique solution since the matrix $A = \begin{bmatrix} 1 & 1 \\ e^{\sqrt{5}b} & e^{-\sqrt{5}b} \end{bmatrix}$ is .5pt
invertible ($b \neq 0$).

The difference is that in a), we have $p(x) = -1 < 0$. Thus, there is no guarantee of .5pt
existence and uniqueness of solution (the general theorem does not apply). Whereas in b), $p(x) = 1 > 0$ therefore we know (theorem) there exists a unique solution.

Ex 2. [3pts]

See page 5 for the implementation of the linear shooting-method. 2pts

Application. In figure 1-left, we plot the numerical solutions using $N = 10^3$ of the BVP:

$$\begin{cases} y'' = 4(y - x) \\ y(0) = 0\ ,\ y(1) = 2. \end{cases}$$ 1pt

and the exact solution. The two curves are in excellent agreement.

To estimate the accuracy of the method, we compute the solution for different values of $\Delta x = \frac{1}{N}$ with $N = [10, 30, 70, 200, 550, 1500, 4000, 10000]$. Then, we estimate the

difference between the numerical solution and the exact solution. We use for example the L^∞ norm:

$$\text{error}(\Delta x) = \max_i |y_{exact}(x_i) - y_i|.$$

We plot in figure 1-right the result in loglog scale. The error is decaying with Δx up to $\Delta x \approx 10^{-3}$ where the error reaches a plateau around 10^{-14} (near machine precision). To estimate the accuracy of the method, we perform a linear regression with the first 6 points. We obtain a slope of: $c = 3.9383$ which is close to the theoretical value of 4.

Ex 3. [3pts] 2pts

See page 5 for the implementation of the shooting-method with the secant method.

We use $N = 1000$ grid points. Starting with the slope: $s_0 = 0$ and $s_1 = 1$, we obtain the following estimations using the secant method:

.5pt

s_n	$\lvert y(b) - \beta \rvert$	$\lVert y - y_{exact} \rVert_\infty$
0	$2.2 \cdot 10^{-1}$	$2.2 \cdot 10^{-1}$
1	$1.1 \cdot 10^{0}$	$1.1 \cdot 10^{0}$
-0.249652477692151	$3.1 \cdot 10^{-4}$	$3.1 \cdot 10^{-4}$
-0.250003674168686	$3.3 \cdot 10^{-6}$	$3.3 \cdot 10^{-6}$
-0.249999999926475	$6.6 \cdot 10^{-11}$	$6.6 \cdot 10^{-11}$
-0.250000000000004	$3.9 \cdot 10^{-16}$	$7.2 \cdot 10^{-16}$

Thus, with a tolerance of 10^{-3}, the algorithm stops after 2 iterations. If we keep going, the algorithm reaches machine precision after 5 iterations. We plot in figure 2 the solution after 2 iterations along with the exact solution. .5pt

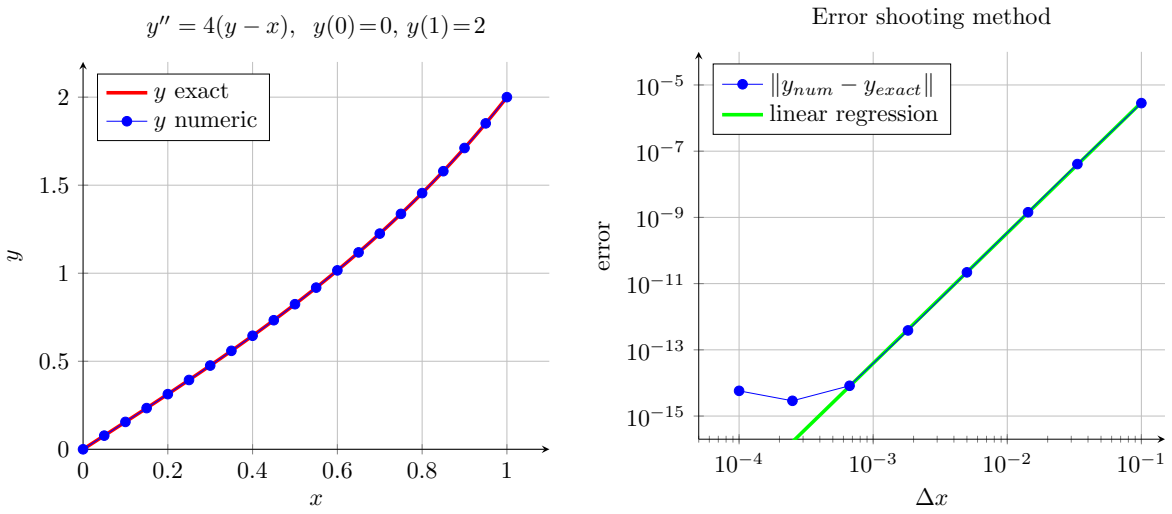

Figure 1: **Left**: The solution of the boundary value problem (blue) and the exact solution (green). The two curves are in excellent agreement. Numerical parameter: $N = 100$. **Right**: The error between the numerical solution $y(x)$ and the exact solution y_{exact} for different values of Δx in loglog scale. We observe that the shooting method with RK4 is 4th order accurate (check the linear regression in green).

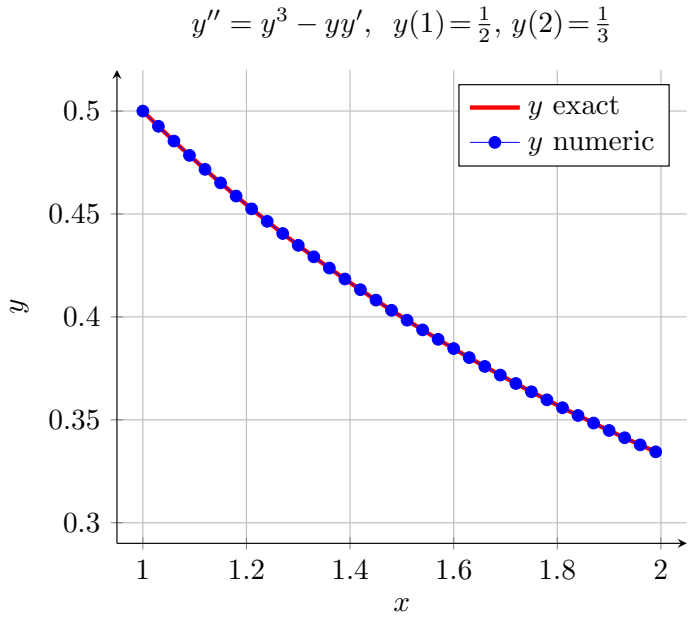

Figure 2: Numerical solution of the non-linear boundary value problems and the exact solution.

```
%% Solve the linear BVP
%%---------------------
%%     y'' = p(x) y' + q(x) y + r(x)
%%     y(0)= alpha, y(b)= beta
%% y'' = p(x) y' + q(x) y + r(x)
p = @(x) 0;
q = @(x) 4;
r = @(x) -4*x;
%%   y(0)= alpha, y(b)= beta
a     = 0;
b     = 1;
alpha = 0;
beta  = 2;
%% Numerical Parameters
N = 100;
%%--------------- solve 2 IVP
f1 = @(x,z) [z(2); p(x)*z(2) + q(x)*z(1) + r(x)];
f2 = @(x,z) [z(2); p(x)*z(2) + q(x)*z(1)];
s1 = 0;
s2 = 1;
%% Runge-Kutta 4:
[z1,x] = RK4(f1,a,[alpha;s1],b,1/N);
[z2,x] = RK4(f2,a,[0;s2]    ,b,1/N);
%% The solution
C = (beta-z1(1,end))/z2(1,end);
y = z1(1,:) + C*z2(1,:);
%%--------- plot
yExact = exp(2)/(exp(4)-1)*(exp(2*x)-exp(-2*x)) + x;
plot(x,y,'linewidth',5,
     x,yExact,'linewidth',5)
xlabel('x')
ylabel('y')
legend('y numeric','y exact','location','northwest')
title('y''=p y + q y' + r, y(0)=a,y(b)=beta')
```

```
%% Solve the non-linear BVP
%%-------------------------
%%   y'' = y^3-yy'
%%   y(1)=1/2, y(2)=1/3
f = @(x,z) [z(2);z(1)^3-z(1)*z(2)];
a = 1; alpha = 1/2;
b = 2; beta  = 1/3;
%% Parameters
s0 = 0;
s1 = 1;
N  = 1000;
nbIter = 5;
%% save all?
saveAllfunctions = 1;
%% Initialization
[z,x] = RK4(f,a,[alpha;s0],b,1/N);
y_b_km1 = z(1,end);
s_k     = s1;
s_km1   = s0;
if (saveAllfunctions==1)
  stockY = zeros(nbIter+1,N+1);
  stockS = zeros(1,nbIter+1);
  stockY(1,:) = z(1,:);
  stockS(1)   = s0;
end
%% error
yExact = @(x) 1./(1+x);
stockError = zeros(1,nbIter+1);
stockError(1) = max(z(1,:)-yExact(x));

%%---------------------------%%
%%-----       loop        ---%%
%%---------------------------%%
for k=1:nbIter
  %% Solve a new IVP
  [z,x] = RK4(f,a,[alpha,s_k],b,1/N);
  y_b_k = z(1,end);
  %% the new s (secant method)
  s = s_k - (y_b_k - beta)/(y_b_k-y_b_km1)*(s_k-s_km1);
  %% save
  if (saveAllfunctions==1)
    stockY(k+1,:) = z(1,:);
    stockS(k+1)   = s_k;
  end
```

```
  stockError(k+1) = max(abs(z(1,:)-yExact(x)));
  %% update
  s_km1   = s_k;
  s_k     = s;
  y_b_km1 = y_b_k;
end
%%--------------------------%%
%%---- plot
plot(x,stockY(3,:),'linewidth',5,
     x,yExact(x),'linewidth',5)
xlabel('x')
ylabel('y')
legend('y numeric','y exact','location','northeast')
title('y''=y^3-yy', y(1)=1/2,y(2)=1/3')
```