Math 337 Homework 04

Andy Reagan

February 13, 2014

1. For the Modified Euler method, obtain Equations 4.21 and 4.22. Then plot the stability region boundary given by 4.22.

   Solution: Writing out the ME method as a difference equation, we have:

$$Y_{i+1} = Y_i + \frac{1}{2}hf(x_i, Y_i) + \frac{1}{2}hf(x_i + h, Y_i + hf(x_i, Y_i)). \tag{1}$$

   Plugging in the model problem, $f(x, y) = \lambda y$, we have the difference equation

$$\begin{aligned} Y_{i+1} &= Y_i + \frac{1}{2}h\lambda Y_i + \frac{1}{2}h\lambda Y_i + \frac{1}{2}h\lambda hf(x_i, Y_i)) && (2)\\ &= Y_i + h\lambda Y_i + \frac{1}{2}\lambda h^2 \lambda Y_i && (3)\\ &= Y_i + h\lambda Y_i + \frac{1}{2}(h\lambda)^2 Y_i && (4)\\ &= Y_i \left(1 + h\lambda + \frac{1}{2}(h\lambda)^2\right). && (5)\end{aligned}$$

   The above Equation 5 is the desired form of Equation 4.21 from the notes. Since this equation approximates the growth rate of the error, for the method to be stable we require the RHS be less than one in magnitude (modulus), and for complex λ we write the boundary of this region this as

$$\left|1 + h\lambda + \frac{1}{2}(h\lambda)^2\right| = 1. \tag{6}$$

   Writting $\lambda = \lambda_R + i\lambda_I$ we have, equivalently,

$$\left(1 + h\lambda_R + \frac{1}{2}(h\lambda_R)^2 - \frac{1}{2}(h\lambda_I)^2\right) + \left(h\lambda_I + h^2\lambda_I\lambda_R\right)^2 = 1. \tag{7}$$

   The following code makes a plot:

```
% make a contour plot of the stability region of ME

numz = 100;
numr = 3;
x = linspace(-numr,numr,numz); % real
y = linspace(-numr,numr,numz); % imag
[xz,yz] = meshgrid(x,y);
```

```
h = 1;
stable_region = (1+h.*xz+0.5.*((h.*xz).^2-(h.*yz).^2)).^2+(h.*yz+h^2.*yz.*xz).^2;
% or do it with a complex number
% z = xz+1i*yz;
% stable_region = abs(1+z*h+0.5*z.^2*h^2);

hf = figure;
contour(xz,yz,stable_region,[1,1],'k');
% set(gca,'XTick',1:(numz/12):numz)
% set(gca,'XTickLabel',x(1:(numz/12):end))
grid on;
xlabel('h␣\lambda␣_R','FontSize',24);
ylabel('h␣\lambda␣_I','FontSize',24);
saveas(hf,'andy_hw04_prb01.png')
% close(hf);
```

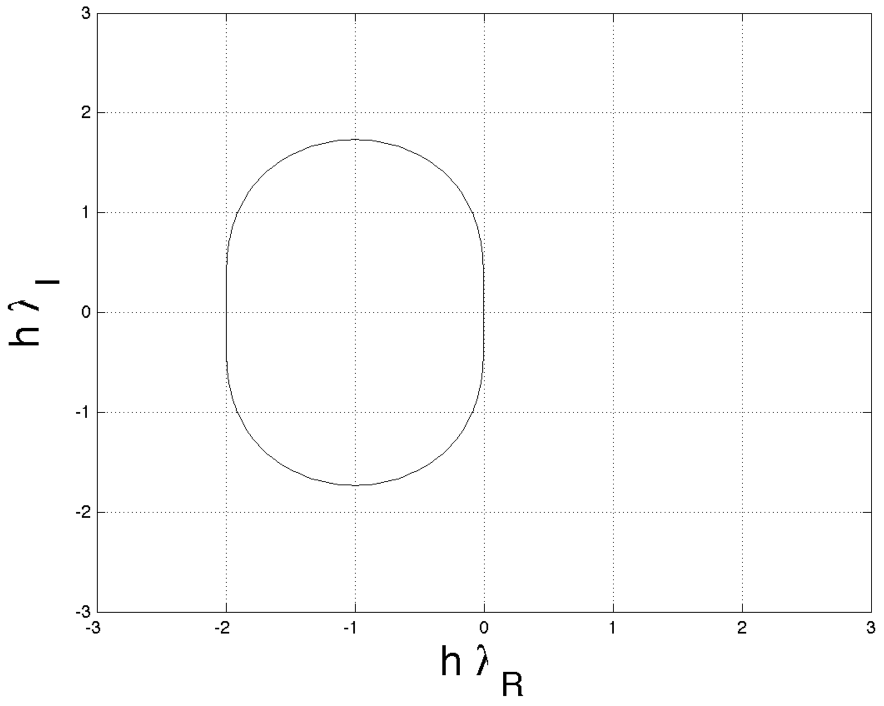

Figure 1: Stability of the ME method.

2. Use inequality 4.23 to obtain the bound 4.24 for the stability of the cRK method when $\lambda < 0$.

   Solution: From Equation 4.23, the cRK method is stable when

$$\left| \sum_{k=0}^{4} \frac{(h\lambda)^k}{k!} \right| \leq 1. \tag{8}$$

   Taking the simple route, we make a plot of this function, and note where it becomes greater than one:

```
% make a really quick plot of
% the stable region of the cRK method

% create figure instance
figure;
h = 1;
x = linspace(-4,1,100);
y = zeros(1,100);
% build the sum
% non vectorized but works
for k=0:4
    y = y+h^k.*x.^k./factorial(k);
end
% this won't matter, but just in case
y = abs(y);
% make the plot
plot(x,y);
hold on;
% plot a red line at 1
plot(x,ones(size(x)),'r');
xlabel('h␣\lambda␣_R','FontSize',24);
ylabel('error␣growth␣rate␣of␣cRK','FontSize',16);
saveas(hf,'andy_hw04_prb02_01.png');
% close(hf);

% find the stable region
stable_region = find(y < 1);
minstable =  x(min(stable_region)); % 2.78
% this should be zero
% for small number of x,y points, less than 0
maxstable =  x(max(stable_region));
% write out the latex
fprintf('the␣stable␣region␣is␣$%f␣\\leq␣h\\lambda␣\\leq␣%f$\n',minstable,maxstable);
```

   Increasing the number of points in the above plot, this code then tells us that the stable region is $-2.78 \leq h\lambda \leq 0$.

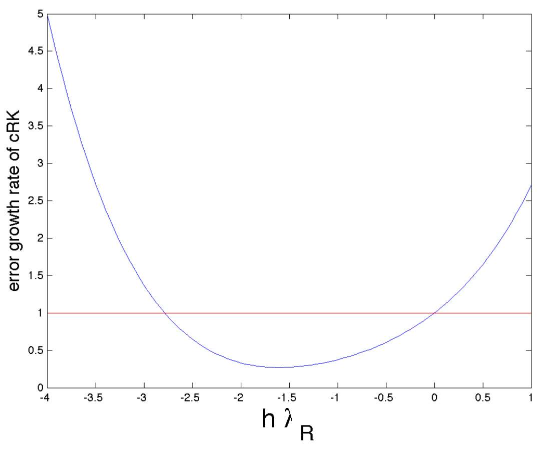

Figure 2: Stability of the cRK method.

3. Show analytically that the stability region of the Midpoint method coincides with that for the ME method.

   Solution: Similar to problem 1, we begin by writing the midpoint method in one step:

$$Y_{i+1} = Y_i + hf(x_i + h/2, Y_i + h/2 \cdot f(x_i, Y_i)). \tag{9}$$

   Plugging in the model problem, $f(x,y) = \lambda y$, we have the difference equation

$$\begin{aligned} Y_{i+1} &= Y_i + h\lambda Y_i + \frac{1}{2}h\lambda h f(x_i, Y_i)) & (10)\\ &= Y_i + h\lambda Y_i + \frac{1}{2}\lambda^2 h^2 Y_i & (11)\\ &= Y_i\left(1 + h\lambda + \frac{1}{2}(h\lambda)^2\right). & (12)\end{aligned}$$

   This equation is the same as Equation 5, but we could continue and we have the stability region as, again,

$$\left|1 + h\lambda + \frac{1}{2}(h\lambda)^2\right| = 1, \tag{13}$$

   as desired.

4. Use Equations 4.27 and 4.30 to plot the boundary of the stability region of the 2nd-order Adams-Bashforth method.

   Solution: From the notes, we have the boundary of the stability region given by both $|r_1| \le 1$ and $|r_2| \le 1$ for r_1, r_2:

$$r_1 = \frac{1}{2}\left\{\left(1+\frac{3}{2}\lambda h\right) + \sqrt{\left(1+\frac{3}{2}\lambda h\right)^2 - 2\lambda h}\right\} \tag{14}$$

$$r_2 = \frac{1}{2}\left\{\left(1+\frac{3}{2}\lambda h\right) - \sqrt{\left(1+\frac{3}{2}\lambda h\right)^2 - 2\lambda h}\right\}. \tag{15}$$

```
% make a contour plot of the stability region of A-B 2nd order

% number of points in x and y
numz = 1000;
numr = 3; % bound of x,y
x = linspace(-numr,numr,numz); % real
y = linspace(-numr,numr,numz); % imag
[xz,yz] = meshgrid(x,y); % make matrices
h = 1; % h just scales the axes
z = xz+1i*yz; % use this for complex math
stable_region1 = abs(0.5.*((1+1.5.*z.*h)+sqrt((1+1.5.*z.*h).^2-2.*z.*h)));
stable_region2 = abs(0.5.*((1+1.5.*z.*h)-sqrt((1+1.5.*z.*h).^2-2.*z.*h)));

% non-vectorized search for the boundary
% could use 'find' in a smart way to vectorize
% but this works
stable_region = zeros(numz);
for i=1:numz
    for j=1:numz
        % check if on the boundary of both
        if stable_region1(i,j) <= 1 && stable_region2(i,j) <= 1
            stable_region(i,j) = 1;
        end
    end
end

% plot the first figure
% looks like that in notes
hf = figure;
contour(xz,yz,stable_region1,[1,1],'k');
hold on;
contour(xz,yz,stable_region2,[1,1],'k');
% set(gca,'XTick',1:(numz/12):numz)
% set(gca,'XTickLabel',x(1:(numz/12):end))
grid on;
xlabel('h␣\lambda␣_R','FontSize',24);
ylabel('h␣\lambda␣_I','FontSize',24);
saveas(hf,'andy_hw04_prb04_01.png')
% close(hf);

% plot the second figure
% cleaner boundary from search
gf = figure;
contour(xz,yz,stable_region,[1,1],'k');
grid on;
xlabel('h␣\lambda␣_R','FontSize',24);
ylabel('h␣\lambda␣_I','FontSize',24);
```

```
saveas(gf,'andy_hw04_prb04_02.png')
% close(gf);
```

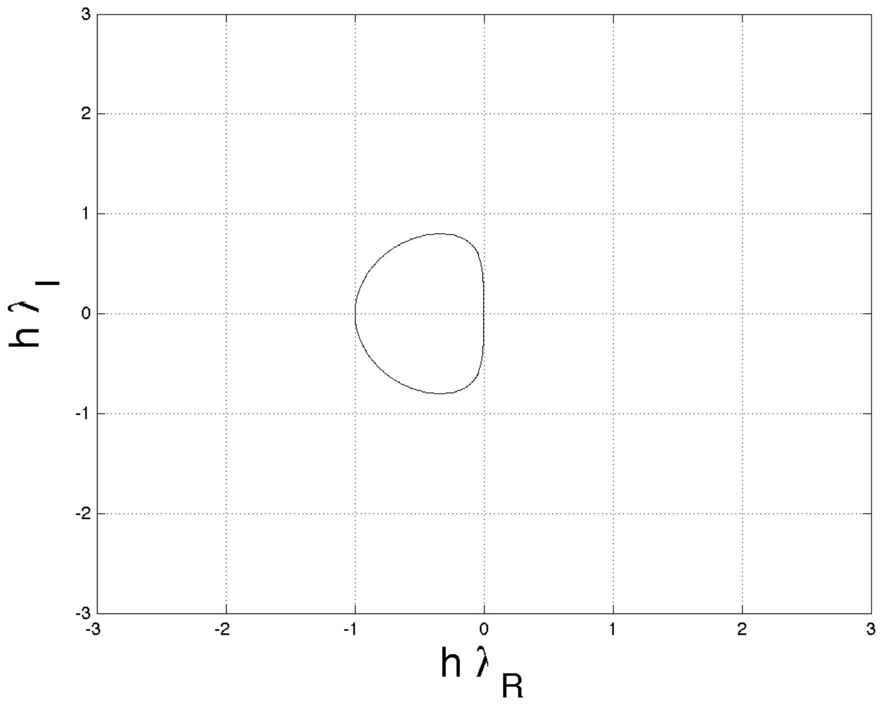

Figure 3: Stability of the 2nd-order Adams Bashforth method.

5. Obtain the analog of Equations 4.26 and 4.27 for the P-C method (3.33) of Lecture 3. Plot its stability region following the suggestions of Problem 4. Is this stability region larger or smaller than that of the 2nd-order Adams-Bashforth method? In general, try to make an educated guess about how the stability region of a P-C method is related to the stability regions of its predictor and corrector equations.

   Solution: The P-C method is given by equation 3.33 from Lecture 3, which is:

$$\text{Predictor} \;:\; Y_{i+1}^{p} = Y_i + h/2\cdot(3f_i - f_{i-1}) \tag{16}$$

$$\text{Corrector} \;:\; Y_{i+1}^{c} = Y_i + h/2\cdot(f_i + f_{i+1}^{p}). \tag{17}$$

   Writing this as one equation, we have

$$\begin{aligned} Y_{i+1} &= Y_i + h/2\cdot(\lambda Y_i + \lambda(Y_i + h/2\cdot(3\lambda Y_i - \lambda Y_{i-1}))) && (18)\\ &= Y_i + \frac{1}{2}\lambda h Y_i + \frac{1}{2}\lambda h Y_i + \frac{3}{4}\lambda^2 h^2 Y_i - \frac{1}{4}\lambda^2 h^2 Y_{i-1} && (19)\\ &= Y_i + \lambda h Y_i + \frac{3}{4}\lambda^2 h^2 Y_i - \frac{1}{4}\lambda^2 h^2 Y_{i-1} && (20)\end{aligned}$$

   I switch now from subscripts Y_i to Y_n for less confusion with the imaginary unit. Writing this as a difference equation we have

$$Y_{n+1} - Y_n - \lambda h Y_n - \frac{3}{4}\lambda^2 h^2 Y_n + \frac{1}{4}\lambda^2 h^2 Y_{n-1} = 0. \tag{21}$$

Substituting $Y_n = r^n$ in the above, we have

$$r^{n+1} - r^n - \lambda h r^n - \frac{3}{4}\lambda^2 h^2 r^n + \frac{1}{4}\lambda^2 h^2 r^{n-1} = 0. \tag{22}$$

Cancelling the r^{n-1} this becomes

$$r^2 + r\left(-1 - \lambda h - \frac{3}{4}\lambda^2 h^2\right) + \frac{1}{4}\lambda^2 h^2 = 0. \tag{23}$$

Solving this for the roots of r we have the roots

$$r_2 = \frac{1}{2}\left\{1 + \lambda h + \frac{3}{4}\lambda^2 h^2 + \sqrt{\left(1 + \lambda h + \frac{3}{4}\lambda^2 h^2\right)^2 - \lambda^2 h^2}\right\} \tag{24}$$

$$r_2 = \frac{1}{2}\left\{1 + \lambda h + \frac{3}{4}\lambda^2 h^2 - \sqrt{\left(1 + \lambda h + \frac{3}{4}\lambda^2 h^2\right)^2 - \lambda^2 h^2}\right\} \tag{25}$$

The following code plots the boundary of the stable region.

The stability region is larger than that of the Adams-Bashforth method alone, indicating that the P-C scheme has improved the stability.

In general, I expect a P-C method to have a larger stability region than the method with the smallest region of stability. This example also shows that areas of stability not found in either method can be stable in the P-C, and the stable region can be smaller than the largest stable region of the P-C methods.

```
% make a contour plot of the stability region of A-B 2nd order

% number of points in x and y
numz = 1000;
numr = 3; % bound of x,y
x = linspace(-numr,numr,numz); % real
y = linspace(-numr,numr,numz); % imag
[xz,yz] = meshgrid(x,y); % make matrices
h = 1; % h just scales the axes
z = xz+1i*yz; % use this for complex math
stable_region1 = abs(0.5.*((1+z.*h+3/4*h^2.*(z.^2))+sqrt((1+z.*h+3/4*h^2.*(z.^2)).^2-z
    .^2.*h^2)));
stable_region2 = abs(0.5.*((1+z.*h+3/4*h^2.*(z.^2))-sqrt((1+z.*h+3/4*h^2.*(z.^2)).^2-z
    .^2.*h^2)));

% non-vectorized search for the boundary
% could use 'find' in a smart way to vectorize
% but this works
stable_region = zeros(numz);
for i=1:numz
    for j=1:numz
        % check if on the boundary of both
        if stable_region1(i,j) <= 1 && stable_region2(i,j) <= 1
            stable_region(i,j) = 1;
        end
    end
end

% plot the first figure
% looks like that in notes
hf = figure;
contour(xz,yz,stable_region1,[1,1],'k');
hold on;
contour(xz,yz,stable_region2,[1,1],'k');
% set(gca,'XTick',1:(numz/12):numz)
% set(gca,'XTickLabel',x(1:(numz/12):end))
grid on;
xlabel('h␣\lambda␣_R','FontSize',24);
ylabel('h␣\lambda␣_I','FontSize',24);
saveas(hf,'andy_hw04_prb05_01.png')
% close(hf);

% plot the second figure
% cleaner boundary from search
gf = figure;
contour(xz,yz,stable_region,[1,1],'k');
grid on;
xlabel('h␣\lambda␣_R','FontSize',24);
ylabel('h␣\lambda␣_I','FontSize',24);
saveas(gf,'andy_hw04_prb05_02.png')
% close(gf);
```

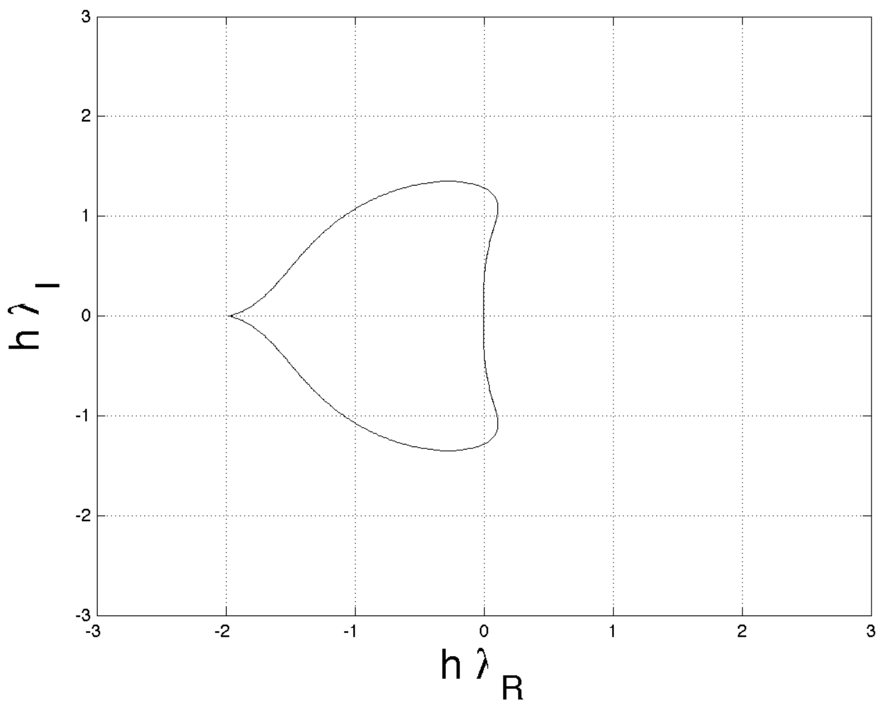

Figure 4: Stability of the P-C method given in Equation 3.33 from Lecture 3.

6. Find *analytically* the stability region of the Modified Implicit Euler method (3.43) and make a sketch of the this region. Explain why your result implies that this method is A-stable.

   Solution: The modified Euler method is given by

$$Y_{i+1} = Y_i + \frac{h}{2}\left(f(x_i, Y_i) + f(x_{i+1}, Y_{i+1})\right). \tag{26}$$

   Expanding this equation for the the model equation (4.15) we have

$$\begin{aligned} Y_{i+1} &= Y_i + \frac{h}{2}\left(\lambda Y_i + \lambda Y_{i+1}\right) && (27)\\ &= Y_i(1 + h\lambda/2) + h\lambda/2 \cdot Y_{i+1}. && (28) \end{aligned}$$

   Rearranging this we have

$$Y_{i+1}(1 - h\lambda/2) = Y_i(1 + h\lambda/2). \tag{29}$$

   Recognizing this as a recursion, we write

$$Y_n = Y_0\left(\frac{1 + h\lambda/2}{1 - h\lambda/2}\right)^n. \tag{30}$$

   The modified Implicit Euler method is therefore stable for

$$\left|\frac{1 + h\lambda/2}{1 - h\lambda/2}\right| \le 1 \qquad \Rightarrow \qquad |1 + h\lambda/2| \le |1 - h\lambda/2|. \tag{31}$$

   This inequality clearly holds for all $\lambda < 0$, making the method A-stable. In more detail, let $\lambda = \lambda_R + i\lambda_I$ where λ_R and λ_I are the real and imaginary parts of the complex λ, respectively. Substituting this in, we have

$$\begin{aligned} |1 + h\lambda/2| &\le |1 - h\lambda/2| && (32)\\ \sqrt{(1 + h\lambda_R/2)^2 + (h\lambda_I/2)^2} &\le \sqrt{(1 - h\lambda_R/2)^2 + (-h\lambda_I/2)^2} && (33)\\ (1 + h\lambda_R/2)^2 + (h\lambda_I/2)^2 &\le (1 - h\lambda_R/2)^2 + (h\lambda_I/2)^2 && (34)\\ (1 + h\lambda_R/2)^2 &\le (1 - h\lambda_R/2)^2 && (35)\\ 1 + h\lambda_R/2 &\le 1 - h\lambda_R/2 && (36)\\ h\lambda_R/2 &\le -h\lambda_R/2 && (37)\\ h\lambda_R &\le 0 && (38) \end{aligned}$$

   I make a sketch of this region:

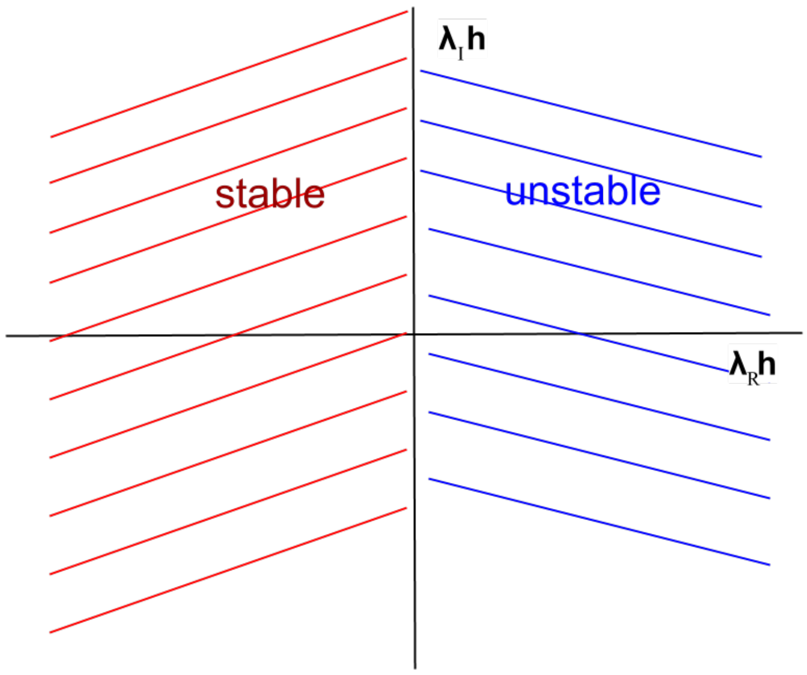

Figure 5: Sketch of the stable region of the modified Euler method.

7. Solve the IVP

$$y' = 20y, \qquad y(0) = 1$$

with $h = 0.125$ up to $x = 1.5$ using (a) simple Euler, (b) cRK, (c) implicit Euler methods. Plot your result from (a). In a separate figure, plot your results from (b) and (c) along with the exact solution. Which method, the 4-th order (b), or the 1st-order (c), gives the more accurate solution in this case?

Without doing additional calculations, what do you expect to change in those results if you use $h = 0.15$ instead? Write a bried but coherent paragraph explaining your answer.

Solution: I solve the ODE using the given methods in the following MATLAB script:

```
% compare the simple Euler,
% cRK, and implicit Euler methods

% include my_ME.m and my_cRK.m from hw02
try
    addpath('../shared_methods');
catch
    fprintf('this␣code␣will␣not␣work␣');
    fprintf('without␣my_cRK␣and␣my_ME␣in␣path\n');
end

% define h, methods to use
h = 0.125;
methods = {'my_SE','my_cRK','my_IE'};
x = 0:h:1.5;
```

```
% store the solutions as rows in a matrix
sol_mat = zeros(length(methods)+1,length(x));
% our ODE
dy = @(t,y,params) -20*y;
% our IC
y0 = 1;

% analytical solution
for i=1:length(x)
    t=x(i); sol_mat(1,i) = exp(-20*t);
end

% solve for each method
for i=1:length(methods)
    fprintf('solving␣with␣%s\n',methods{i});
    method = str2func(methods{i});
    sol_mat(i+1,:) = method(dy,x,y0,h,[]);
end

% make plots
figure;
plot(x,sol_mat(2,:));
xlabel('x','FontSize',24);
ylabel('y','FontSize',24);
saveas(gcf,'andy_hw04_prb07_01.png');
% close(fh);

figure;
plot(x,sol_mat(1,:),x,sol_mat(3,:),x,sol_mat(4,:));
legend('analytical','cRK','implicit␣Euler');
xlabel('x','FontSize',24);
ylabel('y','FontSize',24);
saveas(gcf,'andy_hw04_prb07_02.png');
% close(fh);
```

The plots are produced from this script directly. They are as follows.

We observe that the 1st-order implicit Euler method is more accurate in this case. An increase in h will move λh further away from the stable region for the cRK method, yet remain within the stable region of the implicit Euler method. For this reason, I would expect that the implicit Euler solution would improve while the cRK solution would become less stable, and likely degrade in accuracy. Specifically, with $\lambda = -20$, and $h = 0.125$ we have $\lambda h = 2.5$. Given the stability bounds for cRK found in problem 4, an increase of h to 0.15 will make $\lambda h = 3$ and therefore cRK will be unstable. Since implicit Euler is A-stable, it will remain stable for $h = 0.15$.

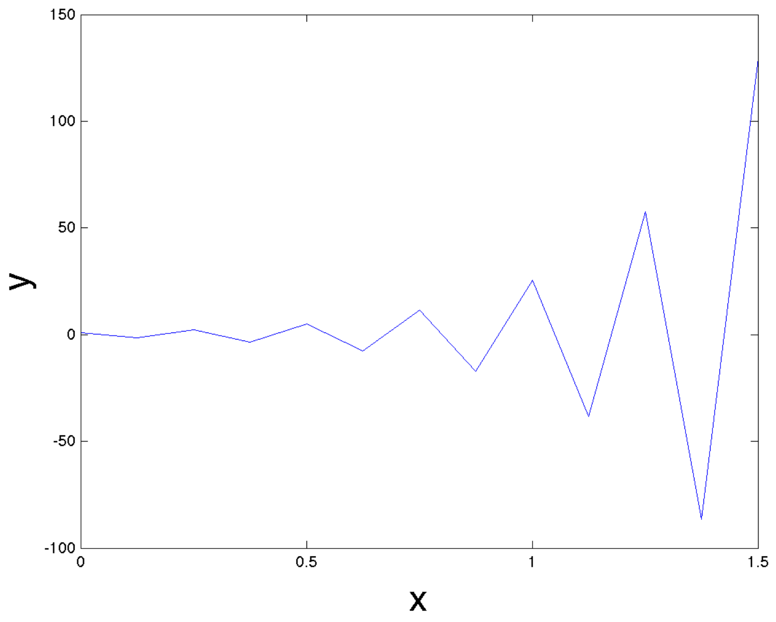

Figure 6: Solution with the simple Euler method.

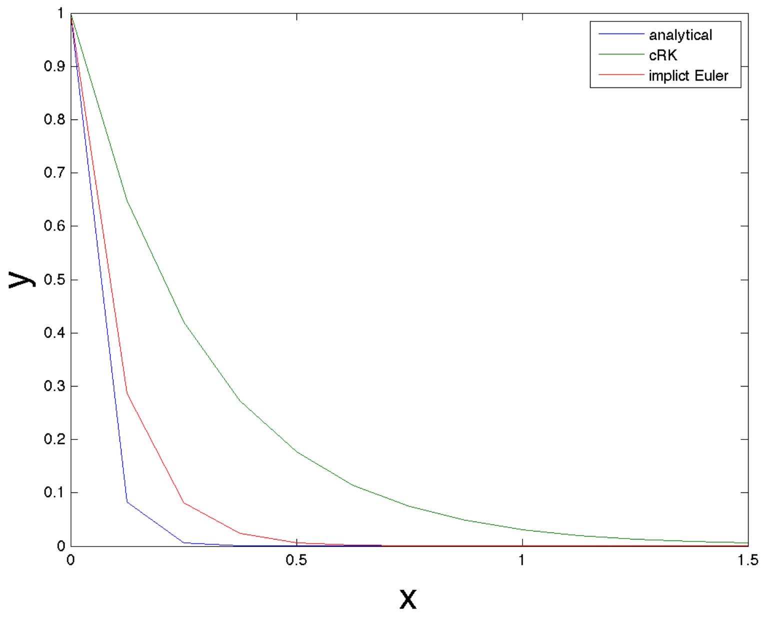

Figure 7: The exact, cRK, and implicit Euler solutions.

Bonus As shown in the notes, the Leap-frog method, introduced in Lecture 3, is unstable for $\lambda < 0$. Now, construct a P-C method where the Leap-frog method is used as the predictor and the trapezoial rule is used as the corrector (as in method (3.33)). Repeat Problem 5 for this new P-C method.

How does this region compare with the stability regions of the predictor equation and of the corrector equation alone? Does this graph agree with what you observed in problem 5?

Solution: We write this P-C method as:

$$\text{Predictor} \;:\; Y^p_{i+1} = Y_{i-1} + 2hf(x_i, Y_i) \tag{39}$$

$$\text{Corrector} \;:\; Y^c_{i+1} = Y_i + h/2 \cdot (f(x_i, Y_i) + f(x_{i+1}, Y^p_{i+1}). \tag{40}$$

This problem now closely follows the Problem 5. We write out this method in one step as:

$$Y_{i+1} = Y_i + h/2 \cdot (f(x_i, Y_i) + f(x_{i+1}, Y_{i-1} + 2hf(x_i, Y_i))). \tag{41}$$

Substituting in the model equation we have:

$$\begin{aligned} Y_{i+1} &= Y_i + h/2 \cdot (\lambda Y_i + \lambda Y_{i-1} + \lambda 2h\lambda Y_i) && (42)\\ &= Y_i + h/2\lambda Y_i + h/2\lambda Y_{i-1} + h/2\lambda 2h\lambda Y_i && (43)\\ &= Y_i(1 + h\lambda/2 + (h\lambda)^2) + Y_{i-1}(h\lambda/2) && (44)\end{aligned}$$

Substituting $Y_n = r^n$ in the above, we have

$$r^{n+1} - r^n(1 + h\lambda/2 + (h\lambda)^2) - r^{n-1}(h\lambda/2) = 0. \tag{45}$$

Cancelling the r^{n-1} this becomes

$$r^2 - r(1 + (h\lambda)/2 + (h\lambda)^2) - (h\lambda)/2 = 0. \tag{46}$$

Let z denote $h\lambda$. Solving the previous equation for the roots of r we have the roots

$$r_1 = \frac{1}{2}\left\{1 + (h\lambda)/2 + (h\lambda)^2 + \sqrt{(1 + (h\lambda)/2 + (h\lambda)^2)^2 + 2\lambda h}\right\} \tag{47}$$

$$r_2 = \frac{1}{2}\left\{1 + (h\lambda)/2 + (h\lambda)^2 - \sqrt{(1 + (h\lambda)/2 + (h\lambda)^2)^2 + 2\lambda h}\right\} \tag{48}$$

The following code plots the boundary of the stable region.

```
% make a contour plot of the stability region of A-B 2nd order

% number of points in x and y
numz = 1000;
numr = 3; % bound of x,y
x = linspace(-numr,numr,numz); % real
y = linspace(-numr,numr,numz); % imag
[xz,yz] = meshgrid(x,y); % make matrices
h = 1; % h just scales the axes
z = xz+1i*yz; % use this for complex math
stable_region1 = abs(0.5.*((1+h.*z./2+(h.*z).^2)+sqrt((1+h.*z./2+(h.*z).^2).^2+2.*h.*z
    )));
```

```
stable_region2 = abs(0.5.*((1+h.*z./2+(h.*z).^2)-sqrt((1+h.*z./2+(h.*z).^2).^2+2.*h.*z
    )));

% non-vectorized search for the boundary
% could use 'find' in a smart way to vectorize
% but this works
stable_region = zeros(numz);
for i=1:numz
    for j=1:numz
        % check if on the boundary of both
        if stable_region1(i,j) <= 1 && stable_region2(i,j) <= 1
            stable_region(i,j) = 1;
        end
    end
end

% plot the first figure
% looks like that in notes
hf = figure;
contour(xz,yz,stable_region1,[1,1],'k');
hold on;
contour(xz,yz,stable_region2,[1,1],'k');
% set(gca,'XTick',1:(numz/12):numz)
% set(gca,'XTickLabel',x(1:(numz/12):end))
grid on;
xlabel('h␣\lambda␣_R','FontSize',24);
ylabel('h␣\lambda␣_I','FontSize',24);
saveas(hf,'andy_hw04_prb08_01.png')
% close(hf);

% plot the second figure
% cleaner boundary from search
gf = figure;
contour(xz,yz,stable_region,[1,1],'k');
grid on;
xlabel('h␣\lambda␣_R','FontSize',24);
ylabel('h␣\lambda␣_I','FontSize',24);
saveas(gf,'andy_hw04_prb08_02.png')
% close(gf);
```

And we have the following stability region. Indeed, this appears to be the product of the two stability regions for the individual predictor and corrector methods.

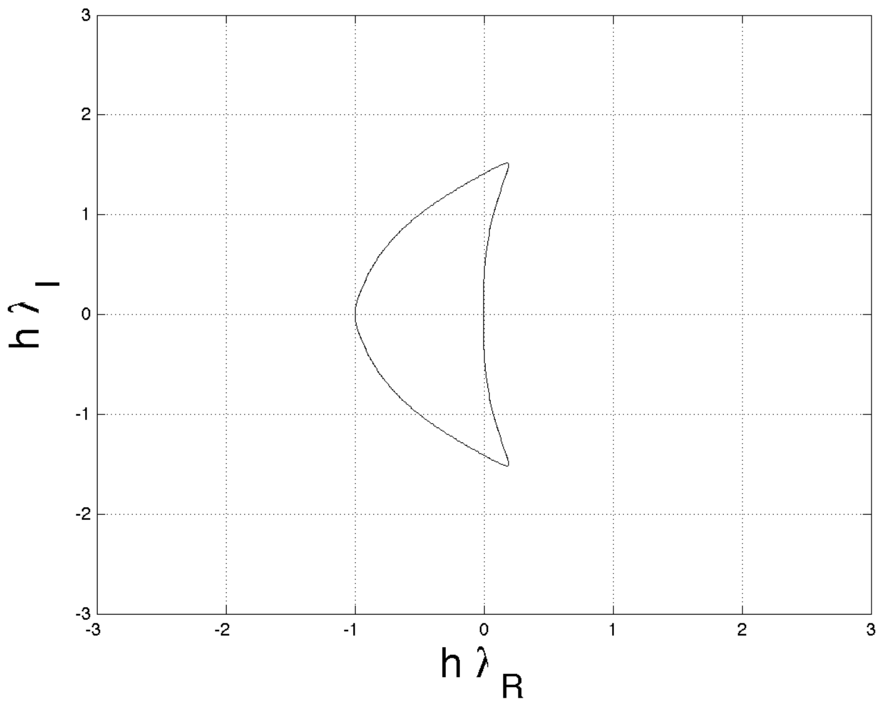

Figure 8: The stability region of a P-C method with Leap-Frog predictor and trapezoidal corrector.

Appendix: Function code

```
function yvec = my_ME(func,tspan,y0,h,params)

yvec = [];
yvec = [yvec y0];
t = tspan(1);
for i=2:length(tspan)
    k1 = func(t,yvec(i-1),params);
    k2 = func(t+h,yvec(i-1)+h*k1,params);
    yvec = [yvec yvec(i-1)+h/2*(k1+k2)];
    t=t+h;
end
```

```
function yvec = my_IE(func,tspan,y0,h,params)
% implicit Euler
%
% note that this function only works
% for problems of the form y' = ay

a = func(0,1,params);
t = tspan(1);

yvec =  [];
yvec = [yvec y0];
for i=2:length(tspan)
    yvec = [yvec yvec(i-1)/(1-a*h)];
    t = t+h;
end
```

```
function yvec = my_cRK(func,tspan,y0,h,params)

% set the coefficients
[a11,a21,a22,a31,a32,a33] = deal(0.5,0.0,0.5,0.0,0.0,1.0);
[b1,b2,b3,b4] = deal(1/6,1/3,1/3,1/6);
[c1,c2,c3] = deal(0.5,0.5,1);
t = tspan(1);

yvec =  []; %linspace(tspan[0],tspan[-1],num=floor((tspan[-1]-tspan[0])/h))
yvec = [yvec y0]; %[0] = y0
for i=2:length(tspan)
    k1 = h*func(t,yvec(i-1),params);
    k2 = h*func(t+c1*h,yvec(i-1)+a11*k1,params);
    k3 = h*func(t+c2*h,yvec(i-1)+a21*k1+a22*k2,params);
    k4 = h*func(t+c3*h,yvec(i-1)+a31*k1+a32*k2+a33*k3,params);
    yvec = [yvec yvec(i-1)+b1*k1+b2*k2+b3*k3+b4*k4]; %[i] = yvec[i-1] + b1*k1 + b2*k2 +
        b3*k3 + b4*k4
    t = t+h;
end
```

```
function yvec = my_SE(func,tspan,y0,h,params)
% simple Euler
%
% not much else to say

t = tspan(1);

yvec =  [];
yvec = [yvec y0];
for i=2:length(tspan)
    yvec = [yvec yvec(i-1)+h*func(t,yvec(i-1),params)];
    t = t+h;
end
```