# CX 4640 Assignment 10

Wesley Ford

November 9th, 2020

1. Consider the general scalar ODE: $y' = f(t, y)$ with $y(0) = y_0$. Suppose that you use fixed-point iteration to solve the nonlinear equation at each step of backward Euler. What is the condition that is needed for convergence of the fixed-point iteration? Assume a constant step size $h$.
   The backwards Euler formula is
   $$y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$$
   For fixed-point iteration to converge we need a function such that $y_{k+1}^{[i+1]} = g\left(y_{k+1}^{[i]}\right)$. Backwards Euler gives us this expression
   $$g\left(y_{k+1}^{[i]}\right) = y_k + hf\left(t_{k+1}, y_{k+1}^{[i]}\right) = y_{k+1}^{[i+1]}$$
   with $y_{k+1}^{[0]} = y_0$. For the fixed point iteration to converge, the magnitude of the derivative of $g$ must be less than 1. Takeing the derivative of $g\left(y_{k+1}\right)$ with respect to $y_{k+1}$
   $$\left|g'\left(y_{k+1}^{[i]}\right)\right| < 1$$
   $$\left|hf'(t_{k+1}, y_{k+1}^{[i]})\right| < 1$$
   $$\left|hf'\left(t_{k+1}, y_{k+1}^{[i]}\right)\right| < 1$$
   $$-1 < hf'(t_{k+1}, y_{k+1}^{[i]}) < 1$$

2. Consider the implicit midpoint method for solving initial-value ODEs.

(a) Write the Runge-Kutta table (Butcher tableau) for this method.

The implicit midpoint method is

$$y_{k+1} = y_k + hf(t_{k+1/2}, y_{k+1/2})$$
$$\text{where } t_{k+1/2} = \frac{t_k + t_{k+1}}{2},\ y_{k+1/2} = \frac{y_k + y_{k+1}}{2}$$

For a general RK method with the form

$$y_{k+1} = y_k + h\sum_{j=1}^{s} b_j f(t_k + c_i h, Y_j)$$
$$Y_j = y_k + h\sum_{i=1}^{s} a_{jk} f(t_k + c_i h, Y_i)$$

For the implicit midpoint method, there is only one step, so $s = 1$ and the coefficient of that step, $b_1 = 1$. $t_{k+1/2} = \frac{t_k + t_{k+1}}{2} = t_k + c_i h$. If $h$ is the size of the timestep, then $c_1 = 1/2$. $Y_1$ then becomes

$$Y_1 = y_k + ha_{11} f(t_{k+1/2}, Y_1) = y_{k+1/2} = \frac{y_k + y_{k+1}}{2}$$
$$y_{k+1} = y_k + hf(t_{k+1/2}, Y_1)$$
$$f(t_{k+1/2}, Y_1) = \frac{y_{k+1} - y_k}{h}$$
$$Y_1 = y_k + a_{11}(y_{k+1} - y_k) = \frac{y_k + y_{k+1}}{2}$$
$$a_{11} = \frac{1}{2}$$

The implict midpoint method has the following values in the standard $s$-stage RK form: $s = 1$, $b_1 = 1$, $c_1 = \frac{1}{2}$ and $a_{11} = \frac{1}{2}$. This leads to the following Runge-Kutta table

| | |
|---|---|
| $\frac{1}{2}$ | $\frac{1}{2}$ |
| | 1 |

(b) What is the region of absolute stability for this method?

Using a test equation $y' = \lambda y$, $Re(\lambda) \leq 0$, the implicit midpoint method becomes

$$y_{k+1} = y_k + h\lambda\left(\frac{y_k + y_{k+1}}{2}\right)$$

This can be rearranged into

$$y_{k+1} = \frac{2 + h\lambda}{2 - h\lambda} y_k$$

To ensure stability, $y_{k+1}$ can not become larger in magnitude that $y_k$ (because of the condition $Re(\lambda) < 0$, $y$ is a decreasing exponential function). Thus

$$\left|\frac{2 + h\lambda}{2 - h\lambda}\right| \leq 1$$
$$|2 + h\lambda| \leq |2 - h\lambda|$$

As long as the $Re(\lambda) \leq 0$, $|2 + h\lambda| \leq |2 - h\lambda|$ will hold, so the region of absolute stability is the entire left portion ($Re(z) \leq 0$) of the complex plane.

3. Chapter 16, question 8, page 532: To draw a circle of radius $r$ on a graphics screen, one may proceed to evaluate paris of values $x = r\cos(\theta), y = r\sin(\theta)$ for a succession of values $\theta$. But this is computationally expensive. A cheaper method may be obtained by considering the ODE

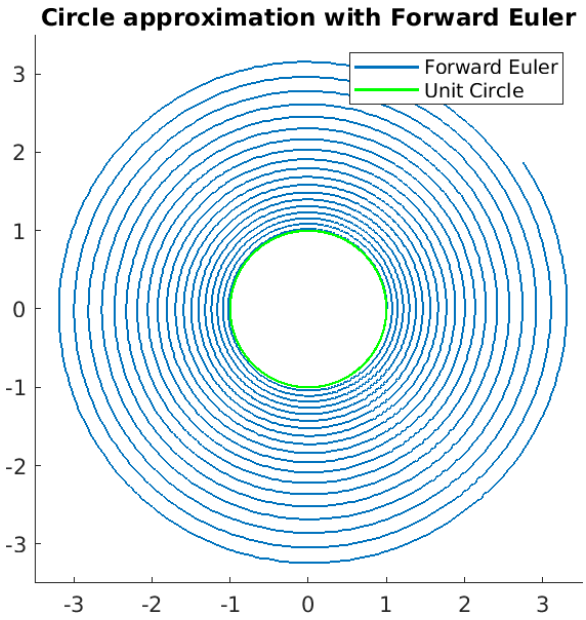
$$\dot{x} = -y, \quad x(0) = r,$$
$$\dot{y} = x, \quad y(0) = 0,$$

where $\dot{x} = \frac{dx}{d\theta}$, and approximating this using a simple discretization method. However, care must be taken to ensure that the obtained approximate solution looks right, i.e., that the approximate curve closes rather than spirals. Carry out this integration using a uniform step size $h = 0.2$ for $0 \leq \theta \leq 120$, applying forward Euler, backward Euler, and the implict trapezoidal method. Determine if the solution spirals in, spirals out, or forms an approximate circle as desired. Explain the observed results.

The Forward Euler approximation uses the following equations

$$x_{k+1} = x_k - hy_k$$
$$y_{k+1} = y_k + hx_k$$

and produces the following graph




The circle in green is the circle that we are trying to approximate. Clearly, the Forward Euler spirals outwards, and does a poor job of approximating the circle. Intuitively this makes sense, as the Forward Euler method uses the derivative at the current point $[x_k, y_k]$ to approximate the next point $[x_{k+1}, y_{k+1}]$, so the circle that is being approximated at each step will always be slightly larger than the current circle being approximated, causing the Forward Euler approximation to spiral outwards. Mathematically, the distance from the origin to the point $[x, y]$ is given by $r^2 = x^2 + y^2$. Ideally for a circle, $r$ is constant. Using the Forward Euler equations, this is not that case

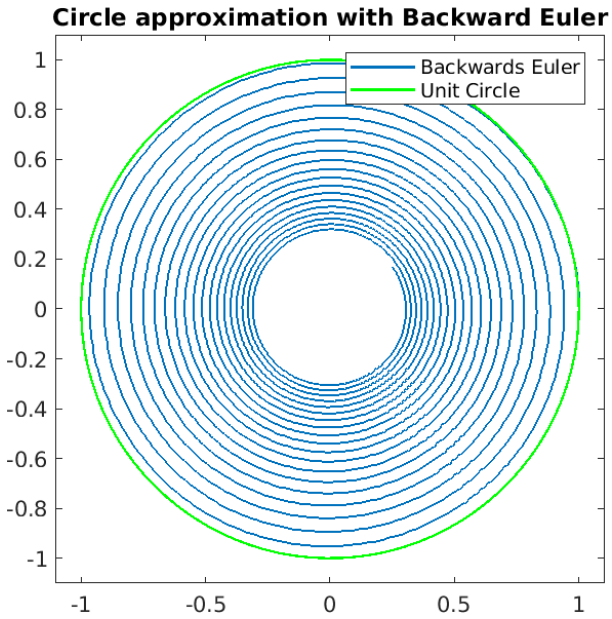
$$r_{k+1}^2 = x_{k+1}^2 + y_{k+1}^2 = (x_k^2 - 2hx_ky_k + h^2y_k^2) + (y_k^2 + 2hx_ky_k + h^2x_k^2)$$
$$r_{k+1}^2 = (1 + h^2)(x_k^2 + y_k^2)$$
$$r_{k+1}^2 = (1 + h^2)r_k^2$$

For each iteration, the approximated points drift slightly further and further away from the origin, leading to the spiral outwards graph.

The Backward Euler approximation using the following equations The Forward Euler approximation uses the following equations

$$x_{k+1} = x_k - hy_{k+1}$$
$$y_{k+1} = y_k + hx_{k+1}$$

and produces the following graph




Clearly, the Backwards Euler spirals inwards from the circle it is attempting to approximate. This is because the Backwards Euler uses the derivative at $[x_{k+1}, y_{k+1}]$ to approximate the point $[x_{k+1}, y_{k+1}]$. This causes the approximation attempt to approximate smaller and smaller circles each time, leading to the spiral inwards pattern. Mathematically:

$$x_{k+1} = x_k - hy_{k+1} = \frac{x_k - hy_k}{1+h^2}$$
$$y_{k+1} = y_k + hx_{k+1} = \frac{y_k + hx_k}{1+h^2}$$
$$r_{k+1}^2 = \left(\frac{x_k - hy_k}{1+h^2}\right)^2 + \left(\frac{y_k + hx_k}{1+h^2}\right)^2$$
$$r_{k+1}^2 = (1+h^2)^{-2}\left(x_k^2 - 2hx_ky_k + h^2y_k^2 + y_k^2 + 2hx_ky_k + h^2x_k^2\right)$$
$$r_{k+1}^2 = \frac{r_k^2}{(1+h^2)}$$

We can see that the radius at each step actually decreases by a factor of $\sqrt{1+h^2}$, leading to the spiral inward pattern.

The implicit trapezoid was implemented using the following equations

$$x_{k+1} = x_k - \frac{h}{2}(y_k + y_{k+1})$$
$$y_{k+1} = y_k + \frac{h}{2}(x_k + x_{k+1})$$

. Solving for $x_{k+1}$ and $y_{k+1}$

$$x_{k+1} = x_k - \frac{h}{2}\left(y_k + y_k + \frac{h}{2}(x_k + x_{k+1})\right) = \frac{x_k\left(1 - \frac{h^2}{4}\right) - hy_k}{1 + \frac{h^2}{4}}$$

$$y_{k+1} = y_k + \frac{h}{2}\left(x_k + x_k - \frac{h}{2}(y_k + y_{k+1})\right) = \frac{y_k\left(1 - \frac{h^2}{4}\right) + hx_k}{1 + \frac{h^2}{4}}$$

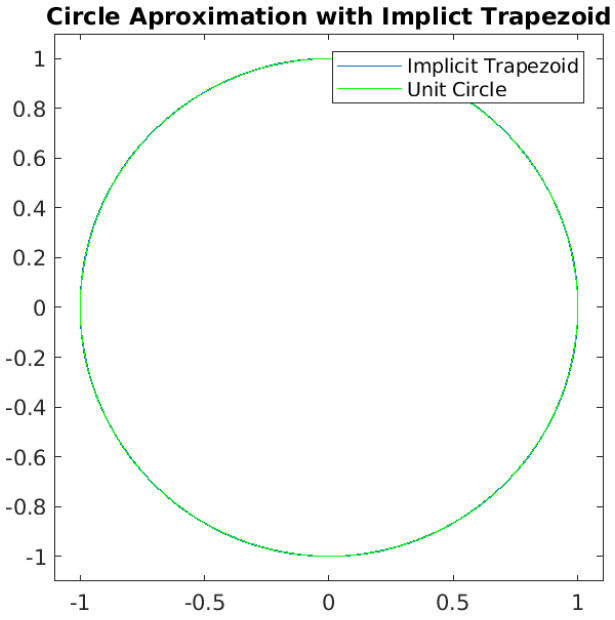
$$r_{k+1}^2 = \left(1 + \frac{h^2}{4}\right)^{-2}\left[\left(x_k\left(1 + \frac{h^2}{4}\right) - hy_k\right)^2 + \left(y_k\left(1 - \frac{h^2}{4}\right) + hx_k\right)^2\right]$$

$$r_{k+1}^2 = \left(1 + \frac{h^2}{4}\right)^{-2}\left[x_k^2\left(\left(1 - \frac{h^2}{4}\right) + h^2\right) + y_k^2\left(\left(1 - \frac{h^2}{4}\right) + h^2\right)\right]$$

$$r_{k+1}^2 = \left(1 + \frac{h^2}{4}\right)^{-2}\left[x_k^2\left(1 + \frac{h^2}{4}\right)^2 + y_k^2\left(1 + \frac{h^2}{4}\right)^2\right]$$

$$r_{k+1}^2 = r_k^2$$

This equality means that for each iteration of the implicit trapezoid function the subsequent radius does not increase or decrease at all! This is evident in the following plot:




```
%Code for question 3, Assignment 10.

clf;
%initalize parameters
h = 0.02;
min = 0;
max = 120;
r= 1;
steps = (120./ h)+1;
```

```
%forward Euler
xfe = zeros(1,steps);
xfe(1) = r;
yfe = zeros(1,steps);
yfe(1) = 0;
for i=1:steps
xfe(i+1) = xfe(i) - h*yfe(i);
yfe(i+1) = yfe(i) + h*xfe(i);
end
hold on
plot(xfe, yfe);
plot(r.*cos(linspace(0,2*pi)), r.*sin(linspace(0,2*pi)), '-g');
xlim([-3.5,3.5]);
ylim([-3.5, 3.5]);
pbaspect([1,1,1]);
hold off
%backward Euler
xbe = zeros(1,steps);
xbe(1) = r;
ybe = zeros(1,steps);
ybe(1) = 0;
for i=1:steps
%P
xbe(i+1) = xbe(i) - h*ybe(i);
ybe(i+1) = ybe(i) + h*xbe(i);
%E
for j = 1:2
fx = -ybe(i+1);
fy = xbe(i+1);
%C
xbe(i+1) = xbe(i) + h*fx;
ybe(i+1) = ybe(i) + h*fy;
end
end

plot(xbe, ybe);
hold on;
plot(r.*cos(linspace(0,2*pi)), r.*sin(linspace(0,2*pi)), '-g');
pbaspect([1,1,1]);
hold off

%Implicit trapezoid
xit = zeros(1,steps);
yit = zeros(1,steps);
xit(1) = r;
yit(1) = 0;

for i= 1:steps
%Predict i+1 val using i val
xit(i+1) = xit(i) + (h./2) .* (-yit(1) - yit(1));
yit(i+1) = yit(i) + (h./2) .* (xit(i) + xit(i));
for j=1:2
%Evaluate fx and fy at thes predicted values i+1
```

```
fx = -yit(i+1);
fy = xit(i+1);
%Correct i+1 vals with new fx/fy
xit(i+1) = xit(i) + (h./2).* (-yit(i) + fx);
yit(i+1) = yit(i) + (h./2).* (xit(i) + fy);

end
end
plot(xit, yit);
hold on
%plot(r.*cos(linspace(0,2.*pi)), r.*sin(linspace(0,2.*pi)));
%xlim([-1.1, 1.1]);
%ylim([-1.1, 1.1]);
pbaspect([1,1,1]);
```