MATH 151B - Applied Numerical Methods - Homework 5

Darren Tsang, 405433124, Discussion 1A

Produced on Saturday, September 12, 2020 @ 06:58:03 PM

Question 1

Part A

We begin with Euler's Method and apply some substitutions:

$$w_{i+1} = w_i + hf(t_i, w_i) = w_i + h\lambda w_i = (1 + h\lambda)w_i = Q(h\lambda)w_i$$

The region of stability is as follows:

$$R = \{h\lambda \in \mathbb{C} : |1 + h\lambda| < 1\}$$

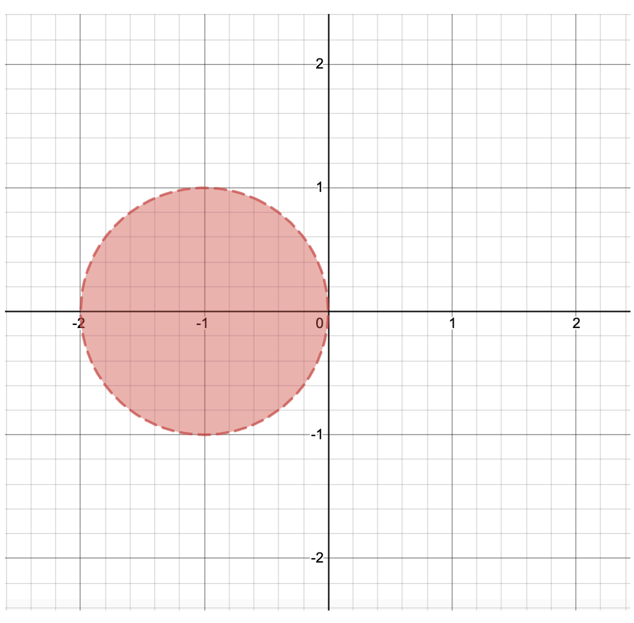
Since $z = a + ib = h\lambda \in \mathbb{C}$:

$$|1 + h\lambda| = |1 + a + ib|$$

Since $|a + ib| = \sqrt{a^2 + b^2}$, the region is:

$$R = \left\{a + ib \in \mathbb{C} = \sqrt{(1 + a)^2 + b^2} < 1\right\} \tag{1}$$

Here is the plot of the region in (1) through Desmos (Note that the real and imaginary component is the x- and y-axis respectively):



Part B

We begin with the Midpoint Method and substituting $f(t,w)=\lambda w$:

$$w_{i+1}=w_i+hf\left(t_i+\frac{h}{2},w_i+\frac{h}{2}f(t_i,w_i)\right)=w_i+hf\left(t_i+\frac{h}{2},w_i+\frac{h\lambda}{2}w_i\right)=w_i+h\lambda\left(w_i+\frac{h\lambda}{2}w_i\right)$$

Doing some more simplification:

$$w_{i+1}=\left(1+h\lambda+\frac{(h\lambda)^2}{2}\right)w_i=Q(h\lambda)w_i$$

The region of stability is as follows:

$$R=\left\{h\lambda\in\mathbb{C}:\left|1+h\lambda+\frac{(h\lambda)^2}{2}\right|<1\right\}$$

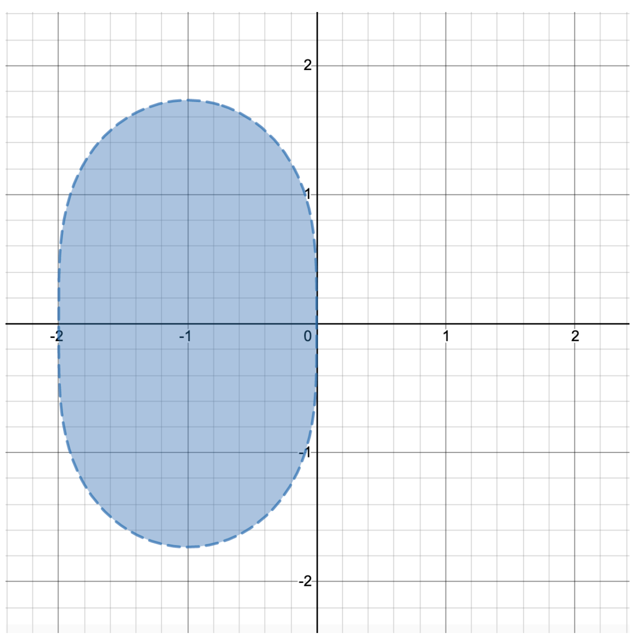
Since $z=a+ib=h\lambda\in\mathbb{C}$:

$$\left|1+h\lambda+\frac{(h\lambda)^2}{2}\right|=\left|1+a+ib+\frac{(a+ib)^2}{2}\right|=\left|1+a+ib+\frac{a^2+b^2+2iab}{2}\right|=\left|(1+a+\frac{a^2-b^2}{2})+(b+ab)i\right|$$

Since $|a+ib|=\sqrt{a^2+b^2}$, the region is:

$$R=\left\{a+ib\in\mathbb{C}:\sqrt{\left(1+a+\frac{a^2-b^2}{2}\right)^2+(b+ab)^2}<1\right\} \tag{2}$$

Here is the plot of region (2) through Desmos (Note that the real and imaginary component is the x- and y-axis respectively):



Question 2

We are given the following IVP:

$$y'' = y' + 2y + \cos(x), \qquad 0 \leq x \leq \frac{\pi}{2}, \qquad y(0) = -.3, \qquad y'(\frac{\pi}{2}) = -.1 \tag{1}$$

Note that the actual solution to (1) is:

$$y(x) = \frac{-(\sin x + 3\cos x)}{10}$$

From (1), we can create the following two IVPs:

$$y_1'' = y_1' + 2y_1 + \cos(x), \qquad 0 \leq x \leq \frac{\pi}{2}, \qquad y_1(0) = -.3, \qquad y_1'(0) = 0 \tag{2}$$

$$y_2'' = y_2' + 2y_2, \qquad 0 \leq x \leq \frac{\pi}{2}, \qquad y_2(0) = 0, \qquad y_2'(0) = 1 \tag{3}$$

Below is my code for the linear shooting method. To estimate $y_1(x_i)$ and $y_2(x_i)$, I will be using the Predictor Corrector algorithm from the previous homework. Then, with those estimates, we can calculate $y_{predict}(x_i)$ as follows:

$$y_{predict}(x_i) = y_1(x_i) + \frac{\beta - y_1(\pi/2)}{y_2(\pi/2)} y_2(x_i)$$

```
def predictor_corrector(func, start, end, initial, h):
  N = round((end-start)/h)
  x = [start]
  w_tilde = [initial]
  w = [initial]

  for i in range(0, N):
    w_tilde.append(w[i] + h*func(x[i] + h/2, w[i] + h*func(x[i], w[i])/2))
    w.append(w[i] + h/2*(func(x[i] + h, w_tilde[i+1]) + func(x[i], w[i])))
    x.append(x[i] + h)

  return(x, [item[0] for item in w])
```

```
def linear_shooting(func1, func2, alpha, beta, start, end, h, y_act):
  x, y1_estimate = predictor_corrector(func1, start, end, np.array([[alpha], [0]]), h)
  x, y2_estimate = predictor_corrector(func2, start, end, np.array([[0], [1]]), h)
  y_predict = y1_estimate + (beta - y1_estimate[-1])/y2_estimate[-1] * y2_estimate

  # change from list of lists to just one list
  y1_estimate = [round(j, 10) for i in y1_estimate for j in i]
  y2_estimate = [round(j, 10) for i in y2_estimate for j in i]
  y_predict = [round(j, 10) for i in y_predict for j in i]

  # get real value of y
  y_actual = [round(y_act(cur), 10) for cur in x]
  error = [abs(p - a) for p, a in zip(y_predict, y_actual)]

  return(pd.DataFrame({'x_i':x, 'y_1(x_i)':y1_estimate, 'y_2(x_i)':y2_estimate,
          'y_predict(x_i)':y_predict, 'y_actual(x_i)':y_actual, 'error':error}))
```

```
def func1(x, u):
  A = [[0,1], [2,1]]
  b = [[0], [math.cos(x)]]
  return(np.dot(A, u) + b)

def func2(x, u):
  A = [[0,1], [2,1]]
  return(np.dot(A, u))

def y_act(x):
  return(-.1*(3*math.cos(x) + math.sin(x)))
```

Part A

```
linear_shooting(func1, func2, -.3, -.1, 0, math.pi/2, math.pi/4, y_act)
```

```
##          x_i  y_1(x_i)  y_2(x_i)  y_predict(x_i)  y_actual(x_i)     error
## 0  0.000000 -0.300000  0.000000       -0.300000      -0.300000  0.000000
## 1  0.785398 -0.151660  1.457178       -0.280976      -0.282843  0.001866
## 2  1.570796  0.569257  7.541409       -0.100000      -0.100000  0.000000
```

Part B

```
linear_shooting(func1, func2, -.3, -.1, 0, math.pi/2, math.pi/8, y_act)
```

```
##          x_i  y_1(x_i)  y_2(x_i)  y_predict(x_i)  y_actual(x_i)     error
## 0  0.000000 -0.300000  0.000000       -0.300000      -0.300000  0.000000
## 1  0.392699 -0.264583  0.515225       -0.314738      -0.315432  0.000695
## 2  0.785398 -0.137579  1.486016       -0.282235      -0.282843  0.000608
## 3  1.178097  0.135854  3.522217       -0.207015      -0.207193  0.000177
## 4  1.570796  0.674472  7.955958       -0.100000      -0.100000  0.000000
```

Question 3

We are given the following IVP:

$$y'' = p(x)y' + q(x)y$$

We define y_2:

$$y_2(x) = 0$$

We can see that $y_2(x)$ is a solution to the IVP because:

$$0 = p(x) \cdot 0 + q(x) \cdot 0 = 0$$

Furthermore,

$$y_2(a) = y_2(b) = 0$$

Thus, by Corollary 11.2, $y_2(x) = 0$ is the only solution to the IVP.

Question 4

We are given the following BVP:

$$y'' = 4y - 4x, \qquad 0 \leq x \leq 1, \qquad y'(0) = 0, \qquad y(1) = 1$$

We will approximate the solution to the BVP by:

$$w_i'' - 4w_i = \left(\frac{-4i}{N}\right) \tag{1}$$

Center Points

We begin with the forward difference formula:

$$w_i' = \frac{w_{i+1} - w_i}{h} \tag{2}$$

Taking the derivative:

$$w_i'' = \frac{w_{i+1}' - w_i'}{h} \tag{3}$$

Applying (2) twice to (3), we get the centered difference formula:

$$w_i'' = \frac{w_{i+1} - 2w_i + w_{i-1}}{h^2} \tag{4}$$

We plug (2) into (1):

$$\frac{w_{i+1} - 2w_i + w_{i-1}}{h^2} - 4w_i = \frac{w_{i+1}}{h^2} - \left(4 + \frac{2}{h^2}\right)w_i + \frac{w_{i-1}}{h^2} = \left(\frac{-4i}{N}\right), \qquad \text{for } i = 1, ..., N-1 \tag{5}$$

Starting Point

We start with the following equations:

$$y(0) = y(0),$$
$$y(h) = y(0) + hy'(0) + \frac{h^2}{2}y''(0)$$
$$y(2h) = y(0) + 2hy'(0) + 2h^2y''(0)$$

We want to find a, b, c such that

$$ay(2h) + by(h) + cy(0) = hy'(0)$$

We plug in $y(2h), y(h), y(0)$, and obtain the following values:

$$a = \frac{-1}{2}, \qquad b = 2, \qquad c = \frac{-3}{2}$$

Thus, in our case, the second order approximation:

$$y'(0) = 0 \approx w_0' = h(\frac{-3}{2}w_0 + 2w_1 - \frac{1}{2}w_2) \longrightarrow w_0' = -3w_0 + 4w_1 - w_2 \tag{6}$$

Ending point

Note that we are given:

$$w_N = 1 \tag{7}$$

From (5), (6), and (7), the system is:

$$Aw = b,$$

where

$$A = \begin{bmatrix} -3 & 4 & -1 & 0 & \cdots & 0 \\ (1/h^2) & (-4-2/h^2) & (1/h^2) & 0 & \cdots & 0 \\ 0 & (1/h^2) & (-4-2/h^2) & (1/h^2) & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \cdots & 0 & (1/h^2) & (-4-2/h^2) & (1/h^2) \\ 0 & \cdots & \cdots & \cdots & \cdots & 1 \end{bmatrix},$$

$$w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_{N-1} \\ w_N \end{bmatrix}, \qquad b = \begin{bmatrix} 0 \\ \frac{-4}{N} \\ \frac{-8}{N} \\ \vdots \\ \frac{-4(N-1)}{N} \\ 1 \end{bmatrix}$$