MATH 417 502 HW8

Keegan Smith

November 4, 2024

Problem 1

Assuming we have:

$$f(t, y(t)) = \lambda \cdot y(t)$$

We can substitute into the original:

$$\begin{aligned}
y_{n+1} &= y_n + \frac{1}{2} \cdot h \cdot \lambda \cdot (y_n + y_{n+1}) \\
y_{n+1} - \frac{1}{2} \cdot h \cdot \lambda \cdot y_{n+1} &= y_n + \frac{1}{2} \cdot h \cdot \lambda \cdot y_n \\
y_{n+1}(1 - \frac{1}{2} \cdot h \cdot \lambda) &= y_n(1 + \frac{1}{2} \cdot h \cdot \lambda) \\
y_{n+1} &= \frac{y_n(1 + \frac{1}{2} \cdot h \cdot \lambda)}{1 - \frac{1}{2} \cdot h \cdot \lambda}
\end{aligned}$$

Thus we have the amplitude factor:

$$w(z) = \frac{1 + \frac{1}{2} \cdot z}{1 - \frac{1}{2} \cdot z}$$

We will call the real part of λ a, and the imaginary part of λ b.
We will call the real part of $w(z)$ $w_r(z)$, and imaginary $w_i(z)$:

$$w_r(z) = \frac{1 + \frac{1}{2}a}{1 - \frac{1}{2}a}$$

$$\begin{aligned}
w_i(z) &= \frac{\frac{1}{2}b}{-\frac{1}{2}b} \\
&= -1
\end{aligned}$$

the rule is stable when $|w(z)| < 1$ so we have:

$$
\begin{aligned}
|w(z)| &= \sqrt{w_r(z)^2 + w_i(z)^2} \\
\sqrt{w_r(z)^2 + w_i(z)^2} &< 1 \\
\sqrt{(\frac{1+\frac{1}{2}a}{1-\frac{1}{2}a})^2 + 1} &< 1 \\
(\frac{1+\frac{1}{2}a}{1-\frac{1}{2}a})^2 &< 1 \\
\frac{1+a+\frac{1}{4}a^2}{1-a+\frac{1}{4}a^2} &< 1 \\
1+a+\frac{1}{4}a^2 &< 1-a+\frac{1}{4}a^2 \\
2a &< 0 \\
a &< 0
\end{aligned}
$$

This when the real part of λ is less than 0, the trapezoidal rule is stable.

Problem 2

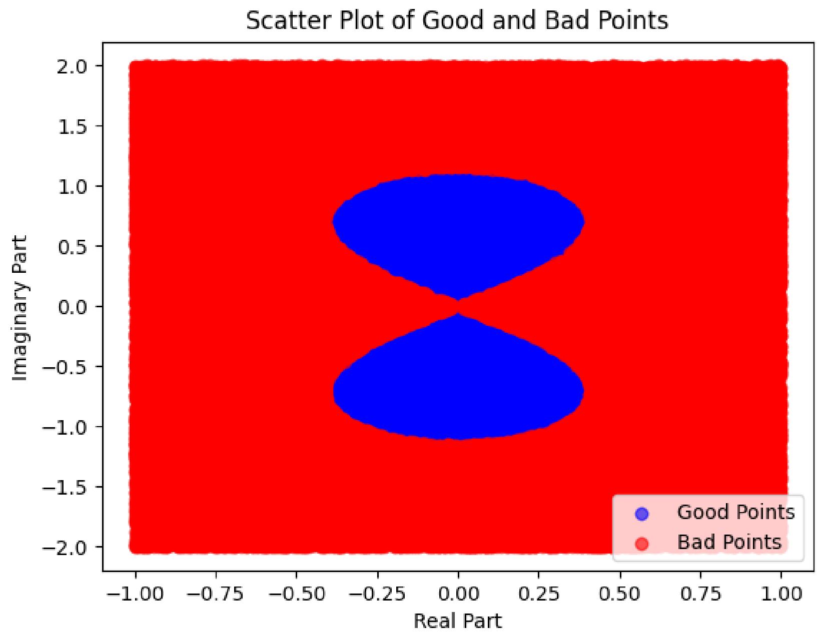
again assuming $f(t, y(t)) = \lambda \cdot y(t)$:

$$
\begin{aligned}
y_{n+1} &= y_n + h \cdot f(t_n + \frac{1}{2}h, y_n + \frac{1}{2} \cdot h \cdot \lambda \cdot y_n) \\
y_{n+1} &= y_n + h \cdot \lambda \cdot (y_n + \frac{1}{2} \cdot h \cdot \lambda \cdot y_n) \\
y_{n+1} &= y_n + h \cdot \lambda \cdot y_n + h \cdot \lambda \cdot \frac{1}{2} \cdot h \cdot \lambda \cdot y_n \\
y_{n+1} &= y_n + h \cdot \lambda \cdot y_n + \frac{1}{2} \cdot h^2 \cdot \lambda^2 \cdot y_n \\
y_{n+1} &= y_n(1 + h \cdot \lambda + \frac{1}{2} \cdot h^2 \cdot \lambda^2)
\end{aligned}
$$

thus we have:

$$
w(z) = 1 + z + \frac{1}{2}z^2
$$

Testing the points yields a Hyperboloid looking figure, where the function is stable within the hyperboloid:




```
import random
import matplotlib.pyplot as plt
class Complex:
    def __init__(self, real, imaginary):
        self.real = real
        self.imaginary = imaginary
    def magnitude(self):
        return (self.real**2 + self.imaginary**2) ** .5
    def add(a, b):
        return Complex(a.real + b.real, a.imaginary + b.imaginary)
    def square(a):
        return Complex(a.real**2 - a.imaginary**2, 2 * a.real * a.imaginary)
def w(z):
    one = Complex(1, 0)
    z_square = Complex.square(z)
    z_square_term = Complex(1/2 * z_square.real, 1/2 * z_square.imaginary)
    result = Complex.add(one, Complex.add(z_square, z_square_term))
    return result
def magnitude(z):
    return w(z).magnitude()
```

```python

def plot(bad_points, good_points):
    plt.scatter(*zip(*good_points), color='blue', label='Good Points', alpha=0.6)
    plt.scatter(*zip(*bad_points), color='red', label='Bad Points', alpha=0.6)
    plt.xlabel("Real Part")
    plt.ylabel("Imaginary Part")
    plt.legend()
    plt.title("Scatter Plot of Good and Bad Points")
    plt.show()
    return

def generate_points(real_interval, imag_interval, num_points):
    result = []
    for i in range(0, num_points):
        real = random.uniform(real_interval[0], real_interval[1])
        imag = random.uniform(imag_interval[0], imag_interval[1])
        result.append([real, imag])
    return result

def main():
    real_interval = [-1, 1]
    imag_interval = [-2, 2]
    num_points = 100000
    points = generate_points(real_interval,imag_interval, num_points)
    good_points = []
    bad_points = []
    for i in range(0, len(points)):
        my_complex = Complex(points[i][0], points[i][1])
        my_magnitude = magnitude(my_complex)
        if(my_magnitude < 1):
            good_points.append(points[i])
        else:
            bad_points.append(points[i])
    plot(bad_points, good_points)

if __name__ == "__main__":
    main()
```