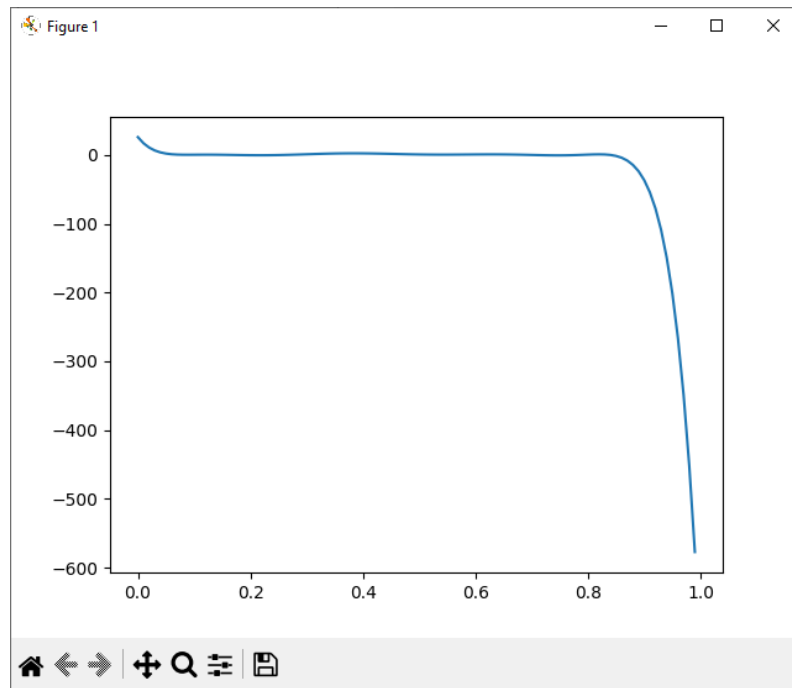# Differential Equations Coding Project

Cam Walter

April 24, 2024

Questions

1. (a) $f(x) = -5 + 0.5x^2 + x^5$

   (b) The solution would not work, since we would need a degree 6 polynomial now. This makes sense, since if we added a row to multiply, there would not be enough columns in A to match up with x's rows.

   (c) $det(V) = 0 \iff \exists x_i, x_j(x_i = x_j)$

   (d)

   (e) We can use spline interpretation to form a piecewise function comprised ofseveral small polynomials to make the graph much better reflect the data points.

2. (a)

Code

```python
import numpy as np
import numpy.matlib as ml
import numpy.linalg as linalg
import matplotlib.pyplot as plt

def interpolate(x_in: list[float], y_in: list[float]) -> list[float]:
    a: list[list[float]] = []
    for i in range(len(x_in)):
        a.append([])
        for j in range(len(x_in)):
            a[i].append(pow(x_in[i], j))

    a_mat = ml.mat(a)
    a_inverse = linalg.inv(a_mat)

    output = np.matmul(a_inverse, ml.transpose(y_in)).tolist()

    for i in range(len(output)):
        for j in range(len(output[i])):
            output[i][j] = round(output[i][j], 2)

    return output

def power_method(AtA: list[list[float]], x_0: list[float]) -> float:
    x = x_0
    eigenvalue = 0
    for _ in range(20):
        y_i = np.matmul(AtA, x)

        y_transpose_y = np.matmul(ml.transpose(y_i), y_i)
        eigenvalue = np.sqrt(np.sqrt(y_transpose_y))
        x = y_i / eigenvalue

    return eigenvalue

def main():
    # Question 1
    print("1a.", end=" ")
    one_a = interpolate(
        x_in = [0., -1., 1., 2., 4., -2.],
        y_in = [-5., -5.5, -3.5, 29., 1027., -35.]
    )
    print(one_a)
```

```python
print("1d.", end=" ")
# x_in = np.random.uniform(low=0, high=1, size=10)
# y_in = np.random.uniform(low=0, high=1, size=10)
# f = open("points.txt", "w")
# for i in range(10):
#     f.write(str(x_in[i]))
#     f.write(" ")
#     f.write(str(y_in[i]))
#     f.write("\n")
# f.close()

x_in: list[float] = []
y_in: list[float] = []

f = open("points.txt", "r")
lines = f.read().splitlines()
for line in lines:
  [x, y] = line.split(" ")
  x_in.append(float(x))
  y_in.append(float(y))

one_d = interpolate(x_in, y_in)
print(one_d)

def graph(formula, x_range):
  x = np.array(x_range)
  y = formula(x)
  plt.plot(x, y)

def my_formula(x):
  y = 0
  for i in range(len(one_d[0])):
    y += one_d[0][i] * pow(x, i)
  return y

graph(my_formula, np.arange(0., 1., 0.01))
plt.show()

# Question 2
a: list[list[float]] = []
for i in range(10):
  a.append([x_in[i], y_in[i]])

AtA = np.matmul(ml.transpose(a), a)
eigen1 = power_method(AtA, [1., 1.])
eigen2 = power_method(linalg.inv(AtA), [1., 1.])
```

3

```python
    print(f"2a. eigenvalue={eigen1}")
    print(f"2b. eigenvalue={eigen2}")

main()
```