Math104A Homework4

December 8, 2024

Brandon Su

Question 1:

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

[2]:
```
def natural_spline(xs, ys):
    n = len(xs) - 1
    a = list(ys)
    b = [0] * n
    d = [0] * n
    h = []
    for i in range(n):
        h.append(xs[i + 1] - xs[i])  # each interval width

    alpha = [0] * (n + 1)

    # alpha values
    for i in range(1, n):
        alpha[i] = (3 / h[i]) * (a[i + 1] - a[i]) - (3 / h[i - 1]) * (a[i] -␣
↪a[i - 1])

    c, l, mu, z = np.zeros(n + 1), np.zeros(n + 1), np.zeros(n + 1), np.zeros(n␣
↪+ 1)
    l[0] = 1
    l[n] = 1

    for i in range(1, n):
        l[i] = 2 * (xs[i + 1] - xs[i - 1]) - h[i - 1] * mu[i - 1]
        mu[i] = h[i] / l[i]
        z[i] = (alpha[i] - h[i - 1] * z[i - 1]) / l[i]

    for j in range(n - 1, -1, -1):
        c[j] = z[j] - mu[j] * c[j + 1]
        b[j] = (a[j + 1] - a[j]) / h[j] - h[j] * (2 * c[j] + c[j + 1]) / 3
        d[j] = (c[j + 1] - c[j]) / (3 * h[j])
```

```
    # combine coefficients
    spline_coefficients = []
    for i in range(n):
        spline_coefficients.append((a[i], b[i], c[i], d[i], xs[i]))  # append␣
↪each tuple
    return spline_coefficients
```

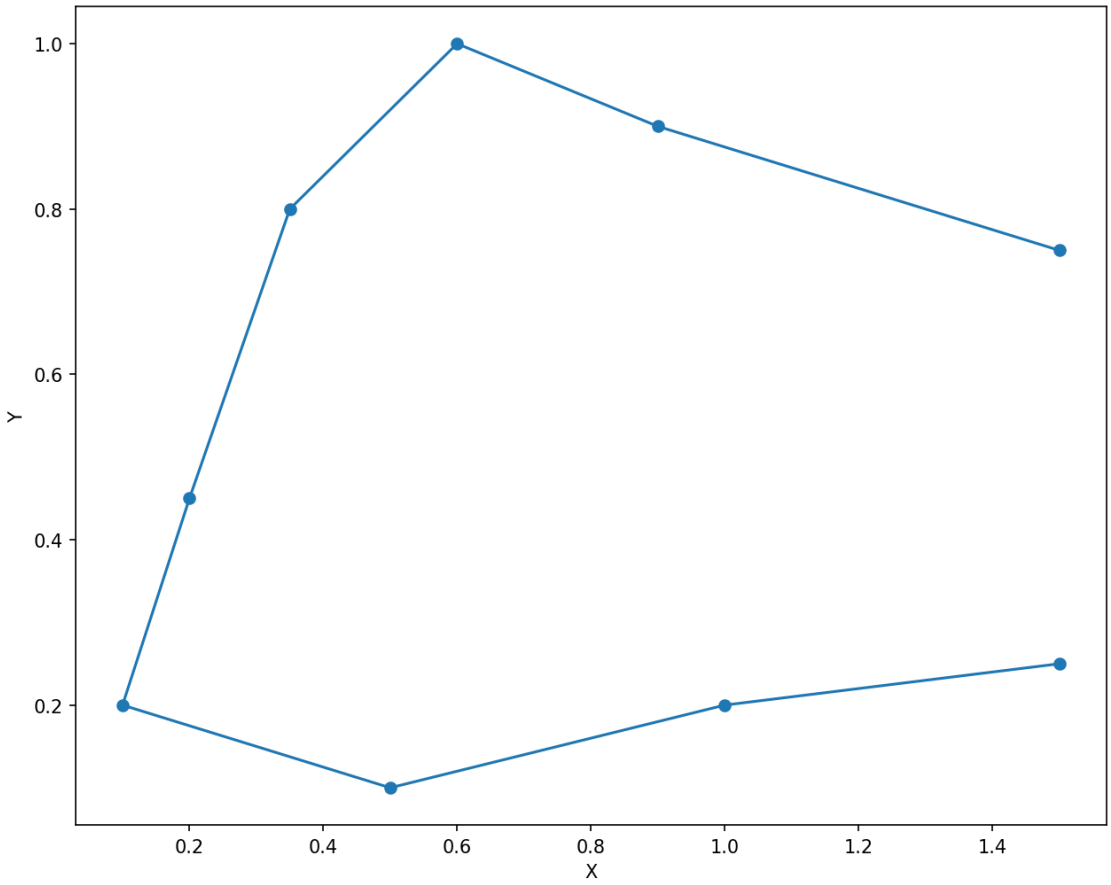
[3]:
```
# evaluate with an example
x = [0, 2, 4, 6, 8]
y = [1, 3, 2, 5, 4]

spline_coeffs = natural_spline(x, y)
print("Spline Coefficients are:")
for coeff in spline_coeffs:
    print(coeff)
```

```
Spline Coefficients are:
(1, 1.5803571428571428, 0.0, -0.14508928571428573, 0)
(3, -0.16071428571428575, -0.8705357142857143, 0.3504464285714286, 2)
(2, 0.5625, 1.2321428571428572, -0.3816964285714286, 4)
(5, 0.9107142857142858, -1.0580357142857144, 0.17633928571428573, 6)
```

Question 2:

[4]:
```
t_j = np.array([0.000, 0.618, 0.935, 1.255, 1.636, 1.905, 2.317, 2.827, 3.330])
x_j = np.array([1.50, 0.90, 0.60, 0.35, 0.20, 0.10, 0.50, 1.00, 1.50])
y_j = np.array([0.75, 0.90, 1.00, 0.80, 0.45, 0.20, 0.10, 0.20, 0.25])
```

[5]:
```
plt.figure(figsize=(10,8), dpi=150)
plt.plot(x_j, y_j, marker="o")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```



[6]:
```
# compute splines
spline_1 = natural_spline(t_j, x_j)
spline_2 = natural_spline(t_j, y_j)
```

[7]:
```
spline_coeffs_1 = pd.DataFrame(spline_1, columns = ["a", "b", "c", "d", "t_j"])
spline_coeffs_2 = pd.DataFrame(spline_2, columns = ["a", "b", "c", "d", "t_j"])
spline_coeffs_1['t_j'] = t_j[:-1]
spline_coeffs_2['t_j'] = t_j[:-1]
spline_coeffs_1
```

[7]:

	a	b	c	d	t_j
0	1.50	-0.974898	0.000000	0.010537	0.000
1	0.90	-0.962825	0.019535	0.102103	0.618
2	0.60	-0.919659	0.116635	0.987167	0.935
3	0.35	-0.541755	1.064316	-1.773549	1.255
4	0.20	-0.503097	-0.962851	5.394568	1.636
5	0.10	0.149959	3.390566	-3.393333	1.905
6	0.50	1.215791	-0.803594	0.670643	2.317
7	1.00	0.919428	0.222490	-0.147442	2.827

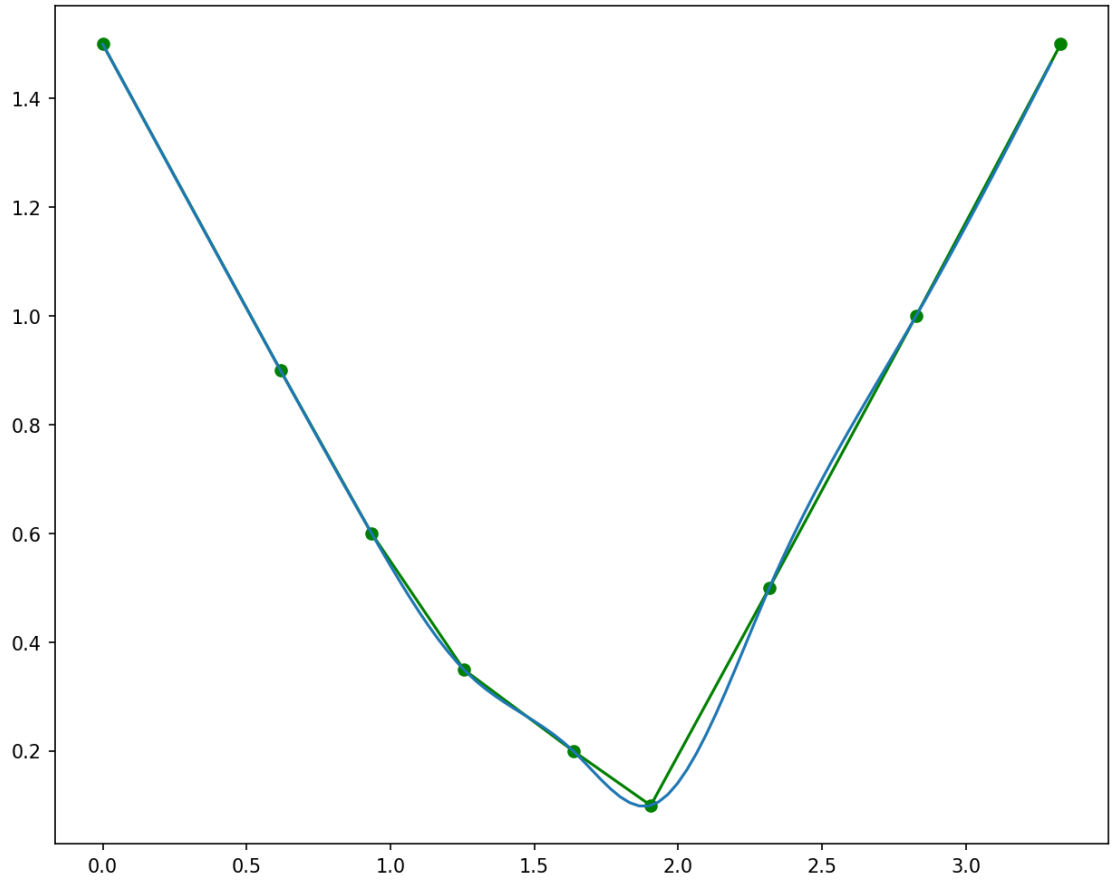
[8]:
```
spline_coeffs_2
```

[8]:

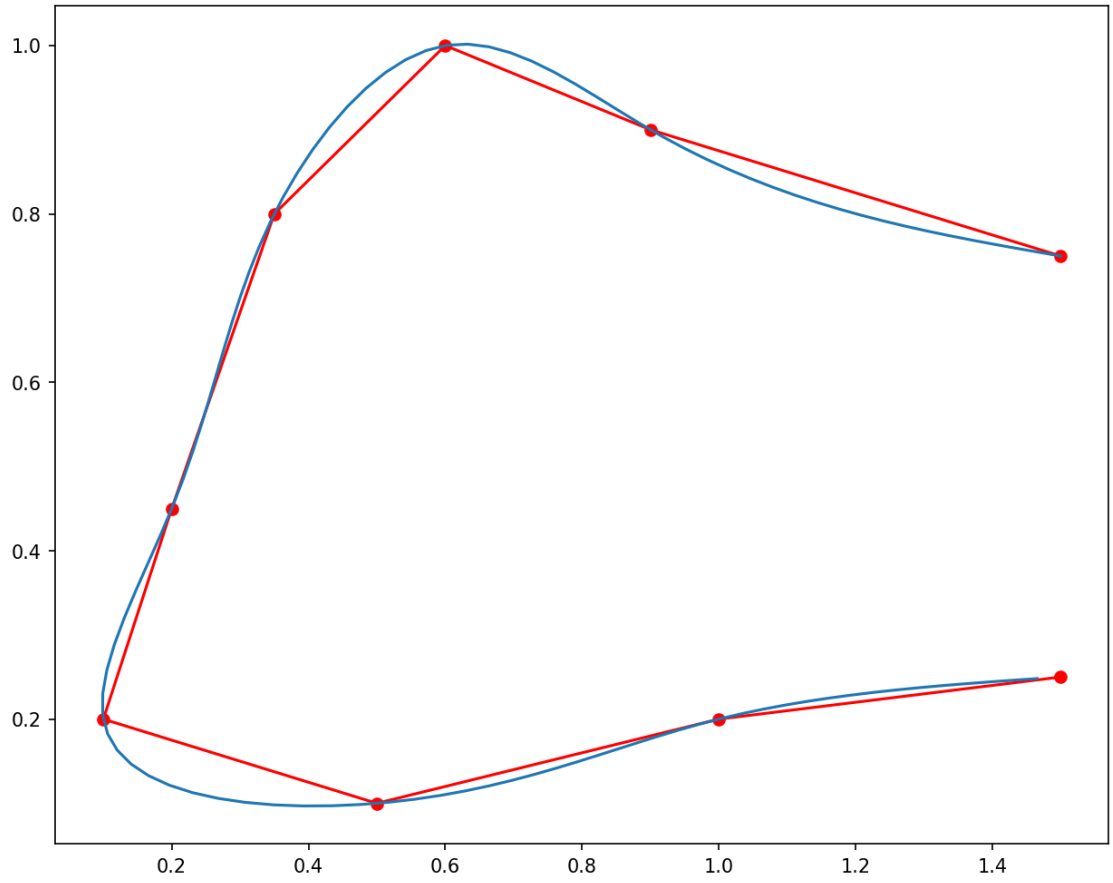
	a	b	c	d	t_j
0	0.75	0.136947	0.000000	0.276944	0.000
1	0.90	0.454261	0.513454	-3.001012	0.618
2	1.00	-0.124915	-2.340509	2.430450	0.935
3	0.80	-0.876207	-0.007277	-0.273187	1.255
4	0.45	-1.000720	-0.319530	2.173896	1.636
5	0.20	-0.700711	1.434804	-0.784397	1.905
6	0.10	0.082127	0.465289	-0.474226	2.317
7	0.20	0.186683	-0.260277	0.172483	2.827

[9]:
```
def spline_parametrization(X_s, C_s, x):
    i = 0
    num = len(X_s) - 1

    for k in range(num):
        if x > X_s[k]:
            i = k

    coefficients = C_s[i]
    a, b, c, d, xj = coefficients[0], coefficients[1], coefficients[2],␣
 ↪coefficients[3], coefficients[4]

    y = a + b * (x - xj) + c * (x - xj)**2 + d * (x - xj)**3

    return y
```

[14]:
```
t_dots = []
for i in range(100):
    t_dots.append(i * (max(t_j) - min(t_j)) * 0.01)
x_dots = []
for i in range(100):
    x_dots.append(spline_parametrization(t_j, spline_1, t_dots[i]))

plt.figure(figsize=(10,8), dpi=150)
plt.plot(t_j, x_j, marker='o', color="green")
plt.plot(t_dots, x_dots)
plt.show()
```



[12]:
```
y_dots = []
for i in range(100):
    y_dots.append(spline_parametrization(t_j, spline_2, t_dots[i]))

plt.figure(figsize=(10,8), dpi=150)
plt.plot(x_j, y_j, marker='o', color='red')
plt.plot(x_dots, y_dots)
plt.show()
```


1.0
0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
1.0
1.2
1.4