Computional Homework1 Report

Student: 纪浩正, jihz2023@mail.sustech.edu.cn

1 Quadratic Interpolation Approximation of $\sqrt{115}$

Given $y = \sqrt{x}$, use $x_0 = 100$, $x_1 = 121$, and $x_2 = 144$ to approximate $\sqrt{115}$ with quadratic interpolation, including an upper bound for the error.

1.1 Lagrange Quadratic Interpolation Formula

The function values are:

$$y_0 = \sqrt{100} = 10, \qquad y_1 = \sqrt{121} = 11, \qquad y_2 = \sqrt{144} = 12$$

The interpolation polynomial:

$$\begin{aligned} P_2(x) &= y_0 \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + y_1 \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + y_2 \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} \\ &= 10 \cdot \frac{(x-121)(x-144)}{(100-121)(100-144)} \\ &\quad + 11 \cdot \frac{(x-100)(x-144)}{(121-100)(121-144)} \\ &\quad + 12 \cdot \frac{(x-100)(x-121)}{(144-100)(144-121)} \end{aligned}$$

1.2 Calculation at $x = 115$

$$(x-121)(x-144) = (115-121)(115-144) = (-6)(-29) = 174$$

$$(100-121)(100-144) = (-21)(-44) = 924$$

$$(x-100)(x-144) = (15)(-29) = -435$$

$$(121-100)(121-144) = (21)(-23) = -483$$

$$(x-100)(x-121) = 15 \times (-6) = -90$$

$$(144-100)(144-121) = 44 \times 23 = 1012$$

Now plug into $P_2(115)$:

$$\begin{aligned} P_2(115) &= 10 \times \frac{174}{924} + 11 \times \frac{-435}{-483} + 12 \times \frac{-90}{1012} \\ &= 10.723 \end{aligned}$$

$$\boxed{\sqrt{115} \approx 10.723}$$

1.3 Error Estimate

The interpolation error formula (for quadratic):

$$|E(x)| = \left| \frac{f^{(3)}(\xi)}{3!}(x - x_0)(x - x_1)(x - x_2) \right|, \quad \xi \in [100, 144]$$

For $f(x) = x^{1/2}$:

$$f^{(3)}(x) = \frac{3}{8}x^{-5/2}$$

On $[100, 144]$, the maximum occurs at $x = 100$:

$$|f^{(3)}(\xi)| \leq \frac{3}{8} \cdot 100^{-5/2} = \frac{3}{8} \cdot 10^{-5} = 3.75 \times 10^{-6}$$

At $x = 115$:

$$(x - 100)(x - 121)(x - 144) = 15 \times (-6) \times (-29) = 2610$$

So,

$$|E(115)| \leq \frac{3.75 \times 10^{-6}}{6} \cdot 2610 = 1.63 \times 10^{-3}$$

$$\boxed{|E(115)| \leq 1.63 \times 10^{-3}}$$

2 Compare Accuracies of Linear and Quadratic Interpolations under Varying Intervals for $f(x) = e^x$

2.1 Defining the Intervals

I choose several special intervals:

$$[-200, -100],\ [-200, 100],\ [-10, 0],\ [0, 1],\ [1, 5],\ [1, 100],\ [100, 101],\ [100, 200]$$

For each interval $[x_0, x_2]$, I vary the second interpolation point x_1 and the interpolated value x throughout the interval. This produces error surfaces which allow for a detailed comparison of interpolation accuracy.

```
import numpy as np
import matplotlib.pyplot as plt
import os

os.makedirs("images", exist_ok=True)


def f(x):
    return np.exp(x)

x0 = 1
x2 = 5
n = 2000

x1 = np.linspace(x0, x2, n)
x = np.linspace(x0, x2, n)
X1, X = np.meshgrid(x1, x)
```

2.2 Calculating Linear and Quadratic Interpolations and the Error Surface

Interpolation Formulas: For points x_0, x_1, x_2 with function values $y_0 = f(x_0), y_1 = f(x_1), y_2 = f(x_2)$:

Linear (Newton form):

$$P_1(x) = y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0)$$

Quadratic (Newton form):

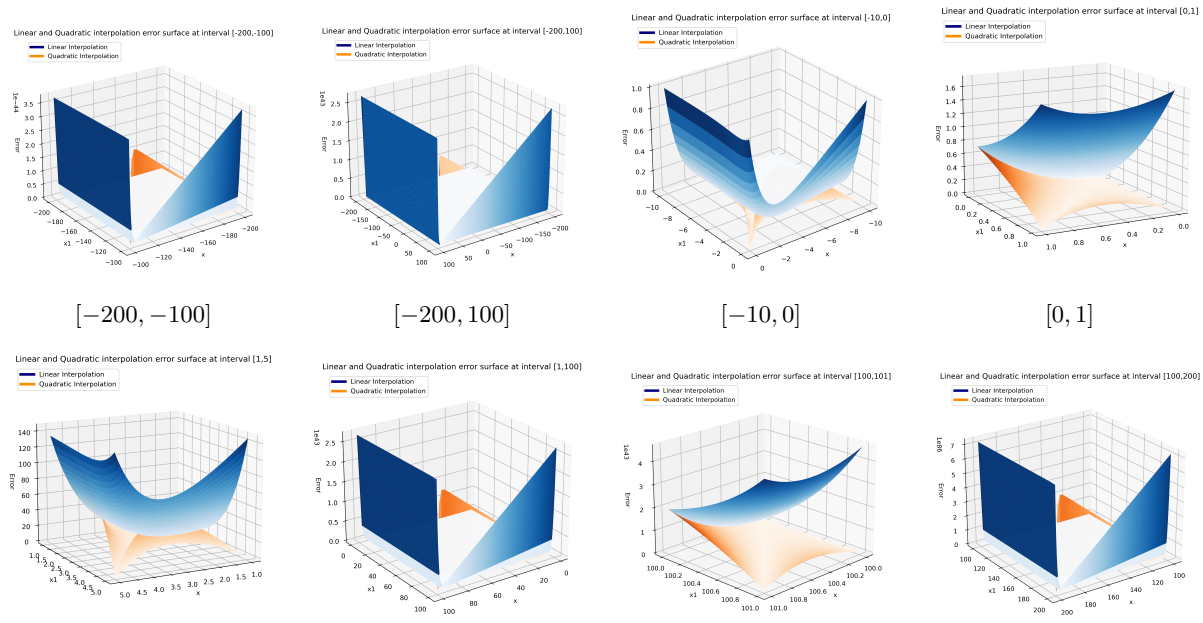
$$P_2(x) = y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0) + \left(\frac{y_2 - y_0}{x_2 - x_0} - \frac{y_1 - y_0}{x_1 - x_0}\right)\frac{(x - x_0)(x - x_1)}{x_2 - x_1}$$

```
lin_In = (f(x0) - f(X1)) / (x0 - X1) * (X - X1) + f(x0)


quad_In = (
    f(x0)
    + (X - x0) / (X1 - x0) * (f(X1) - f(x0))
    + ((f(x2) - f(x0)) / (x2 - x0) - (f(x0) - f(X1)) / (x0 - X1))
    * (x - x0)
    * (x - X1)
    / (x2 - X1)
)
```

2.3 Drawing and Comparing the Error Surfaces

Below are the error surfaces for linear and quadratic interpolations over eight different intervals for $f(x) = e^x$. The plots demonstrate that quadratic interpolation generally provides much higher accuracy than linear interpolation, especially over wider intervals.



$[-200, -100]$ $[-200, 100]$ $[-10, 0]$ $[0, 1]$

$[1, 5]$ $[1, 100]$ $[100, 101]$ $[100, 200]$

Figure 1 Error surfaces for linear and quadratic interpolation on eight different intervals.

```
lin_err = abs(lin_In - f(X))
quad_err = abs(quad_In - f(X))
```

```
fig = plt.figure(figsize=(12, 7))
ax = fig.add_subplot(111, projection="3d")
ax.plot_surface(X, X1, lin_err, cmap="Blues")
ax.plot_surface(X, X1, quad_err, cmap="Oranges")
ax.set_xlabel("x")
ax.set_ylabel("x1")
ax.set_zlabel("Error")
ax.set_title(
    "Linear and Quadratic interpolation error surface at interval [%d,%d]" % (x0, x2)
)
from matplotlib.lines import Line2D

custom_lines = [
    Line2D([0], [0], color="navy", lw=4),
    Line2D([0], [0], color="darkorange", lw=4),
]
ax.legend(
    custom_lines, ["Linear Interpolation", "Quadratic Interpolation"], loc="upper left"
)
ax.view_init(elev=23, azim=55)
plt.draw()
plt.savefig(
    "images/Linear and Quadratic interpolation error surface at interval [%d,%d].png"
    % (x0, x2),
    dpi=300,
    bbox_inches="tight",
    pad_inches=0.2,
)
plt.show()
```

Conclusion: As shown above, quadratic interpolation consistently yields a lower interpolation error than linear interpolation, especially as the interval widens.