MATH 4640 Numerical Analysis - HW 2 Solutions

Austin Barton

February 27, 2025

Problem 1:

$y_0 = -1 \quad y_4 = 1$

$\Delta x_i = 1 \quad \forall i \in \{0,1,2,3,4\}$

$\Delta x_i + \Delta x_{i+1} = 2 \quad \forall i \in \{0,1,2,3\}$

Note: This is important to understand the simplifications to the general form below

$$A = \begin{bmatrix} \frac{1}{3} & \frac{1}{6} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & 0 \\ 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ 0 & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ 0 & 0 & 0 & \frac{1}{6} & \frac{1}{3} \end{bmatrix} \qquad M = \begin{bmatrix} M_0 \\ M_1 \\ M_2 \\ M_3 \\ M_4 \end{bmatrix}$$

$$D = \begin{bmatrix} f_1 - f_0 - y_0 \\ f_2 - f_1 - (f_1 - f_0) \\ f_3 - f_2 - (f_2 - f_1) \\ f_4 - f_3 - (f_3 - f_2) \\ y_4 - (f_4 - f_3) \end{bmatrix} = \begin{bmatrix} -0.3 + 1 \\ -0.3 + 0.3 \\ -0.2 + 0.3 \\ 0.2 + 0.2 \\ 1 - 0.2 \end{bmatrix} = \begin{bmatrix} 0.7 \\ 0 \\ 0.1 \\ 0.4 \\ 0.8 \end{bmatrix}$$

We want to find M s.t.

$AM = D$, below is it expresed as a linear system of equations

(1) $\frac{1}{3}M_0 + \frac{1}{6}M_1 = 0.7$

(2) $\frac{1}{6}M_0 + \frac{2}{3}M_1 + \frac{1}{6}M_2 = 0$

(3) $\frac{1}{6}M_1 + \frac{2}{3}M_2 + \frac{1}{6}M_3 = 0.1$

(4) $\frac{1}{6}M_2 + \frac{2}{3}M_3 + \frac{1}{6}M_4 = 0.4$

(5) $\frac{1}{6}M_3 + \frac{1}{3}M_4 = 0.8$

$$AM = \begin{bmatrix} \frac{2M_0 + M_1}{6} \\ \frac{4M_1 + M_0 + M_2}{6} \\ \frac{4M_2 + M_1 + M_3}{6} \\ \frac{4M_3 + M_2 + M_4}{6} \\ \frac{2M_4 + M_3}{6} \end{bmatrix} = \begin{bmatrix} 0.7 \\ 0 \\ 0.1 \\ 0.4 \\ 0.8 \end{bmatrix}$$

- $\frac{2M_4 + M_3}{6} = 0.8 \Rightarrow 2M_4 + M_3 = 4.8$

  $\Rightarrow M_4 = 2.4 - \frac{1}{2} M_3$

- $\frac{4M_3 + M_2 + M_4}{6} = 0.4 \Rightarrow 4M_3 + M_2 + M_4 = 2.4$

  $\Rightarrow M_2 = 2.4 - M_4 - 4M_3 = 2.4 - (2.4 - \frac{1}{2}M_3) - 4M_3$

  $= \frac{1}{2}M_3 - 4M_3 = -\frac{7}{2}M_3$

  $M_2 = -7/2\, M_3$

- $\frac{4M_2 + M_1 + M_3}{6} = 0.1 \Rightarrow 4M_2 + M_1 + M_3 = 0.6$

  $\Rightarrow M_1 = 0.6 - 4M_2 - M_3 = 0.6 - 4(-\frac{7}{2}M_3) - M_3$

  $= 0.6 + 14M_3 - M_3 = 0.6 + 13M_3$

  $M_1 = 0.6 + 13M_3$

- $\frac{4M_1 + M_0 + M_2}{6} = 0 \Rightarrow M_0 = -4M_1 - M_2$

  $= -4(0.6 + 13M_3) - (-7/2\, M_3) = -2.4 - 52M_3 + \frac{7}{2}M_3$

$$M_0 = -2.4 - 52M_3 + \frac{7}{2}M_3 = -2.4 - \frac{97}{2}M_3$$

$$\frac{2M_0 + M_1}{6} = 0.7 = \frac{2\left(-2.4 - \frac{97}{2}M_3\right) + (0.6 + 13M_3)}{6}$$

$$= \frac{-4.8 - 97M_3 + 0.6 + 13M_3}{6}$$

$$\Rightarrow 4.2 = -4.2 - 84M_3 \Rightarrow 8.4 = -84M_3$$

$$\Rightarrow -0.1 = M_3$$

$M_3 = -0.1$, thus,

$$M_0 = -2.4 - 52(-0.1) + \frac{7}{2}(-0.1) = 2.45$$

$$M_1 = 0.6 + 13(-0.1) = -0.7$$

$$M_2 = -7/2(-0.1) = 0.35$$

$$M_4 = 2.4 - \frac{1}{2}(-0.1) = 2.45$$

$$M = \begin{bmatrix} 2.45 \\ -0.7 \\ 0.35 \\ -0.1 \\ 2.45 \end{bmatrix}$$

$$p_0(x) = \frac{(1-x)^3 M_0 + x^3 M_1}{6} + (1-x)(1.1) + x0.8$$

$$- \frac{1}{6}\left((1-x)M_0 + xM_1\right) \quad , \; x \in [0,1]$$

$$p_1(x) = \frac{(2-x)^3 M_1 + (x-1)^3 M_2}{6} + (2-x)0.8 + (x-1)(0.5)$$

$$- \frac{1}{6}\left((2-x)M_1 + (x-1)M_2\right) \quad , x \in [1,2]$$

$$p_2(x) = \frac{(3-x)^3 M_2 + (x-2)^3 M_3}{6} + (3-x)0.5 + (x-2)0.3$$

$$- \frac{1}{6}\left((3-x)M_2 + (x-2)M_3\right) \quad , \; x \in [2,3]$$

$$p_3(x) = \frac{(4-x)^3 M_3 + (x-3)^3 M_4}{6} + (4-x)0.3 + (x-3)0.5$$

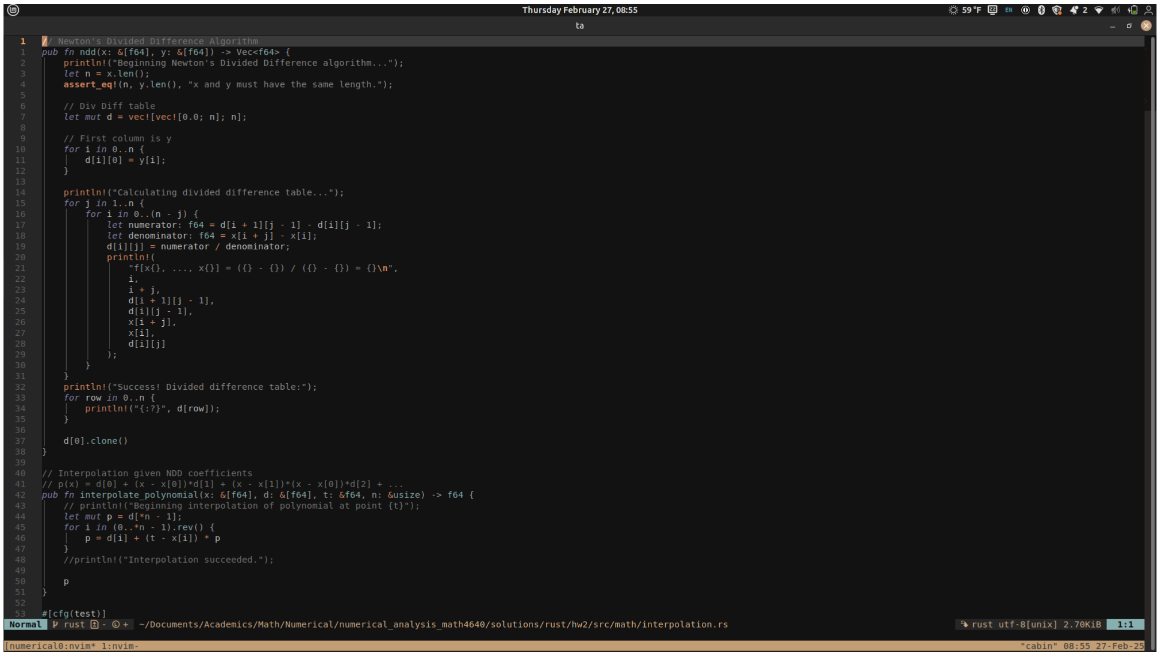
$$- \frac{1}{6}\left((4-x)M_3 + (x-3)M_4\right) \quad , \; x \in [3,4]$$

where
$M_0 = 2.45$
$M_1 = -0.7$
$M_2 = 0.35$
$M_3 = -0.1$
$M_4 = 2.45$

Problem 2:

Problem 3:

Problem 4:



```rust
// Newton's Divided Difference Algorithm
pub fn ndd(x: &[f64], y: &[f64]) -> Vec<f64> {
    println!("Beginning Newton's Divided Difference algorithm...");
    let n = x.len();
    assert_eq!(n, y.len(), "x and y must have the same length.");

    // Div Diff table
    let mut d = vec![vec![0.0; n]; n];

    // First column is y
    for i in 0..n {
        d[i][0] = y[i];
    }

    println!("Calculating divided difference table...");
    for j in 1..n {
        for i in 0..(n - j) {
            let numerator: f64 = d[i + 1][j - 1] - d[i][j - 1];
            let denominator: f64 = x[i + j] - x[i];
            d[i][j] = numerator / denominator;
            println!(
                "f[x{}, ..., x{}] = ({} - {}) / ({} - {}) = {}\n",
                i,
                i + j,
                d[i + 1][j - 1],
                d[i][j - 1],
                x[i + j],
                x[i],
                d[i][j]
            );
        }
    }
    println!("Success! Divided difference table:");
    for row in 0..n {
        println!("{:?}", d[row]);
    }

    d[0].clone()
}

// Interpolation given NDD coefficients
// p(x) = d[0] + (x - x[0])*d[1] + (x - x[1])*(x - x[0])*d[2] + ...
pub fn interpolate_polynomial(x: &[f64], d: &[f64], t: &f64, n: &usize) -> f64 {
    // println!("Beginning interpolation of polynomial at point {t}");
    let mut p = d[*n - 1];
    for i in (0..*n - 1).rev() {
        p = d[i] + (t - x[i]) * p
    }
    //println!("Interpolation succeeded.");

    p
}

#[cfg(test)]
```

Figure 1: Screenshot of code for the Newton's Divided Difference (NDD) algorithm (upper) and the interpolate using the NDD coefficients (lower).

Problem 5:

Problem 6:

Problem 7:

Problem 8:

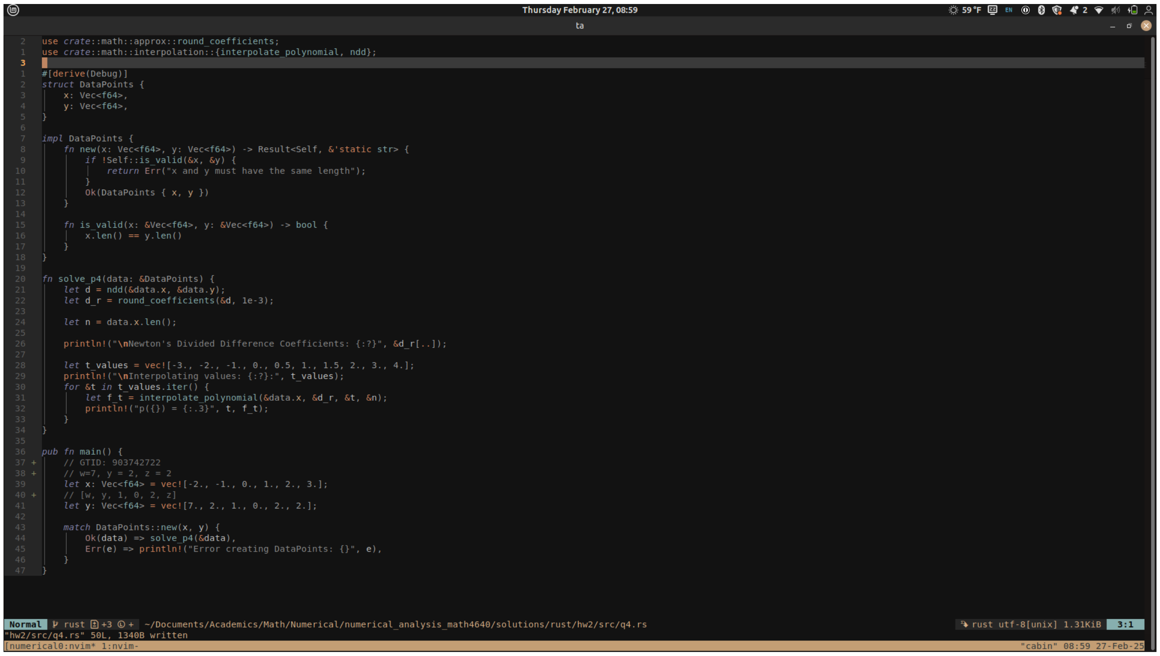


Figure 2: Screenshot of code using the DivDif and Interpolate algorithms.

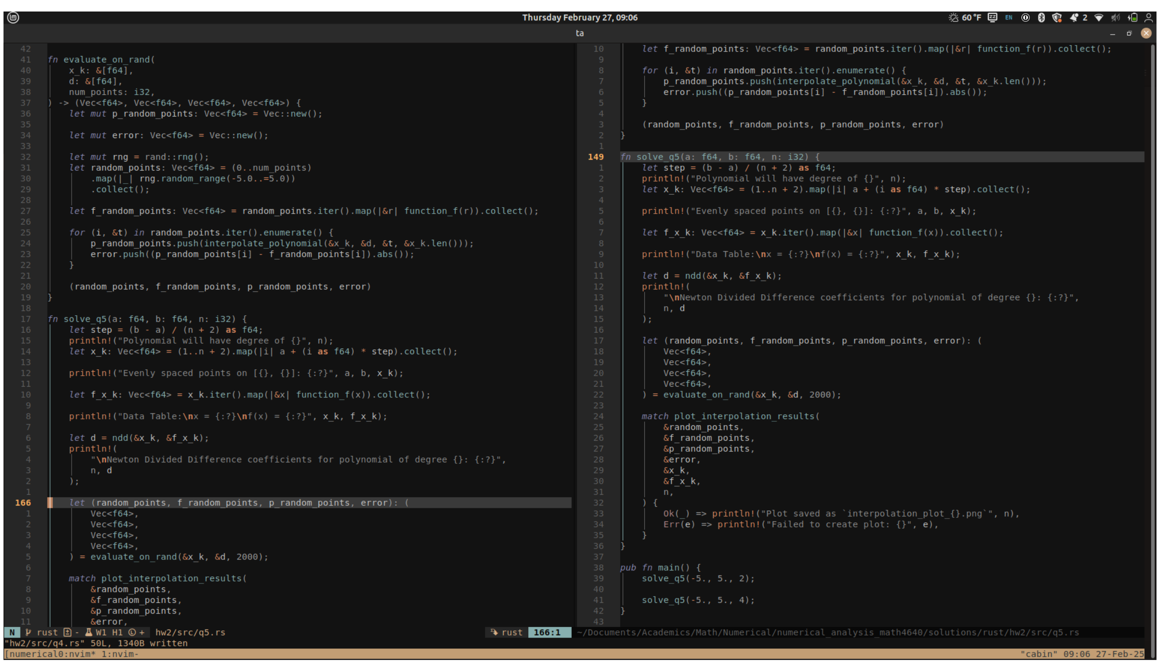


Figure 3: Screenshot of code using DivDif and Interpolate.

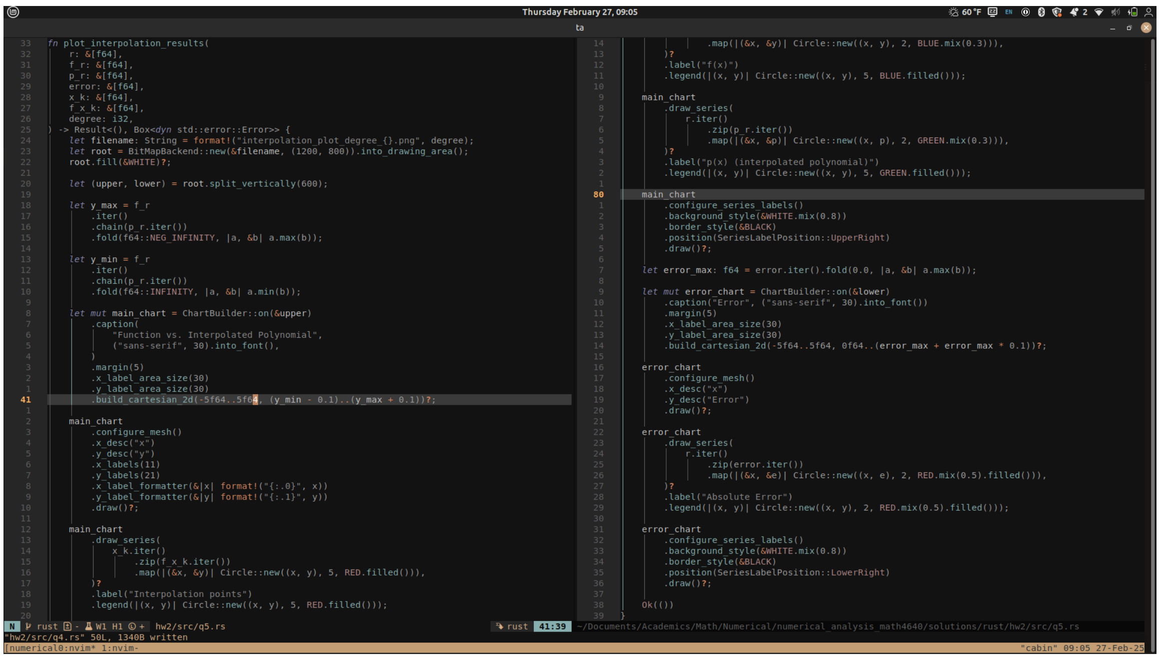


Figure 4: Screenshot of code plotting the true function, the interpolated polynomial, the interpolated points, and the error between the interpolated polynomial and true function.