Homework 4

Problem 1

(4+3+3 points)

a) For computing f', consider the backward differencing scheme. Apply the idea of Richardson extrapolation to derive an estimate with quadratic error term in the step-size h. Compute the exact (not just asymptotic) error.

Answer:

Backward differencing:

$$f'(x) = \frac{f(x) - f(x-h)}{h} = \varphi(h)$$

Taylor Series Expansion:

$$\begin{aligned} f(x-h) &= f(x) - h \cdot f' + \frac{h^2}{2} \cdot f''(x) - \frac{h^3}{6} \cdot f'''(\xi), \quad \xi \in (x-h, x) \\ \varphi(h) &= \frac{f(x) - f(x-h)}{h} = f'(x) - \frac{h}{2} f''(x) + \frac{h^2}{6} f'''(\xi) \end{aligned}$$

From the formulation above, we can find the required equations, at first for h:

$$f'(x) = \varphi(h) + \frac{h}{2} f''(x) - \frac{h^2}{6} f'''(\xi), \quad \xi \in (x-h, x) \tag{1}$$

Now, we evaluate this for $\frac{h}{2}$:

$$f'(x) = \varphi\left(\frac{h}{2}\right) + \frac{h}{4} f''(x) - \frac{h^2}{24} f'''(\xi), \quad \xi \in (x-h, x) \tag{2}$$

Now, we apply (1) $-\, 2 \cdot$ (2):

$$\begin{aligned} -f'(x) &= \varphi(h) - 2 \cdot \varphi\left(\frac{h}{2}\right) - \frac{h^2}{12} \cdot f'''(\xi) \\ \Rightarrow f'(x) &= \left(2\varphi\left(\frac{h}{2}\right) - \varphi(h)\right) + \frac{h^2}{12} \cdot f'''(\xi) \\ \Rightarrow f'(x) &= \left(2\varphi\left(\frac{h}{2}\right) - \varphi(h)\right) + \Theta(h^2) \end{aligned}$$

The error evaluation is provided below:

$$\begin{aligned} E(h) &= \frac{h^2}{12} \cdot f'''(\xi) \quad (ExactError) \\ &= \Theta(h^2) \quad (AsymptoticError) \end{aligned}$$

b) Apply Richardson extrapolation to the estimate from (a) in order to derive an estimate with even better asymptotic error. Compute the exact (not just asymptotic) error.

Answer: In order to solve this problem, we first define the following:

$$\varphi_1(h) = 2 \cdot \varphi\left(\frac{h}{2}\right) - \varphi(h), \quad (termfrom(a))$$

$$f'(x) = \varphi_1(h) + \frac{h^2}{12} \cdot f'''(h) - \frac{h^3}{32} \cdot f^{(4)}(\xi), \quad \xi \in (x-h, x) \tag{1}$$

For $\frac{h}{2}$:

$$f'(x) = \varphi_1(h/2) + \frac{h^2}{48} \cdot f'''(h) - \frac{h^3}{256} \cdot f^{(4)}(\xi), \quad \xi \in \left(x - \frac{h}{2}, x\right) \tag{2}$$

Now, we perform $(1) - 4 \cdot (2)$:

$$\begin{aligned}
-3f' &= \left(\varphi_1(h) - 4 \cdot \varphi_1\left(\frac{h}{2}\right)\right) - \frac{h^3}{64} \cdot f^{(4)}(\xi) \\
f'(x) &= \frac{4\varphi_1\left(\frac{h}{2}\right) - 4\varphi_1(h)}{3} + \frac{h^3}{192} \cdot f^4(\xi) \\
f'(x) &= \frac{4\varphi_1\left(\frac{h}{2}\right) - 4\varphi_1(h)}{3} + \Theta(h^3)
\end{aligned}$$

From this, we have the exact error:

$$\begin{aligned}
E(h) &= \frac{h^3}{192} \cdot f^4(\xi) \quad (ExactError) \\
&= \Theta(h^3) \quad (AsymptoticError)
\end{aligned}$$

Here, we see that the asymptotic error is $\Theta(h^3)$, which is better than the error from the previous solution, $\Theta(h^2)$.

c) Apply the estimates from (a) and (b) and the backward differencing scheme itself to compute $f'(\pi)$ for a function $f(x) = \cos(x)$ and step size $h = 0.3$.
Answer:
In summary, we maintain the following:

$$f'(x) = cos(x), \quad h = 0.3, \quad x = \pi$$

First, we apply Backward Differencing:

$$\begin{aligned}
f'(x) &\approx \frac{f(x) - f(x-h)}{h} \\
f'(\pi) &\approx \frac{f(\pi) - f(\pi - 0.3)}{0.3} \\
&= \frac{-1 + 0.9553364}{0.3} \\
&= -0.14888
\end{aligned}$$

Based on our estimate from (a), we have:

$$\begin{aligned}
\varphi(h) &= \frac{f(x) - f(x-h)}{h} \\
f'(x) &\approx 2 \cdot \varphi\left(\frac{h}{2}\right) - \varphi(h) \\
&= 2 \cdot \frac{f(x) - f\left(x - \frac{h}{2}\right)}{\frac{h}{2}} - \frac{f(x) - f(x-h)}{h} \\
&= \frac{3 \cdot f(x) - 4 \cdot f\left(x - \frac{h}{2}\right) + f(x-h)}{h} \\
f'(\pi) &\approx \frac{3 \cdot f(\pi) - 4 \cdot f(\pi - 0.15) + f(\pi - 0.3)}{0.3} \\
f'(\pi) &\approx -8.406 \times 10^{-4}
\end{aligned}$$

Based on our estimate from (b), we have:

$$\begin{aligned}
\varphi(h) &= \frac{f(x) - f(x-h)}{h} \\
\varphi_1(h) &= 2 \cdot \varphi\left(\frac{h}{2}\right) - \varphi(h) \\
f'(x) &\approx \frac{4\varphi_1\left(\frac{h}{2}\right) - 4\varphi_1(h)}{3} \\
f'(x) &\approx \frac{1}{3} \cdot \left(4 \cdot \left(2 \cdot \varphi\left(\frac{h}{4}\right) - \varphi\left(\frac{h}{2}\right)\right)\right) - \left(2 \cdot \varphi\left(\frac{h}{2}\right) - \varphi(h)\right) \\
&= \frac{1}{3} \cdot \left(8 \cdot \varphi\left(\frac{h}{4}\right) - 6 \cdot \varphi\left(\frac{h}{2}\right) + \varphi(h)\right) \\
&= \frac{1}{3} \cdot \left(8 \cdot \frac{f(x) - f\left(x - \frac{h}{4}\right)}{\frac{h}{4}} - 6 \cdot \frac{f(x) - f\left(x - \frac{h}{2}\right)}{\frac{h}{2}} + \frac{f(x) - f(x-h)}{h}\right) \\
&= \frac{1}{3h} \cdot \left(21 \cdot f(x) - 32 \cdot f\left(x - \frac{h}{4}\right) + 12 \cdot f\left(x - \frac{h}{2}\right) - f(x-h)\right) \\
f'(\pi) &\approx \frac{1}{3 \cdot (0.3)} \cdot \left(21 \cdot f(\pi) - 32 \cdot f\left(\pi - \frac{0.3}{4}\right) + 12 \cdot f\left(\pi - \frac{0.3}{2}\right) - f(\pi - 0.3)\right) \\
f'(\pi) &\approx 1.397 \times 10^{-4}
\end{aligned}$$

The actual values are the following:

$$\begin{aligned}
f(x) &= cos(x) \\
f'(x) &= -sin(x) \\
f'(\pi) &= 0
\end{aligned}$$

We can see that the estimates from (a) and (b) are closest to the actual solution, with estimate (b) being the closest.

Problem 2

(3+3+3+3 points) Consider the measurement values $p_0 = 7$, $p_1 = 6$, and $p_2 = 8$ that have been obtained at the nodes $u_0 = 0$, $u_1 = \frac{\pi}{4}$,and $u_2 = \frac{\pi}{2}$. Assume that a function $p(u) = a\cos(u) + bu$ is supposed to approximate these values in the least squares sense.

a) Formulate the normal equations as a linear system of equations.
Answer:

First, we tally the provided data:

i	0	1	2	
p	7	6	8	
u	0	$\frac{\pi}{4}$	$\frac{\pi}{2}$	

The following function is supposed to approximate this dataset in the $L2$ sense:

$$p(u) = a \cdot cos(u) + b \cdot u$$

In order to calculate the $L2$ error, we use the following equation and substitute $p(u)$:

$$E_2(a,b) = \sum_{i=0}^{n}(p_i - p(u_i))^2, \quad n = 2$$
$$\Rightarrow E_2(a,b) = \sum_{i=0}^{n}(p_i - (a \cdot cos(u_i) + b \cdot u_i))^2$$

We need to find a and b such that the approximation error in the $L2$ norm, E_2, is minimized. Hence, a and b are determined by solving $\frac{\partial E}{\partial a} = 0$ and $\frac{\partial E}{\partial b} = 0$, which yields a system of normal equations.

$$\frac{\partial E}{\partial a} = \sum_{i=0}^{n}(2(p_i - a \cdot cos(u_i) + b \cdot u_i)(-cos(u_i)))$$
$$\Rightarrow \frac{\partial E}{\partial a} = \sum_{i=0}^{n}(-2 \cdot p_i \cdot cos(u_i) + 2 \cdot a \cdot cos^2(u_i) + 2 \cdot b \cdot u_i \cdot cos(u_i)) = 0$$
$$\Rightarrow \sum_{i=0}^{n}(-2 \cdot p_i \cdot cos(u_i)) + a \cdot \sum_{i=0}^{n}(2 \cdot cos^2(u_i)) + b \cdot \sum_{i=0}^{n}(2 \cdot u_i \cdot cos(u_i)) = 0$$
$$\Rightarrow a \cdot \sum_{i=0}^{n}(cos^2(u_i)) + b \cdot \sum_{i=0}^{n}(u_i \cdot cos(u_i)) = \sum_{i=0}^{n}(p_i \cdot cos(u_i)) \longrightarrow (1)$$

$$\frac{\partial E}{\partial b} = \sum_{i=0}^{n}(2(p_i - a \cdot cos(u_i) + b \cdot u_i)(-u_i))$$

$$\Rightarrow \sum_{i=0}^{n}(-2 \cdot p_i \cdot u_i + a \cdot 2 \cdot u_i \cdot cos(u_i) + 2 \cdot b \cdot u_i^2) = 0$$

$$\Rightarrow -\sum_{i=0}^{n}(2 \cdot p_i \cdot u_i) + a \cdot \sum_{i=0}^{n}(2 \cdot u_i \cdot cos(u_i)) + b \cdot \sum_{i=0}^{n}(2 \cdot u_i^2) = 0$$

$$\Rightarrow a \cdot \sum_{i=0}^{n}(u_i \cdot cos(u_i)) + b \cdot \sum_{i=0}^{n}(u_i^2) = \sum_{i=0}^{n}(p_i \cdot u_i) \longrightarrow (2)$$

The system of normal equations is provided below:

$$a \cdot \sum_{i=0}^{n}(cos^2(u_i)) + b \cdot \sum_{i=0}^{n}(u_i \cdot cos(u_i)) = \sum_{i=0}^{n}(p_i \cdot cos(u_i)) \tag{3}$$

$$a \cdot \sum_{i=0}^{n}(u_i \cdot cos(u_i)) + b \cdot \sum_{i=0}^{n}(u_i^2) = \sum_{i=0}^{n}(p_i \cdot u_i) \tag{4}$$

b) Solve the normal equations.
Answer:
From equations (1) and (2), we can determine the following matrix:

$$\begin{bmatrix} \sum_{i=0}^{n}(cos^2(u_i)) & \sum_{i=0}^{n}(u_i \cdot cos(u_i)) \\ \sum_{i=0}^{n}(u_i \cdot cos(u_i)) & \sum_{i=0}^{n}(u_i^2) \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{n}(p_i \cdot cos(u_i)) \\ \sum_{i=0}^{n}(p_i \cdot u_i) \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} 1 + 0.5 + 0 & 0 + 0.555 + 0 \\ 0.555 & 0 + 2.467 + 0.616 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 7 + 3\sqrt{2} + 0 \\ 0 + \frac{3\pi}{2} + 4\pi \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} 1.5 & 0.555 \\ 0.555 & 3.083 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 11.243 \\ 17.279 \end{bmatrix}$$

This allows us to solve the system of equations using matrix inverse:

$$\begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{4.316} \cdot \begin{bmatrix} 3.083 & -0.555 \\ -0.555 & 1.5 \end{bmatrix} \cdot \begin{bmatrix} 11.243 \\ 17.279 \end{bmatrix} = \begin{bmatrix} 5.8091 \\ 4.5595 \end{bmatrix}$$

c) Compute the error in the l_2 sense that is minimized in b). What is the solution and what is the error if the last measurement value is $p_2 = 5$? Briefly explain what happens in the latter case.

Answer:

$$\begin{aligned} E(5.8091, 4.5595) &= \sum_{i=0}^{n}(p_i - (5.8091 \cdot cos(u_i) + 4.5595 \cdot u_i))^2, \quad n = 2 \\ &= 1.4182 + 2.8516 + 0.7022 \\ &= 4.972 \end{aligned}$$

Now, we compute the same for $p_2 = 5$.
First, we reevaluate the data table:

i	0	1	2	
p	7	6	5	
u	0	$\frac{\pi}{4}$	$\frac{\pi}{2}$	

We will use the system of normal equations from the last section, so we can start from the following matrix evaluation for the system:

$$\begin{bmatrix} \sum_{i=0}^{n}(cos^2(u_i)) & \sum_{i=0}^{n}(u_i \cdot cos(u_i)) \\ \sum_{i=0}^{n}(u_i \cdot cos(u_i)) & \sum_{i=0}^{n}(u_i^2) \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{n}(p_i \cdot cos(u_i)) \\ \sum_{i=0}^{n}(p_i \cdot u_i) \end{bmatrix}$$

When evaluated, the final system looks as follows:

$$\begin{bmatrix} 1 + 0.5 + 0 & 0 + 0.555 + 0 \\ 0.555 & 0 + 2.467 + 0.616 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 7 + 3\sqrt{2} + 0 \\ 0 + \frac{3\pi}{2} + \frac{5\pi}{2} \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{4.316} \cdot \begin{bmatrix} 3.083 & -0.555 \\ -0.555 & 1.5 \end{bmatrix} \cdot \begin{bmatrix} 11.243 \\ 4\pi \end{bmatrix} = \begin{bmatrix} 6.4152 \\ 2.9216 \end{bmatrix}$$

Using this solution, we evaluate the error again:

$$\begin{aligned} E(6.4152, 2.9216) &= \sum_{i=0}^{n}(p_i - (6.4152 \cdot cos(u_i) + 2.9216 \cdot u_i))^2, \quad n = 2 \\ &= 0.3412 + 0.6903 + 0.1687 \\ &= 1.2002 \end{aligned}$$

The error decreases because $p(u)$ aligns better with the dataset with $p_2 = 5$, due to the symmetry of the cos(u) function.

d) Do the same computations, but for the linear function $p_l(u) = au + b$ and compute the error in the l_2 sense. Which is the better approximation for this data set, comparing d) with b)?

$$E_2(a,b) = \sum_{i=0}^{n}(p_i - p(u_i))^2, \quad n = 2$$

$$\Rightarrow E_2(a,b) = \sum_{i=0}^{n}(p_i - (a \cdot u_i + b))^2$$

Again, we need to minimize the error, so we find the constants that allow us to accomplish this:

$$\frac{\partial E}{\partial a} = \sum_{i=0}^{n}(2(p_i - a \cdot cos(u_i) + b \cdot u_i)(-u_i))$$

$$\Rightarrow \frac{\partial E}{\partial a} = \sum_{i=0}^{n}(-p_i \cdot u_i + a \cdot u_i^2 + b \cdot u_i) = 0$$

$$\Rightarrow a \cdot \sum_{i=0}^{n}(u_i^2) + b \cdot \sum_{i=0}^{n}(u_i) = \sum_{i=0}^{n}(p_i \cdot u_i) \longrightarrow (1)$$

$$\frac{\partial E}{\partial b} = \sum_{i=0}^{n}(2(p_i - a \cdot cos(u_i) + b \cdot u_i)(-1)) = 0$$

$$\Rightarrow a \cdot \sum_{i=0}^{n}(u_i) + b \cdot \sum_{i=0}^{n}(1) = \sum_{i=0}^{n}(p_i) \longrightarrow (2)$$

$$\begin{bmatrix} \sum_{i=0}^{n}(u_i^2) & \sum_{i=0}^{n}(u_i) \\ \sum_{i=0}^{n}(u_i) & \sum_{i=0}^{n}(1) \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{n}(p_i \cdot u_i) \\ \sum_{i=0}^{n}(p_i) \end{bmatrix}$$

Substituting the values provides us the following matrix system:

$$\begin{bmatrix} 3.0842 & 2.356 \\ 2.356 & 3 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 17.279 \\ 21 \end{bmatrix}$$

We now solve for a and b:

$$\begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{3.702} \cdot \begin{bmatrix} 3 & -2.356 \\ -2.356 & 3.0842 \end{bmatrix} \cdot \begin{bmatrix} 17.279 \\ 21 \end{bmatrix} = \begin{bmatrix} 0.6378 \\ 6.499 \end{bmatrix}$$

With this solution, we can now evaluate the error:

$$\begin{aligned} E(0.6378, 6.499) &= \sum_{i=0}^{n} (p_i - (0.6378 \cdot u_i + 6.499))^2, \quad n = 2 \\ &= 0.251 + 0.99985 + 0.249 \\ &= 1.5 \end{aligned}$$

For the dataset provided, $p(u) = a \cdot u_i + b$ is the better approximation, as it has lower error.

Problem 3

(2+1 points) Consider the function $f(x) = e^{x^2+1}$ and the set of equally spaced nodes $0, 0.25, 0.5, 0.75, 1$.

a) Derive the polynomial $p_n(x)$ in Newton form that interpolates $f(x)$ at the given nodes.

Answer:
First, we need to tabulate the data in order to get a clear picture:

i	0	1	2	3	4
x_i	0	0.25	0.5	0.75	1
y_i	2.718	2.894	4.490	4.771	7.389

The polynomial in Newton form can be written as follows:

$p_i(x) = \sum_{i=0}^{n}(C_i \cdot n_i(x))$, where $n_i(x) = (x - x_0)(x - x_1)...(x - x_{i-1})$, and $n_0(x) = 1$

Using this definition, we can start deriving the polynomials based on node levels. The derivation looks as follows:

$$\begin{aligned}
&p_0(x) = C_0 \cdot n_0(x) \\
&\Rightarrow p_0(x_0) = C_0 \cdot n_0(x_0) = y_0 \\
&\Rightarrow p_0(0) = C_0 = 2.718
\end{aligned}$$

$$\begin{aligned}
&p_1(x) = C_0 \cdot n_0(x) + C_1 \cdot n_1(x) \\
&\Rightarrow p_1(x) = C_0 + C_1 \cdot (x - x_0) \\
&\Rightarrow p_1(x_1) = C_0 + C_1 \cdot (x_1 - x_0) = y_1 \\
&\Rightarrow C_1 = \frac{y_1 - C_0}{x_1 - x_0} = \frac{2.894 - 2.718}{0.25 - 0} = 0.704
\end{aligned}$$

$$\begin{aligned}
&p_2(x) = C_0 \cdot n_0(x) + C_1 \cdot n_1(x) + C_2 \cdot n_2(x) \\
&\Rightarrow p_2(x) = C_0 + C_1 \cdot (x - x_0) + C_2 \cdot (x - x_0) \cdot (x - x_1) \\
&\Rightarrow p_2(x_2) = C_0 + C_1 \cdot (x_2 - x_0) + C_2 \cdot (x_2 - x_0) \cdot (x_2 - x_1) = y_2 \\
&\Rightarrow C_2 = \frac{y_2 - C_0 - C_1(x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)} = \frac{3.490 - 2.718 - 0.704(0.5 - 0)}{(0.5 - 0)(0.5 - 0.25)} = 3.36
\end{aligned}$$

$$\begin{aligned}
&p_3(x) = C_0 \cdot n_0(x) + C_1 \cdot n_1(x) + C_2 \cdot n_2(x) + C_3 \cdot n_3(x) \\
&\Rightarrow p_3(x) = C_0 + C_1 \cdot (x - x_0) + C_2 \cdot (x - x_0) \cdot (x - x_1) + C_3 \cdot (x - x_0) \cdot (x - x_1) \cdot (x - x_2) \\
&\Rightarrow p_3(x_3) = C_0 + C_1 \cdot (x_3 - x_0) + C_2 \cdot (x_3 - x_0) \cdot (x_3 - x_1) + C_3 \cdot (x_3 - x_0) \cdot (x_3 - x_1) \cdot (x_3 - x_2) = y_3 \\
&\Rightarrow C_3 = \frac{y_3 - C_0 - C_1 \cdot (x_3 - x_0) - C_2 \cdot (x_3 - x_0)(x_3 - x_1)}{(x_3 - x_0) \cdot (x_3 - x_1) \cdot (x_3 - x_2)} = 2.8267
\end{aligned}$$

$$\begin{aligned}
&p_4(x) = C_0 \cdot n_0(x) + C_1 \cdot n_1(x) + C_2 \cdot n_2(x) + C_3 \cdot n_3(x) + C_4 \cdot n_4(x) \\
&\Rightarrow p_4(x) = C_0 + C_1 \cdot (x - x_0) + C_2 \cdot (x - x_0) \cdot (x - x_1) + C_3 \cdot (x - x_0) \cdot (x - x_1) \cdot (x - x_2) \\
&\qquad\qquad + C_4 \cdot (x - x_0) \cdot (x - x_1) \cdot (x - x_2) \cdot (x - x_3) \\
&\Rightarrow p_4(x_4) = C_0 + C_1 \cdot (x_4 - x_0) + C_2 \cdot (x_4 - x_0) \cdot (x_4 - x_1) + C_3 \cdot (x_4 - x_0) \cdot (x_4 - x_1) \cdot (x_4 - x_2) \\
&\qquad\qquad + C_4 \cdot (x_4 - x_0) \cdot (x_4 - x_1) \cdot (x_4 - x_2) \cdot (x_4 - x_3) = y_4 \\
&\Rightarrow C_4 = \frac{y_4 - C_0 - C_1 \cdot (x_4 - x_0) - C_2 \cdot (x_4 - x_0) \cdot (x_4 - x_1) - C_3 \cdot (x_4 - x_0) \cdot (x_4 - x_1) \cdot (x_4 - x_2)}{(x_4 - x_0) \cdot (x_4 - x_1) \cdot (x_4 - x_2) \cdot (x_4 - x_3)} \\
&\Rightarrow C_4 = 4.1279
\end{aligned}$$

With this, we have everything we require to construct our polynomial. In order to do this, we go back to the definition:

$p_i(x) = \sum_{i=0}^{n}(C_i \cdot n_i(x))$, where $n_i(x) = (x-x_0)(x-x_1)...(x-x_{i-1})$, and $n_0(x) = 1$

$$\Rightarrow p_4(x) = C_0 \cdot n_0(x) + C_1 \cdot n_1(x) + C_2 \cdot n_2(x) + C_3 \cdot n_3(x) + C_4 \cdot n_4(x)$$
$$\Rightarrow p_4(x) = C_0 + C_1 \cdot (x-x_0) + C_2 \cdot (x-x_0) \cdot (x-x_1) + C_3 \cdot (x-x_0) \cdot (x-x_1) \cdot (x-x_2) + C_4 \cdot (x-x_0) \cdot (x-x_1) \cdot (x-x_2) \cdot (x-x_3)$$

On substituting all the values on this equation and simplifying, we reach the following final solution:

$$p_4(x) = 2.718 - 0.16969x + 4.07799x^2 - 3.3653x^3 + 4.1279x^4$$

b) Calculate the interpolation error $\max_{x\in[0,1]}|f(x) - p_n(x)|$. Note: Use numerical means/illustrations.
Answer:
Let,

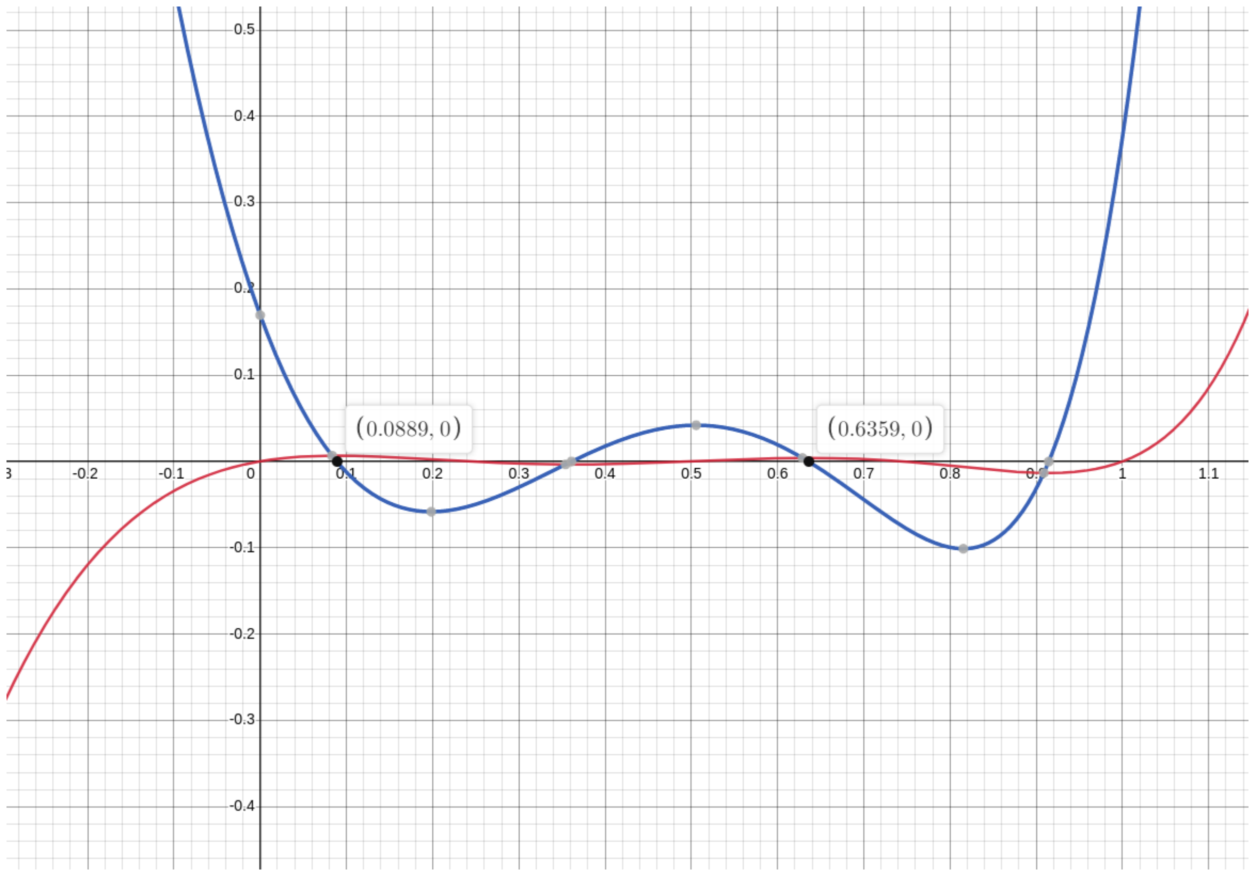
$$g(x) = f(x) - p_4(x)$$
$$\Rightarrow max_{x\in[0,1]}|g(x)| = max_{x\in[0,1]}|f(x) - p_4(x)|$$

Now, we evaluate $g(x)$ and its first and second derivatives:

$$g(x) = e^{x^2+1} - 2.718 + 0.16969x - 4.07799x^2 + 3.3653x^3 - 4.1279x^4$$
$$\frac{dg(x)}{dx} = (2x)e^{x^2+1} + 0.16969 - 8.15598x + 10.0959x^2 - 16.5116x^3$$
$$\frac{d^2g(x)}{dx^2} = 2(2x^2+1)e^{x^2+1} - 8.15598 + 20.1918x - 49.5348x^2$$

As we can see, an analytical solution is not possible for this case. Hence, we need to resort to a numerical solution. Our target is to maximize $g(x)$, which means we need to find the roots of its derivative, since those are the critical points of $g(x)$, and then verify which are the maxima and which are the minima, which would be possible using the second derivative.

However, since the construction of $g(x)$ is by default quite complex, we can study the function some more in order to reduce the number of steps we require to reach a solution. For example, let us study the graph of $g(x)$ and its derivative:



On this graph, the red line is $g(x)$ and the blue line is its derivative.
We can clearly see that $g(x)$ contains 4 critical points and all of them are inside the provided range between 0 and 1, and out of those, two are maxima. The same can be deduced from looking at the roots of $\frac{dg(x)}{dx}$, which exist exactly at the critical points of $g(x)$.

On the graph, I have marked exactly the values of x that we require in order to provide some reference to what we are looking for. We already have the values we need, but it's still a good idea to explore the numerical solution since this type of ideal situation is not always likely. Most times, we'll be able to sketch the graphs and estimate bounds or starting points based on it, but won't be able to pinpoint exact values.

Since there are multiple critical points on the provided range and we need to pursue a numerical route towards a solution, one option is to sample multiple points within this range and try to converge to each critical point. However, since we have the curve as a reference, we can reduce the number of steps required by just noticing that since the maxima are bounded by $0 < x < 0.2$ and $0.6 < x < 0.7$, and the maxima are exactly what we want, we can attempt at convergence to roots of $\frac{dg(x)}{dx}$ within these bounds. The secant method is the best algorithm in order to accomplish this.

The equation for the secant method is provided below

$$x_n = x_{n-1} - f(x_{n-1}) \cdot \frac{x_{n-1} - x_{n-2}}{f(x_{n-1}) - f(x_{n-2})}$$

Even with our work somewhat simplified, executing the secant method by hand on this problem could get quite strenuous, which is why I have prepared a simple script to execute the iterations:

```python
import math

# We need to find maximum of this g(x)
def g(x):
    return math.exp(x**2 + 1) - 2.718 + 0.16969*x
                - 4.07799*x**2 + 3.3653*x**3 - 4.1279*x**4

# To find maximum of g(x), we need to find roots of gprime(x)
def gprime(x):
    return math.exp(x**2 + 1)*2*x + 0.16969 - 8.15598*x
                + 10.0959*x**2 - 16.5116*x**3

# To find roots and confirm if it's a maxima, we need gdoubleprime(x)
# g(x) is at maximum if gdoubleprime(x) is negative
def gdoubleprime(x):
    return 2*(2*x + 1)*math.exp(x**2 * 1) - 8.15598
                                + 20.1918*x - 49.5348*x**2

def nm_step(x, step_num):
    val = x - gprime(x)/gdoubleprime(x)
    step_num += 1
    print(f"Step Num: {step_num}, x: {val}")
    return val, step_num

def secant_step(x1, x2, step_num, f):
    step_num += 1
    val = x2 - f(x2)*(x2 - x1)/(f(x2) - f(x1))
    print(f"Step Num: {step_num}, x: {val}")
    return x2, val, step_num

# Execution:
# We use secant method here because we can bound it.
def calculate_first_max():
    step_num = 0
    x1 = 0      # These bounds are found by analyzing the graph
    x2 = 0.2
    for i in range(19):
        x1, x2, step_num = secant_step(x1, x2, step_num, gprime)
    return x2

def calculate_second_max():
    step_num = 0
    x1 = 0.6      # These bounds are found by analyzing the graph
    x2 = 0.7
    for i in range(6):
        x1, x2, step_num = secant_step(x1, x2, step_num, gprime)
    return x2

if __name__== "__main__":
    print("Calculating x for first maximum in x -> [0, 1], in section
    [0, 0.2]:")
    max_x_1 = calculate_first_max()
    print("Calculating x for second maximum in x -> [0, 1], in section
    [0.6, 0.7]:")
    max_x_2 = calculate_second_max()

    print("First maximum value: ", g(max_x_1))
    print("Second maximum value: ", g(max_x_2))

```

The results of the iterations are provided below:

```
Calculating x for first maximum in x -> [0, 1], in section [0, 0.2]:
Step Num: 1, x: 0.14900380590331655
Step Num: 2, x: -0.08821605866212776
Step Num: 3, x: 0.12814220849590097
Step Num: 4, x: 0.11341387156881519
Step Num: 5, x: 0.0806834268432286
Step Num: 6, x: 0.09017014204250429
Step Num: 7, x: 0.08892692738129448
Step Num: 8, x: 0.08886390723627205
Step Num: 9, x: 0.08886440328278541
Step Num: 10, x: 0.08886440309559314
Step Num: 11, x: 0.08886440309559253
Step Num: 12, x: 0.0888644030955926
Step Num: 13, x: 0.08886440309559303
Step Num: 14, x: 0.08886440309559265
Step Num: 15, x: 0.08886440309559258
Step Num: 16, x: 0.08886440309559261
Step Num: 17, x: 0.0888644030955926
Step Num: 18, x: 0.08886440309559261
Step Num: 19, x: 0.08886440309559261
Calculating x for second maximum in x -> [0, 1], in section [0.6, 0.7]:
Step Num: 1, x: 0.6307415305490232
Step Num: 2, x: 0.6353034667646523
Step Num: 3, x: 0.6358951723781623
Step Num: 4, x: 0.6358866382102938
Step Num: 5, x: 0.6358866518459403
Step Num: 6, x: 0.6358866518462564
First maximum value:  0.006812940563426053
Second maximum value:  0.004225969342977254
```

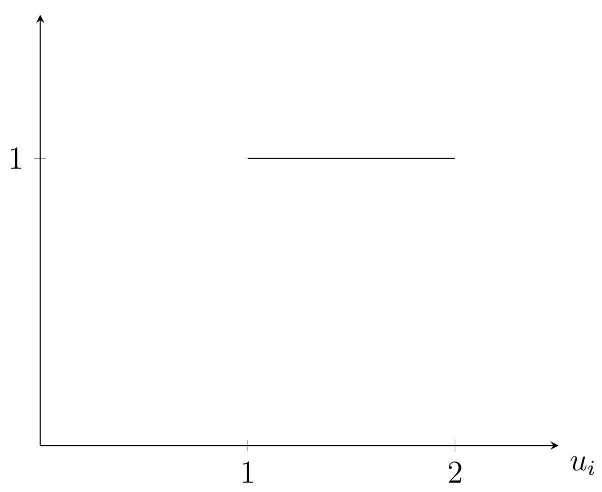
From the results, we can see that $\frac{dg(x)}{dx}$ has roots at $x \approx 0.0889$ and $x \approx 0.636$, which are the same as what we found in the graph. Evaluating $g(x)$ using these values gives us the maxima $g(0.0889) \approx 0.00681$ and $g(0.636) \approx 0.004226$. For our error, we choose the highest between the two, which is $g(0.0889) \approx 0.00681$.

Problem 4

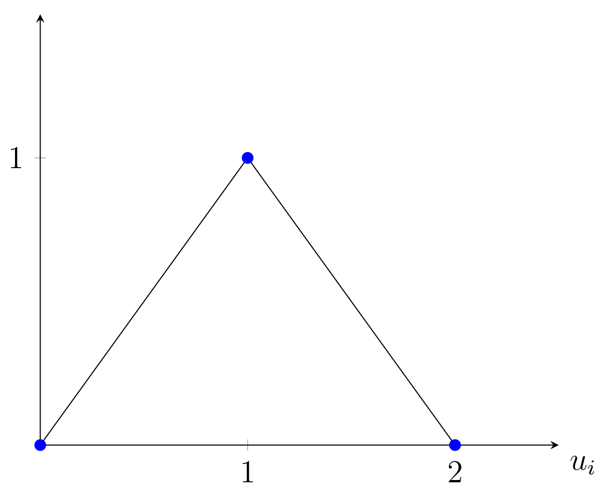
(2 + 3 points) Consider the B-splines over the nodes $u_i \in \{0, 1, 2\}$, $i = 0, 1, 2$

a) Sketch the B-splines $N_1^0(u)$ and $N_1^1(u)$.

N_1^0



N_1^1



b) Use the recursive formulation of splines
$N_i^0(u) = 1$ if $u \in [u_i, u_{i+1})$ and 0 else,
and
$N_i^n(u) = \alpha_i^{n-1}(u) N_i^{n-1}(u) + (1 - \alpha_{i+1}^{n-1}(u)) N_{i+1}^{n-1}(u)$ with $\alpha_i^{n-1}(u) = \dfrac{u - u_i}{u_{i+n} - u_i}$
to compute the necessary splines for the linear spline interpolation with $n = 1$. Compute the collocation matrix and solve the system of linear equations $s(u_i) = \sum_i c_i N_i^1(u_i)$ for the values $p_i \in \{5, -2, 2\}$. Where necessary, use for the external nodes $u_{-1} = -1$ and $u_3 = 3$ to construct the splines. Finally, interpolate at $s(1.5)$. Remark: For the collocation matrix it is, in general, not necessary to construct the spines first, as you only need to evaluate them at the nodes u_i; remember the property of minimal support.

<u>**Answer:**</u>

The coallocation matrix is:

$$\Phi = \begin{bmatrix} N_0^1(u_0) & N_1^1(u_0) & N_2^1(u_0) \\ N_0^1(u_1) & N_1^1(u_1) & N_2^1(u_1) \\ N_0^1(u_2) & N_1^1(u_2) & N_2^1(u_2) \end{bmatrix}$$

We are given $n = 1$. So:

$$N_{i+1}^1(u) = \alpha_i^0(u) N_i^0(u) + (1 - \alpha_{i+1}^0(u)) N_{i+1}^0(u) \tag{5}$$

$$\alpha_i^0 = \frac{u - u_i}{u_{i+1} - u_i} \tag{6}$$

First:

$$N_0^0(u) = \begin{cases} 1 & u \in [u_0, u_1) \\ 0 & \text{else} \end{cases}, \quad N_1^0(u) = \begin{cases} 1 & u \in [u_1, u_2) \\ 0 & \text{else} \end{cases}, \quad N_2^0(u) = \begin{cases} 1 & u \in [u_2, u_3) \\ 0 & \text{else} \end{cases}$$

For $i = 0$:

$$\alpha^0_{-1} = u + 1 \qquad \alpha^0_0 = u \qquad N^1_0(u) = (u+1) \cdot N^0_{-1}(u) + (1-u) \cdot N^0_0(u)$$

$$\begin{aligned} N^1_0(u_0) &= (0+1) \cdot 0 + (1-0) \cdot 1 = 1 \\ N^1_0(u_1) &= (1+1) \cdot 0 + (1-1) \cdot 0 = 0 \\ N^1_0(u_2) &= (2+1) \cdot 0 + (1-2) \cdot 0 = 0 \end{aligned}$$

For $i = 1$:

$$\alpha^0_0 = u \qquad \alpha^0_1 = u - 1 \qquad N^1_1(u) = u \cdot N^0_0(u) + (2-u)N^0_1(u)$$

$$\begin{aligned} N^1_1(u_0) &= 0 \cdot 1 + (2-0) \cdot 0 = 0 \\ N^1_1(u_1) &= 1 \cdot 0 + (2-1) \cdot 1 = 1 \\ N^1_1(u_2) &= 2 \cdot 0 + (2-2) \cdot 0 = 0 \end{aligned}$$

For $i = 2$:

$$\alpha^0_1 = u - 1 \qquad \alpha^0_2 = u - 2 \qquad N^1_2(u) = (u-1) \cdot N^0_1(u) + (3-u)N^0_2(u)$$

$$\begin{aligned} N^1_2(u_0) &= (0-1) \cdot 0 + (3-0) \cdot 0 = 0 \\ N^1_2(u_1) &= (1-1) \cdot 1 + (3-1) \cdot 0 = 0 \\ N^1_2(u_2) &= (2-1) \cdot 0 + (3-2) \cdot 1 = 1 \end{aligned}$$

Plugging in the values into the co-allocation matrix:

$$\Phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Finding cofficient:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_0 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 5 \\ -2 \\ 2 \end{bmatrix}$$

So, $\lambda_2 = 2,\ \lambda_1 = -2,\ \lambda_0 = 5$

$$\begin{aligned}
s(u) &= 5N_0^1(u) - 2N_1^1(u) + 2N_2^1(u)\\
s(1.5) &= 5N_0^1(1.5) - 2N_1^1(1.5) + 2N_2^1(1.5)\\
s(1.5) &= 5(1.5+1)\cdot 0 + (1-1.5)\cdot 0 - 2N_1^1(1.5) + 2N_2^1(1.5)\\
s(1.5) &= -2(1.5\cdot 0 + (2-1.5)\cdot 1) + 2N_2^1(1.5)\\
s(1.5) &= -2(0.5) + 2((1.5-1)\cdot 1 + (3-1.5)\cdot 0)\\
s(1.5) &= \mathbf{0}
\end{aligned}$$