# Assignment 3

Lagrange Interpolation Function

Sahil Raj
*2301CS41*

31 August 2025

# Contents

# Problem 1: Interpolation of Function with Several Orders of Lagrange Interpolation Function

## Problem Statement

The objective of this problem is to construct and analyze interpolating polynomials using the Lagrange interpolation method. Given a set of data points $(x_i, f(x_i))$, we approximate the function $f(x)$ by polynomials of varying order and compare their behavior. The interpolating polynomials are also used to estimate $f(7)$.

**NOTE**: The code can be accessed using this link: MATLAB, Julia.

## Methodology

The Lagrange interpolation method constructs a polynomial $P_k(x)$ of degree $k$ that passes through $k+1$ data points. It is expressed as:

$$P_k(x) = \sum_{i=0}^{k} f(x_i)\, L_i(x),$$

where the $i^{th}$ Lagrange basis polynomial is given by:

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{k} \frac{x - x_j}{x_i - x_j}.$$

This ensures $L_i(x_j) = \delta_{ij}$, so that $P_k(x_j) = f(x_j)$.

### Pseudo-code

1. Store the given data points $(X, Y)$.
2. Define a function to compute the Lagrange basis $L_i(x)$ for a given order $k$.
3. Define the interpolating polynomial function $P_k(x)$ using the basis functions and data points.
4. For each order $k = 1, 2, 3$:
   - Plot the interpolating polynomial along with the data points.
   - Evaluate $P_k(7)$ and print the result.

# Results

The given dataset is:

$$(x, f(x)) = \{(0.5,\ 1.625), (1.5,\ 5.875), (3.0,\ 31.0),\\ (5.0,\ 131.0), (6.5,\ 282.125), (8.0,\ 521.0)\}.$$

Interpolating polynomials of order $1, 2$, and $3$ were constructed. The figure below shows the data points and the interpolated curves.

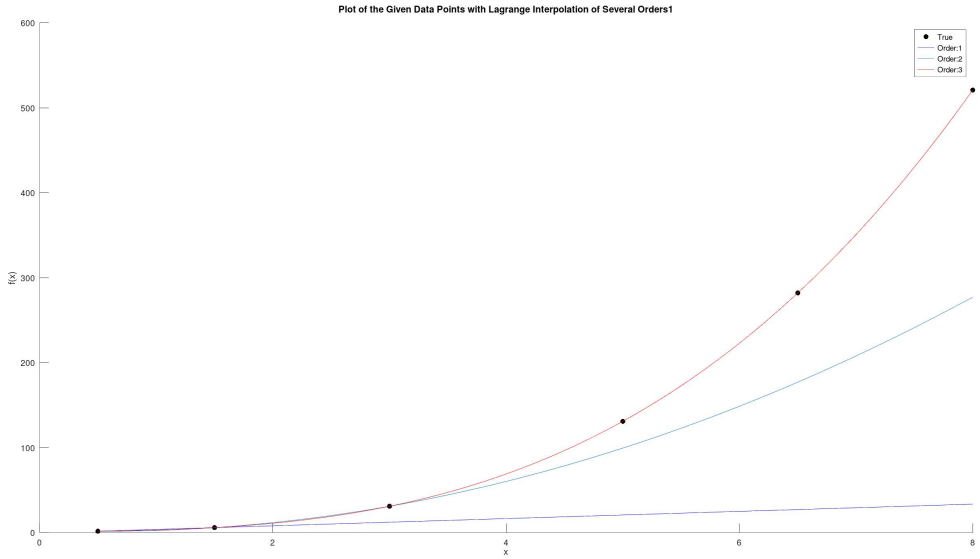


Figure 1.1: Interpolating Functions for Different Orders of Interpolation

The computed estimates of $f(7)$ are:

$$P_1(7) = 29.250000$$
$$P_2(7) = 208.000000$$
$$P_3(7) = 351.000000$$

# Conclusion

The Lagrange interpolation method was successfully applied to approximate the function given discrete data points. Lower-order polynomials (e.g., $k = 1$) provide rough estimates and do not capture the curvature well, while higher-order polynomials fit the data more accurately. The evaluation of $f(7)$ demonstrates how increasing the order improves approximation quality. However, very high-order polynomials may suffer from oscillations (Runge's phenomenon), so the choice of order must balance accuracy and stability.

# Problem 2: Interpolation of Position Data of Earth around Sun

## Problem Statement

The objective of this problem is to interpolate the position of the Earth around the Sun on a weekly basis, given the observed coordinates on the first week of each month. The goal is to use Lagrange interpolation to estimate the Earth's position throughout the year and visualize the interpolated orbit.

**NOTE**: The code can be accessed using this link: MATLAB, Julia.

## Methodology

The Earth's orbit is approximately circular, but only monthly observations (12 data points) are available. To estimate weekly positions (48 weeks), an interpolating polynomial of degree 11 was constructed separately for the $x$- and $y$-coordinates.

The Lagrange interpolation formula is:

$$P_k(x) = \sum_{i=0}^{k} f(x_i)\, L_i(x),$$

where the basis polynomial $L_i(x)$ is defined as:

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{k} \frac{x - x_j}{x_i - x_j}.$$

Here: - $x_i$ corresponds to the week number of the observation, - $f(x_i)$ corresponds to the observed coordinate ($X$ or $Y$).

Two independent interpolations were performed:

$$P_{11}^{(x)}(\text{week}), \quad P_{11}^{(y)}(\text{week}),$$

giving the parametric coordinates of Earth's estimated orbit.

### Pseudo-code

1. Store observed coordinates $(X, Y)$ for each month, along with corresponding week indices.

2. Define Lagrange basis function $L_i(x)$.
3. Define interpolating polynomial $P_k(x)$.
4. For each week $w = 1, \ldots, 48$:
   - Compute interpolated $x = P_{11}^{(x)}(w)$.
   - Compute interpolated $y = P_{11}^{(y)}(w)$.

# Results

The original monthly observations (12 points) are:

$$(x, y) = \{(0.97, 0.25), (0.80, 0.60), (0.55, 0.83), \ldots, (-0.31, -0.95)\}.$$

Using degree-11 Lagrange interpolation, the Earth's position was estimated for all 48 weeks. The figures show:

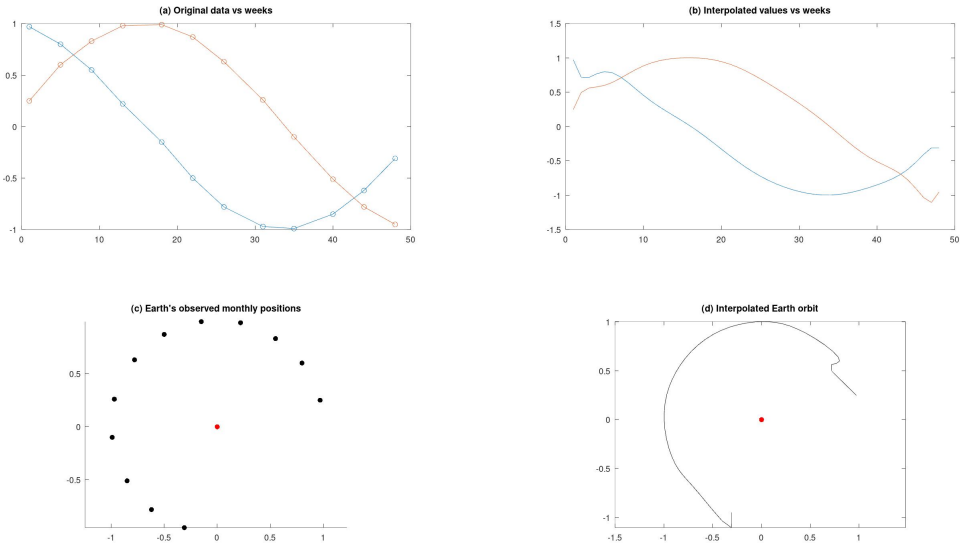


Figure 2.1: (a) Original monthly points forming a near-circular orbit. (b) Time evolution of $x$ and $y$ coordinates across 48 weeks. (c) Observed position data points around the sun (d) Interpolated orbit tracing a closed loop around the Sun.

However, oscillations were observed at the edges of the interpolation domain (start and end weeks). As per my findings, this may be related to **Runge phenomenon**, where high-order polynomial interpolation over equally spaced points produces oscillations near the boundaries. *However, any hints on this from the instructor would be invaluable.*

# Conclusion

The interpolated orbit successfully reconstructs the Earth's approximate trajectory using only monthly data points. The Lagrange interpolation method provides a smooth approximation across weeks, but the observed edge oscillations highlight the limitations of high-order interpolation (Runge phenomenon).

# Problem 3: Error Function for Interpolation of $e^x$ from Values at Given Points

## Problem Statement

The objective of this problem is to analyze the error in approximating the exponential function $f(x) = e^x$ using Lagrange interpolation. Specifically, we evaluate and plot the theoretical error bound over a given set of interpolation nodes.

**NOTE**: The code can be accessed using this link: MATLAB, Julia.

## Methodology

The interpolation error for a function $f(x)$ using $n$ nodes is given by:

$$R_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{i=0}^{n} (x - x_i), \quad \xi \in [\min(X), \max(X)].$$

For $f(x) = e^x$, all derivatives are $f^{(k)}(x) = e^x$. Therefore, an upper bound for the error can be expressed as:

$$|R_n(x)| \leq \frac{e^{\max(X)}}{(n+1)!} \prod_{i=0}^{n} |x - x_i|.$$

Steps:

1. Define interpolation nodes $X = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$.
2. Implement a function to compute the error bound $err\_bound(X, x)$.
3. Sample points $x \in [0.1, 0.6]$ with step 0.01.
4. For each sampled point, compute the error bound.
5. Plot the error bound as a function of $x$.

## Results

The chosen interpolation nodes are:

$$X = \{0.1,\ 0.2,\ 0.3,\ 0.4,\ 0.5,\ 0.6\}.$$

The theoretical error bound was evaluated for points in the interval $[0.1, 0.6]$. The resulting plot shows how the error grows as $x$ moves away from the interpolation nodes, with the error minimized near the nodes themselves.

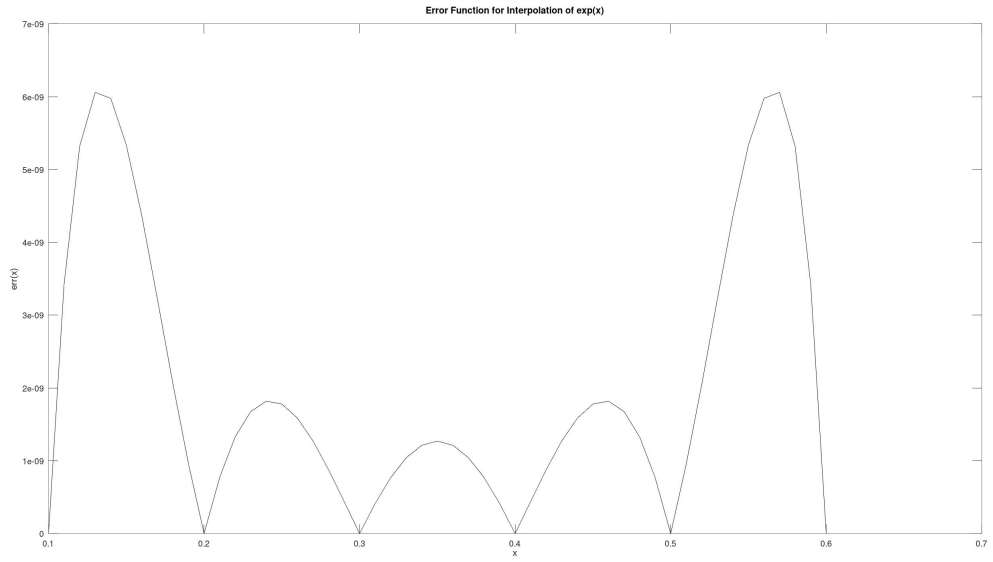


Figure 3.1: Error Bound Plot for values of x

# Conclusion

The error bound for Lagrange interpolation of $e^x$ was computed and plotted. The bound illustrates two key properties:

- The error vanishes at interpolation nodes $x_i$ since the product term contains $(x - x_i)$.
- The error increases between nodes, with larger deviation near the ends of the interval.

This confirms the theoretical behavior of interpolation errors and highlights that interpolation accuracy depends strongly on both the function's smoothness and the placement of interpolation nodes.