Convergence of the Interpolation Process for Selected Mathematical Functions

Zuzanna Sieńko

November 2023

1 Problem Description

In this project, we focus on examining the convergence of the interpolation process for selected mathematical functions. The main goal is to understand how various aspects of interpolation, such as the choice of interpolation nodes or the applied method, affect the accuracy and efficiency of interpolation. Particular attention is given to the analysis of interpolation errors, investigating whether there are strategies to minimize these errors, and how different approaches to the selection of interpolation nodes can affect the final outcome.

The interpolation will be conducted using Newton's method, which is a popular and widely used technique in numerical analysis. We will examine its effectiveness in various scenarios and for different types of mathematical functions.

2 Polynomial Interpolation

Assume we have a function f, which is unknown to us (we do not know its explicit formula) and we only know its values at a finite number of points. Our task is to find an approximating function p_n, which effectively replicates the characteristics and behavior of function f. In our task, p_n will be assumed to be a polynomial function.

Let's assume that:

- $x_i \in \mathbb{R}$, where $i = 0, 1, \ldots, n$ - interpolation nodes,
- $f_i = f(x_i)$, where f is the interpolated function.

Our goal is to find the interpolating function p_n, i.e., a function satisfying the condition:

$$p(x_i) = f(x_i), \quad \text{where} \quad i = 0, 1, \ldots, n.$$

2.1 Existence and Uniqueness Theorem for Interpolating Polynomial

If we have a set of interpolation nodes $x_0, x_1, \ldots, x_n$, which are pairwise different ($x_i \neq x_j$ for $i \neq j$, $i, j = 0, 1, \ldots, n$), and we know the corresponding function values $f_i = f(x_i)$, then there exists exactly one polynomial $p_n \in \Pi_n$ of degree no higher than n, such that it satisfies the condition:

$$p_n(x_i) = f(x_i), \quad \text{for} \quad i = 0, 1, \ldots, n.$$

2.2 Interpolation Using Newton's Method

Newton's interpolation method is based on creating an interpolating polynomial, which is based on Newton's formula. In this method, the interpolating polynomial $p_n(x)$ is presented as follows:

$$p_n(x) = c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \ldots + c_n(x - x_0)(x - x_1)\ldots(x - x_{n-1}).$$

In the above formula, the polynomial p_n is constructed based on the functions 1, $(x - x_0)$, $(x - x_0)(x - x_1)$, and so on, up to $(x - x_0)(x - x_1)\ldots(x - x_{n-1})$.

The key coefficients $c_1, \ldots, c_n$ are determined through the following calculations:

$$\begin{aligned} c_1 &= f_{0,1}, \\ c_2 &= f_{0,1,2}, \\ &\vdots \\ c_n &= f_{0,1,\ldots,n}, \end{aligned}$$

where the so-called divided differences $f_{0,1,\ldots,k}$ for $k = 1, \ldots, n$ are used, calculated based on formulas:

For the first-order divided differences, calculated based on two adjacent nodes x_j and x_{j+1} for $j = 0, \ldots, n-1$, we use:

$$f_{j,j+1} = \frac{f(x_{j+1}) - f(x_j)}{x_{j+1} - x_j}.$$

For the second-order divided differences, based on first-order differences, for $j = 0, \ldots, n-2$, we use:

$$f_{j,j+1,j+2} = \frac{f_{j+1,j+2} - f_{j,j+1}}{x_{j+2} - x_j}.$$

Similarly, the divided differences of higher orders are calculated iteratively, for each j from 0 to $n - k$, where k is the degree of the difference:

$$f_{j,j+1,\ldots,j+k} = \frac{f_{j+1,\ldots,j+k} - f_{j,\ldots,j+k-1}}{x_{j+k} - x_j}.$$

x_i	$f(x_i)$	f_{01}	f_{012}	f_{0123}	f_{01234}
x_0	$f(x_0)$				
x_1	$f(x_1)$	f_{01}			
x_2	$f(x_2)$	f_{12}	f_{012}		
x_3	$f(x_3)$	f_{23}	f_{123}	f_{0123}	
x_4	$f(x_4)$	f_{34}	f_{234}	f_{1234}	f_{01234}

2.3 Interpolation Error

Interpolation is the process of approximating a function, often unknown or without an explicit formula, using a polynomial. Understanding the error of this approximation is crucial.

Theorem Assume that f is a continuous function on the interval $[a, b]$, and p_n is an interpolating polynomial of degree at most n for f, based on distinct nodes $x_0, \ldots, x_n$ belonging to $[a, b]$. For every $x \in [a, b]$, there exists $\xi \in (a, b)$, for which

$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)(x - x_1)\ldots(x - x_n).$$

This error can be estimated as:

$$|f(x) - p_n(x)| \leq M_{n+1}\frac{|(x - x_0)(x - x_1)\ldots(x - x_n)|}{(n+1)!},$$

where $M_{n+1} = \max_{x\in[a,b]} |f^{(n+1)}(x)|$.

In situations where we interpolate data, whose exact mathematical representation is unknown – for example, in the case of measurement data – it is not possible to directly determine or estimate $f^{(n+1)}$. Nevertheless, one can optimize the interpolation process by carefully choosing interpolation nodes.

Key to increasing interpolation accuracy is the appropriate selection of points $x_0, \ldots, x_n$, which allows minimizing the maximum value of the expression:

$$\max_{x\in[a,b]} |(x - x_0)(x - x_1)\ldots(x - x_n)|.$$

Estimating the derivative $f^{(n+1)}(x)$ in practice can be done by expanding the function $y(x+h)$ into a Taylor series and omitting higher orders, leading to a simplified approximation formula. Thus, while the exact estimation of $f^{(n+1)}(x)$ is difficult, an approximate value that is sufficient for interpolation purposes can be obtained.

3 Examining the Convergence of the Interpolation Process

Convergence in the context of interpolation is a complex process that depends on many factors, including the number and distribution of interpolation nodes, the degree of the interpolating polynomial, the characteristics of the interpolated function, and the impact of numerical errors. Examining convergence helps understand how these different aspects interact with each other and affect the accuracy of the polynomial approximation of the function.

During the convergence study, we focus on several key aspects:

- How the interpolation error changes as the number of interpolation nodes increases.
- The impact of the choice of interpolation nodes (evenly spaced, random, Chebyshev nodes) on the convergence of interpolation, but also the impact of specific sets of nodes, such as Chebyshev nodes, on minimizing the Runge effect.
- The dependence on the degree of the interpolating polynomial, how increasing the degree affects the accuracy of interpolation, especially in the context of evenly spaced nodes.
- The analysis of the type of function being interpolated, how different types of functions (smooth, with singularities, irregular) affect the efficiency of interpolation.
- Assessing the impact of numerical errors on the results of interpolation, how rounding errors and calculation precision affect convergence and accuracy of interpolation.

4 Program Description

As part of this project, a series of functions have been implemented in the MATLAB environment for analyzing the convergence of the interpolation function process. The following methods have been developed:

- **coefsNewtonsPolynomial(x,fi)** - a function that calculates the coefficients of the interpolating polynomial using Newton's method, utilizing divided differences. It accepts x - a horizontal vector of interpolation nodes, where the nodes must be pairwise different, and fi - a horizontal vector of the values of the function f at the given nodes.
- **coefsNewtonsPolynomialFunction(x,f)** - a similar function to the above, but accepting the interpolated function f as an argument.
- **generateNewtonsPolynomial(coefs,nodes)** - a function generating the interpolating polynomial as a function. It uses the helper function **evaluatePolynomial(x, coefs, nodes)**, which determines the form of the interpolating polynomial according to Newton's method. It accepts *coefs* - the coefficients of the interpolating polynomial and *nodes* - the interpolation nodes.
- **generateNumericErrorMatrix(p,x,f)** - a function creating a matrix of numerical errors. It compares errors with the built-in method *interp1* and presents the results in tabular form. It accepts p - the interpolating function, x - the interpolation nodes, and f - the interpolated function.
- **calculateMaxInterpolationError(f, p, a, b)** - a function calculating the interpolation error. It generates an error function as $-|p(x) - f(x)|$ and looks for its extremum, i.e., where the interpolation error is the greatest. It accepts f - the interpolated function, p - the interpolating function, and a,b - the boundaries of the interval where the error estimate is sought. The formula from the paragraph on interpolation error was not used, as it is difficult to implement and rarely used in practice.
- **drawNewtonsPolynomial(p,f,x)** - a function creating a graph showing the interpolated function, the interpolating function, and the interpolation nodes. It accepts p - the interpolating function, f - the interpolated function, and x - the interpolation nodes.
- **chebyshevRoots(n)** - a function returning a vector of n roots of the Chebyshev polynomial.

5 Examples

In this section, we present several examples of functions and their properties in the context of polynomial interpolation.

5.1 Runge's Function

We consider the Runge function defined as:

$$f(x) = \frac{1}{1 + 12x^2}. \tag{1}$$

This is a specific case of the general Runge function, which is formulated as:

$$f(x) = \frac{1}{1 + ax^2}, \tag{2}$$

where a is a parameter.

The function $f(x) = \frac{1}{1+x^2}$ is well-known in numerical analysis due to its properties in polynomial interpolation, especially when using uniformly distributed interpolation nodes, which leads to large errors at the ends of the interpolation interval. This effect is known as Runge's phenomenon.

In the case of Runge's function, the issue is that a high-degree interpolating polynomial begins to exhibit strong oscillations near the ends of the interpolation interval. These oscillations result from the accumulation of numerical errors related to the precision of calculations, particularly in situations where a large number of interpolation nodes are used. These errors intensify with the increase in the polynomial's degree, leading to a situation where the interpolating polynomial significantly deviates from the interpolated function, despite good approximation in other areas.

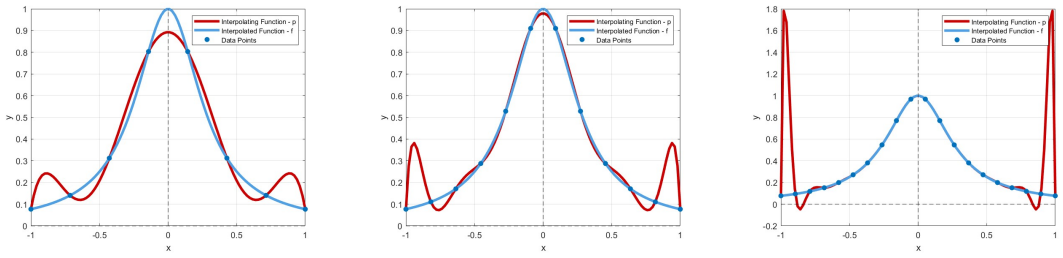

Figure 1: Graphs of the function f on the interval $[-1, 1]$ for a different number of uniformly distributed interpolation nodes. The number of nodes and their corresponding maximum interpolation errors are as follows: 8 nodes - error 0.1476, 12 nodes - error 0.2970, 20 nodes - error 1.7817.

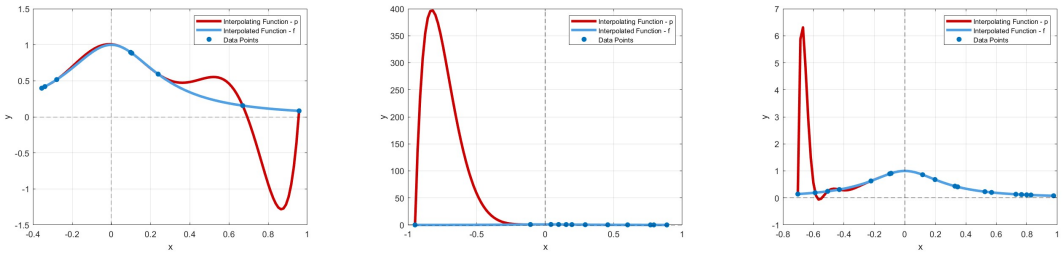

Figure 2: Graphs of the function f on the interval $[-1, 1]$ for a different number of non-uniformly distributed interpolation nodes. The number of nodes and their corresponding maximum interpolation errors are as follows: 8 nodes - error 1.3827, 12 nodes - error 397.4487, 20 nodes - error 6.3557.

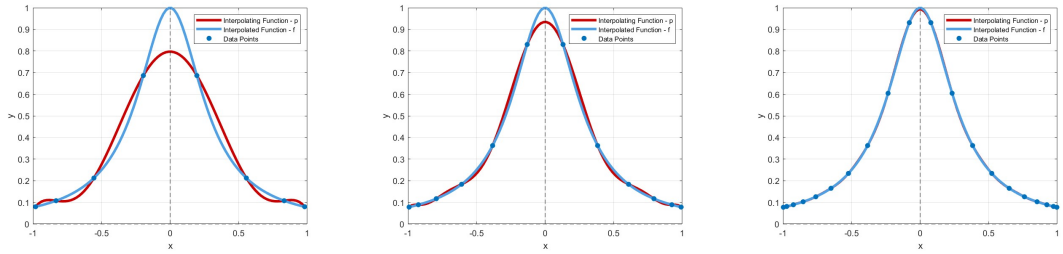

Figure 3: Graphs of the function f on the interval $[-1, 1]$ for a different number of non-uniformly distributed interpolation nodes, which are roots of the Chebyshev polynomial. The number of nodes and their corresponding maximum interpolation errors are as follows: 8 nodes - error 0.2028, 12 nodes - error 0.0655, 20 nodes - error 0.0067.

Observations indicate that both the number and distribution of interpolation nodes significantly impact the convergence of the interpolation process. For example, for uniformly distributed nodes (see Figure 1), the interpolation error significantly increases with the number of nodes. In contrast, in the case of randomly distributed non-uniform nodes (see Figure 2), we observe a high interpolation error regardless of their number, with an exceptionally large error reaching up to 400 at 12 nodes. To minimize the Runge effect and improve convergence, using nodes that are roots of the Chebyshev polynomial proves to be effective, demonstrating a significant acceleration in the convergence of the interpolation process.

5.2 Heaviside Function

Another example is the function whose specific behavior can affect the convergence of the interpolation process. The Heaviside function, often called the step function, is a fundamental tool in control theory and mathematical analysis. It is defined as follows:

$$H(x) = \begin{cases} 0 & \text{for } x < 0, \\ 1 & \text{for } x \geq 0. \end{cases} \tag{3}$$

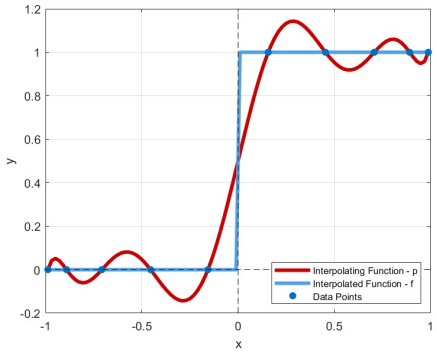

Figure 4: Graph of the function $H(x)$ for 10 Chebyshev nodes. Interpolation error: 0.4996.

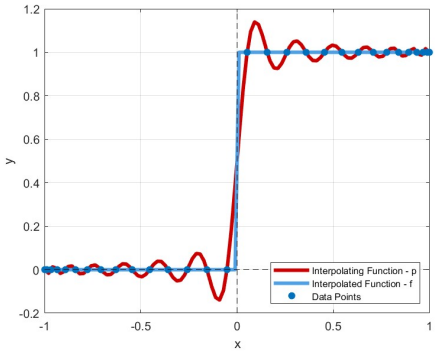

Figure 5: Graph of the function $H(x)$ for 30 Chebyshev nodes. Interpolation error: 0.4989.

We notice that in the case of selected interpolation nodes, which are roots of the Chebyshev polynomial, the characteristic step present in the graph of the Heaviside function does not undergo significant changes in the interpolation process.

5.3 Irregular Functions, Example 1

The first example of an irregular function was chosen due to the presence of multiple complex sinusoidal components of various frequencies and amplitudes.

$$f(x) = \sin(4x) + 0.3\cos(12x) + 0.5\sin(7x)$$

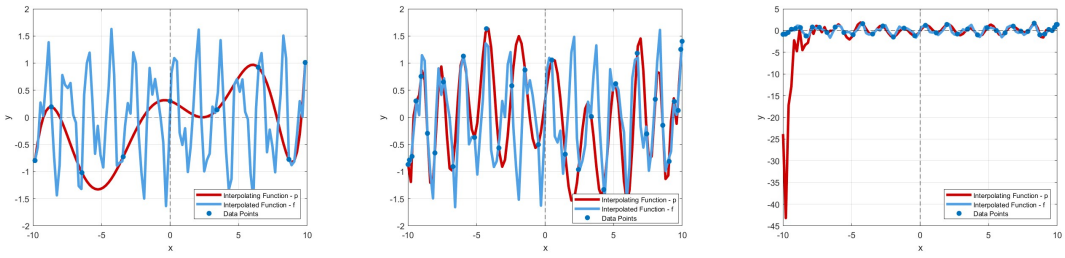

Figure 6: Graphs of the function f on the interval $[-10, 10]$ for a different number of interpolation nodes - roots of the Chebyshev polynomial. The number of nodes and their corresponding maximum interpolation errors are as follows: 9 nodes - error 2.8028, 32 nodes - error 3.2374, 40 nodes - error 77.4341.

In practice, when interpolating irregular functions, it is important to properly adjust the number of nodes to the characteristics of the function. Choosing too few nodes can lead to

inaccurate interpolation (see Figure 6), where the interpolating function does not reflect the complex behavior of the interpolated function.

On the other hand, choosing too many interpolation nodes can lead to overfitting (see Figure 6). This means that the interpolating function tries to replicate each data point too precisely, which can lead to excessive complexity of the interpolating function and difficulties in its interpretation.

5.4 Irregular Functions, Example 2

The next example of a function with irregular behavior is the function $f(x)$ defined as:

$$f(x) = \begin{cases} \sin(5x) & \text{for } x < 0 \\ e^{-2x} & \text{for } x \geq 0 \end{cases}$$

This is an example of a piecewise function, which is defined by different formulas on different intervals of x. It is characterized by sinusoidal oscillations to the left of zero and an exponentially decreasing nature to the right of zero. The function $f(x)$ is discontinuous at the point $x = 0$, as it changes its behavior at this boundary.

In the example below, we present the interpolation results for the function $f(x)$. Both graphs have the same number of interpolation nodes, only the interval changes. It is noted that the interpolating function tries to approximate the behavior of the interpolated function as closely as possible.

In the interval $[-1, 1]$, the function f behaves as a smoothly varying polynomial, leading to interpolation with a very small error. However, in the interval $[-10, 10]$, a greater difference between the interpolating and interpolated function is observed. This is due to the fact that the function f exhibits extremely different behavior on the positive and negative sides of x, which becomes more evident in this graph.

Therefore, we can conclude that the choice of the interpolation interval can significantly affect the quality of interpolation, especially when the interpolated function has varied behavior across different areas of the interval.

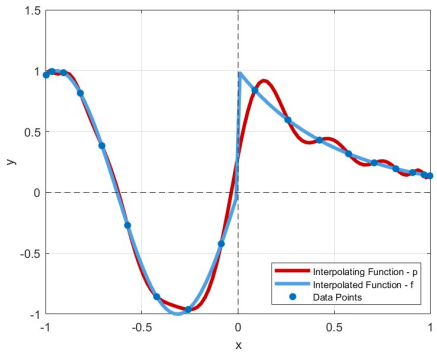

Figure 7: Graph of the function $f(x)$ for 18 Chebyshev nodes on the interval $[-1, 1]$. Interpolation error: 0.6987.

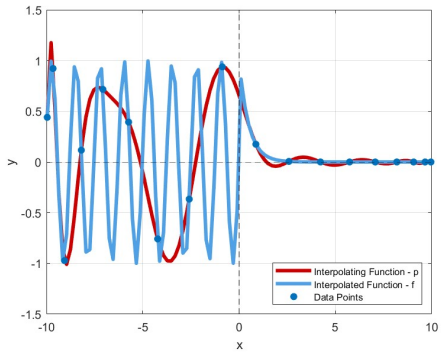

Figure 8: Graph of the function $f(x)$ for 18 Chebyshev nodes on the interval $[-10, 10]$. Interpolation error: 1.9666.

The situation is further illustrated by choosing a larger number of interpolation nodes. In the example below, the differences are even more apparent. Attention should be paid to the behavior of the function in the interval $[-10, 10]$, where this time, with a larger number of selected nodes, the interpolating function becomes more irregular and undergoes abrupt changes in value in the area of negative x.

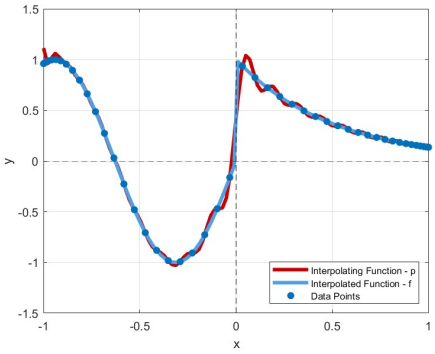

Figure 9: Graph of the function $f(x)$ for 48 Chebyshev nodes on the interval $[-1, 1]$. Interpolation error: 0.6536.

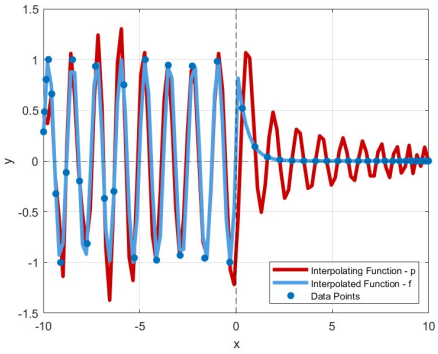

Figure 10: Graph of the function $f(x)$ for 48 Chebyshev nodes on the interval $[-10, 10]$. Interpolation error: 2.0716.

5.5 The Uniqueness Theorem in Practice

The theorem states that if we choose any $n+1$ distinct interpolation nodes, there exists exactly one interpolating polynomial that satisfies the interpolation condition.

Consider any polynomial $f(x)$ defined as:

$$f(x) = 3x^8 - 5x^7 + 2x^6 + 4x^5 - 6x^4 + 2x^3 + 3x^2 - x + 5.$$

The theorem asserts that, regardless of the choice of $n + 1$ distinct interpolation nodes, there exists exactly one interpolating polynomial $P_n(x)$ of degree at most n that satisfies the interpolation condition for these nodes.

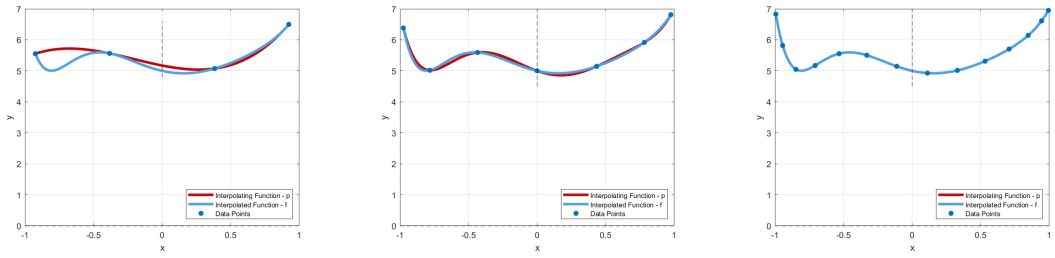

Figure 11: Graphs of the function f on the interval $[-1, 1]$ for different numbers of interpolation nodes - roots of the Chebyshev polynomial. The number of nodes and their corresponding maximum interpolation errors are as follows: 4 nodes - error 0.6767, 7 nodes - error 0.1204, 14 nodes - error 2.3803×10^{-13}.

Indeed, in the presented charts for $n < 9$, interpolation errors, which are visible, can be observed. However, for $n \geq 9$, the interpolated and interpolating functions overlap to the extent that the interpolation error becomes negligibly small. This is a result of numerical effects and approximations that become more accurate as the number of interpolation nodes is increased.

5.6 The Function $f(x) = \frac{1}{x}$

The function $f(x) = \frac{1}{x}$ is discontinuous at the point $x = 0$, which poses a challenge in its interpolation. The choice of interpolation nodes must be carefully considered to avoid errors near $x = 0$. In the case of the function $f(x) = \frac{1}{x}$, Chebyshev nodes, which avoid clustering around zero, can lead to more accurate interpolation results. Let's analyze the following examples:

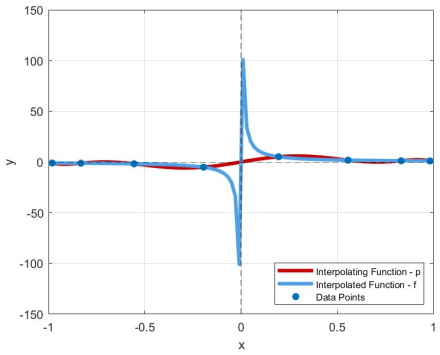

Figure 12: Graph of the function $f(x)$ for 8 Chebyshev nodes. Interpolation error: 1.0195×10^4.

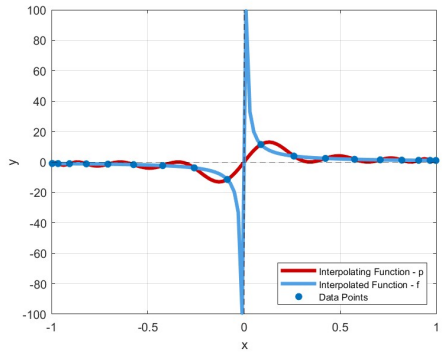

Figure 13: Graph of the function $f(x)$ for 18 Chebyshev nodes. Interpolation error: 1.0034×10^4.

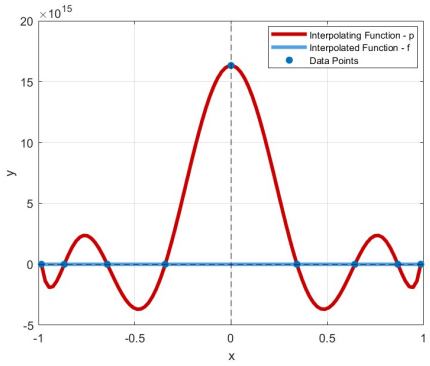

Figure 14: Graph of the function $f(x)$ for 9 Chebyshev nodes. Interpolation error: 1.6331×10^{16}.

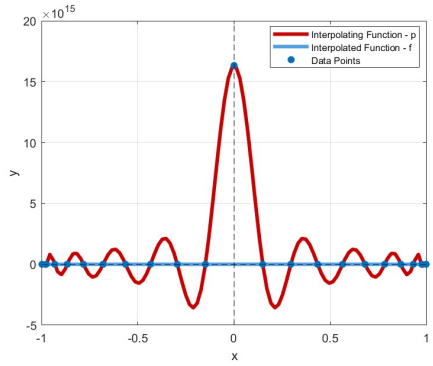

Figure 15: Graph of the function $f(x)$ for 21 Chebyshev nodes. Interpolation error: 1.6331×10^{16}.

In the interpolation process of the function $f(x) = \frac{1}{x}$ using Newton's polynomial and Chebyshev nodes, a clear dependency of the results on the parity of the number of chosen nodes was observed. This dependency has a significant impact on the quality and characteristics of interpolation, manifesting in differences in interpolation errors, which can reach values of the order of 10^{12}.

- Interpolation with an even number of nodes: In the case of an even number of nodes, the interpolating function tends to gently envelop the graph of the interpolated function. In this setting, interpolation seems to approximate the function better, maintaining lower variance in errors across the interval.
- Interpolation with an odd number of nodes: For an odd number of nodes, the interpolating function exhibits more pronounced deviations from the interpolated function. In these

cases, interpolation can significantly alter the shape of the graph, leading to much higher interpolation errors.

These observations suggest that the choice of the number of Chebyshev nodes, especially considering their parity, is a key factor in the interpolation process, especially for functions with strong nonlinearities and asymmetries, such as $\frac{1}{x}$.

Let's examine the values of the interpolated function - f and the interpolating function - p:

x	$p(x)$	$f(x)$	Error	Interp1 Error
0.98481	1.0311	1.0311	0	0
0.86603	1.3333	1.3333	0	0
0.64279	2.4203	2.4203	4.4409×10^{-16}	0
0.34202	8.5486	8.5486	0	0
6.1232×10^{-17}	2.6671×10^{32}	2.6671×10^{32}	3.6029×10^{16}	0
-0.34202	5.7646×10^{17}	8.5486	5.7646×10^{17}	0
-0.64279	2.8823×10^{18}	2.4203	2.8823×10^{18}	0
-0.86603	-1.7294×10^{18}	1.3333	1.7294×10^{18}	0
-0.98481	-1.1529×10^{18}	1.0311	1.1529×10^{18}	0

Table 1: Table of interpolation results for the function $f(x)$

Here we see a large discrepancy in both interpolation errors and the values of the function p, especially noticeable for negative values of x. The values in the columns for $p(x)$ and $f(x)$ are very large, close to 2.6671×10^{32}. Small differences between these values, at such immense numbers, can lead to significant errors.

Here are a few theories regarding these unusual results:

- Numerical precision: In floating-point systems, such as MATLAB, precision is limited. This means small differences between very large numbers may not be accurately represented.
- Error calculation: The error in the fourth column is calculated as $|\, p(x) - f(x) \,|$. Even small differences between $p(x)$ and $f(x)$ can be significant on this scale of values.
- Numerical representation of large numbers: In MATLAB, very large numbers may not be represented accurately due to limited precision, leading to apparent errors.

Considering the choice of the function $\frac{1}{x}$ and its characteristics, several additional reasons for anomalies in the interpolation results can be identified:

- The asymmetry of the function $\frac{1}{x}$ poses a challenge for interpolating polynomials, especially near zero, where the function has an infinite asymptote.
- An interpolation node close to zero (with a value of 6.1232×10^{-17}) leads to enormous values of the function $\frac{1}{x}$ at this point, which can cause numerical problems.
- The function $\frac{1}{x}$, with its nonlinear behavior and infinities, is difficult to accurately map with Newton's interpolating polynomial.
- High values of the function and coefficients can affect numerical precision in environments with floating-point arithmetic.
- For functions with strong nonlinearities, like $\frac{1}{x}$, other interpolation methods, such as spline or rational interpolation, may be more suitable than Newton's polynomial interpolation.

6 Summary and Conclusions

In this project, we conducted a detailed analysis of the convergence of the interpolation process for various mathematical functions, using Newton's interpolation method. In examining convergence, we paid special attention to the impact of the choice of interpolation nodes and the method used on the accuracy and efficiency of interpolation.

One of the key findings is that the choice and placement of interpolation nodes significantly affect the quality of interpolation. Chebyshev nodes, by avoiding clustering around critical points, proved effective in minimizing Runge's phenomenon, especially for functions with strong nonlinearities and asymmetries. Conversely, evenly spaced nodes often led to larger errors, especially at the ends of interpolation intervals.

Analyzing different types of functions, we noticed that irregular, oscillatory, or singular functions are more challenging to interpolate, requiring the use of a larger number of nodes or special interpolation methods. Functions such as $\frac{1}{x}$ pose unique challenges to Newton's polynomial interpolation, which may require alternative approaches.

Numerical errors and the way large numbers are represented in floating-point environments, such as MATLAB, significantly impact interpolation errors, especially with extreme function values.

In exploring the convergence of the interpolation process, we also understood that there is no one-size-fits-all solution to every interpolation problem. There are other interpolation methods that may prove more effective in certain cases, especially when the interpolated function has unique properties or when we want to avoid issues associated with high degrees of polynomials.

In summary, our research revealed the complexity and nuances associated with the interpolation process, showing that effective interpolation requires careful selection of methods and strategies tailored to the specific features and requirements of the mathematical function being interpolated.

7 References

7.1 Online Resources

- Newton's interpolation methods: https://sam.nitk.ac.in/courses/MA608/Newton%20interpolation%20methods.pdf
- MATLAB Interpolation: https://www.mathworks.com/help/matlab/interpolation.html
- Stanford Interpolation: https://graphics.stanford.edu/courses/cs205a-13-fall/assets/notes/chapter11.pdf
- Interpolation Error:
  - https://math.okstate.edu/people/binegar/4513-F98/4513-l16.pdf
  - https://www.math.ntnu.no/emner/TMA4130/2021h/lectures/ErrorTheoryInterpolation.pdf

7.2 Books and Lecture Notes

References

[1] D. Kincaid, W. Cheney. *Numerical Analysis*. WNT, 2005.

[2] Iwona Wróbel. *Lecture Notes on Numerical Methods*.