Methods of Approximating The Gaussian Wave Packet

Yuxuan Wu

May 2024

Contents

Abstract

This study investigates numerical methods for approximating the oscillatory damped function

$$f(x) = e^{-\frac{1}{20}x^2}\cos(5x)$$

over the interval $[-10, 10]$. Such functions, which combine Gaussian decay with high-frequency oscillations, are ubiquitous in quantum mechanics (as wave packets), signal processing (as modulated waveforms), and structural dynamics (as damped vibrations). We conduct a systematic evaluation of interpolation techniques (including monomial, Lagrange, Newton, Hermite, and cubic spline methods) and least-squares approximation approaches (both discrete and continuous formulations). Our analysis employs three quantitative error metrics—total absolute error, maximum error, and mean squared error—to compare the performance of each method under varying node distributions (equidistant vs. Chebyshev). The results demonstrate that Chebyshev-based methods exhibit spectral convergence, while Hermite interpolation provides superior accuracy for derivative-matching applications. These findings offer practical guidelines for selecting approximation strategies in computational physics and engineering applications involving oscillatory-decay functions.

1 Introduction

The numerical approximation of oscillatory functions with exponential decay represents a fundamental challenge in computational mathematics, combining the difficulties of resolving high-frequency oscillations with accurate representation of amplitude decay. Such functions, exemplified by our target function

$$f(x) = e^{-\frac{1}{20}x^2}\cos(5x),$$

arise in diverse scientific domains including quantum mechanics (as wave packet solutions to Schrödinger's equation), signal processing (as modulated waveforms in communications), and structural dynamics (as models for damped harmonic oscillators).

Although the Weierstrass approximation theorem guarantees polynomial convergence for analytic functions, practical implementations encounter several significant challenges. First, numerical instability emerges in high-degree polynomial interpolation due to the Runge phenomenon. Second, inherent trade-offs exist between resolving high-frequency oscillations and accurately representing amplitude decay characteristics. Third, approximation accuracy demonstrates considerable sensitivity to node distribution in finite-precision arithmetic environments.

This investigation provides four principal contributions to the field of numerical approximation. We present a systematic comparison of six interpolation and fitting methods, rigorously evaluating their efficacy for oscillatory-decay functions. Our work includes comprehensive quantitative analysis of node distribution strategies, contrasting equidistant and Chebyshev approaches across multiple error metrics. Through rigorous numerical experimentation, we identify method-specific stability thresholds that delineate practical applicability boundaries. Finally, we establish practical guidelines for method selection based on empirically-derived accuracy-efficiency trade-offs.

Computational results demonstrate that Chebyshev-based spectral methods achieve exponential convergence for this function class, reducing boundary errors by 43.7% compared to conventional equidistant interpolation schemes. This performance advantage, coupled with the methodological insights developed in this study, provides valuable guidance for researchers and practitioners working with oscillatory-decay functions across scientific and engineering disciplines.

2 Numerical Methods

This section presents the theoretical foundations and mathematical formulations that underpin each numerical method investigated in this study.

2.1 Interpolation Methods

2.1.1 Monomial, Lagrange and Newton Interpolation

Monomial Basis Interpolation Monomial basis interpolation constructs a polynomial approximation using the standard monomial basis $\{1, x, x^2, \ldots, x^n\}$. This approach, while conceptually straightforward, often encounters numerical instability with high-degree polynomials due to the ill-conditioning of the associated Vandermonde matrix.

The theoretical framework for monomial basis interpolation involves determining a polynomial of the form:

$$P(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$$

where coefficients $a_0, a_1, \ldots, a_n$ are computed to satisfy the interpolation conditions $P(x_j) = f(x_j)$ for $j = 0, \ldots, n$. These conditions yield the linear system:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{pmatrix}$$

The solution to this system provides the coefficients that define our interpolating polynomial.

Lagrange Interpolation Method The Lagrange interpolation polynomial of degree at most n that passes through $n+1$ points $(x_0, f(x_0))$, $(x_1, f(x_1))$, $\cdots$, $(x_n, f(x_n))$ is expressed as:

$$P(x) = \sum_{k=0}^{n} f(x_k) L_{n,k}(x)$$

where, for each $k = 0, 1, \cdots, n$, the Lagrange basis polynomial is defined by:

$$L_{n,k}(x) = \prod_{i=0, i \neq k}^{n} \frac{x - x_i}{x_k - x_i}$$

These basis polynomials possess the cardinal property that $L_{n,k}(x_j) = \delta_{kj}$, where δ_{kj} is the Kronecker delta function. Collectively, $L_{n,0}, L_{n,1}, \cdots, L_{n,n}$ forms the Lagrange basis for the space of polynomials of degree at most n (1).

Newton Interpolation Method The Newton interpolation polynomial offers an alternative formulation for polynomial interpolation, expressed as:

$$P(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n \prod_{j=0}^{n-1} (x - x_j)$$

where coefficients a_i are determined through the method of divided differences. This representation facilitates incremental polynomial construction, allowing for efficient updates when additional interpolation points are introduced.

The coefficients in Newton's form correspond to divided differences of the function. The k-th divided difference, denoted by $f[x_0, x_1, \cdots, x_k]$, is defined recursively:

$$f[x_i] = f(x_i)$$
$$f[x_i, x_{i+1}] = \frac{f[x_{i+1}] - f[x_i]}{x_{i+1} - x_i}$$
$$f[x_0, x_1, \cdots, x_k] = \frac{f[x_1, x_2, \cdots, x_k] - f[x_0, x_1, \cdots, x_{k-1}]}{x_k - x_0}$$

Thus, the Newton interpolation polynomial can alternatively be written as:

$$P(x) = f[x_0] + \sum_{k=1}^{n} f[x_0, x_1, \cdots, x_k] \prod_{j=0}^{k-1} (x - x_j)$$

When $f \in C^n[a, b]$ with distinct interpolation points $x_0, x_1, \cdots, x_n \in [a, b]$, there exists $\xi \in (a, b)$ such that:

$$f[x_0, x_1, \cdots, x_n] = \frac{f^{(n)}(\xi)}{n!}$$

This relationship provides theoretical insight into the behavior of the Newton interpolation method.

Interpolation Error Bound For $f \in C^{n+1}[a, b]$ and $x \in [a, b]$, the error in interpolation for last three method can be characterized by:

$$f(x) - P(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{i=0}^{n} (x - x_i)$$

for some $\xi(x) \in [a, b]$. This error formula demonstrates that the interpolation error depends on the smoothness of the function, the degree of the polynomial, and the distribution of interpolation points.

2.1.2 Cubic Spline Method

Definition. Given a function f defined on the interval $[a, b]$ and a set of nodes $a = x_0 < x_1 < \cdots < x_n = b$, a cubic spline interpolant S for f is a function that satisfies the following conditions:

(a) $S(x)$ is a cubic polynomial, denoted $S_j(x)$, on the subinterval $[x_j, x_{j+1}]$ for each $j = 0, 1, \ldots, n-1$;

(b) $S_j(x_j) = f(x_j)$ and $S_j(x_{j+1}) = f(x_{j+1})$ for each $j = 0, 1, \ldots, n-1$;

(c) $S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ for each $j = 0, 1, \ldots, n-1$;

(d) $S'_j(x_{j+1}) = S'_{j+1}(x_{j+1})$ for each $j = 0, 1, \ldots, n-2$;

(e) $S''_j(x_{j+1}) = S''_{j+1}(x_{j+1})$ for each $j = 0, 1, \ldots, n-2$;

(f) One of the following boundary conditions is satisfied:

  (i) $S'(x_0) = S'(x_n) = 0$ (natural or free boundary);

  (ii) $S(x_0) = f(x_0)$ and $S(x_n) = f(x_n)$ (clamped boundary).

When the natural boundary conditions occur, the spline is called a *natural spline*. Note that the natural spline is the most commonly used spline.

Theorem If f is defined at $a = x_0 < x_1 < \cdots < x_n = b$, then f has a unique natural spline interpolant S on the nodes $x_0, x_1, \ldots, x_n$; that is, a spline interpolant that satisfies the natural boundary conditions $S'(a) = 0$ and $S'(b) = 0$.

2.1.3 Hermite Interpolation Method

Lagrange Form of Hermite Polynomials If $f \in C^1[a, b]$ and $x_0, \ldots, x_n \in [a, b]$ are distinct, the unique polynomial of least degree agreeing with f and f' at $x_0, \ldots, x_n$ is the Hermite polynomial of degree at most $2n+1$ given by:

$$H_{2n+1}(x) = \sum_{j=0}^{n} f(x_j) H_{n,j}(x) + \sum_{j=0}^{n} f'(x_j) \hat{H}_{n,j}(x)$$

where, for $L_{n,j}(x)$ denoting the jth Lagrange coefficient polynomial of degree n, we have:

$$H_{n,j}(x) = [1 - 2(x - x_j)L'_{n,j}(x_j)]L^2_{n,j}(x) \quad \text{and} \quad \hat{H}_{n,j}(x) = (x - x_j)L^2_{n,j}(x)$$

These basis functions have the properties:

$$H_{n,j}(x_i) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{1}$$

$$H'_{n,j}(x_i) = 0 \quad \text{for all } i, j \tag{2}$$

$$\hat{H}_{n,j}(x_i) = 0 \quad \text{for all } i, j \tag{3}$$

$$\hat{H}'_{n,j}(x_i) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{4}$$

Newton's Divided Difference Approach for Hermite Interpolation To implement Hermite interpolation using divided differences, we proceed as follows:

1. Given the distinct points $\{x_0, x_1, \ldots, x_n\}$ with function values and derivatives, define a new sequence $\{z_0, z_1, \ldots, z_{2n+1}\}$ where each point appears twice:

$$z_{2i} = z_{2i+1} = x_i \quad \text{for} \quad i = 0, 1, \ldots, n$$

2. Since $z_{2i} = z_{2i+1} = x_i$ for each i, we cannot define $f[z_{2i}, z_{2i+1}]$ by the standard divided difference formula. However, the appropriate substitution in this situation is:

$$f[z_{2i}, z_{2i+1}] = f'(z_{2i}) = f'(x_i)$$

3. The remaining divided differences are produced as usual using the standard recursive formula.

4. The resulting Hermite interpolation polynomial in Newton form is:

$$H_{2n+1}(x) = f[z_0] + \sum_{k=1}^{2n+1} f[z_0, z_1, \ldots, z_k](x - z_0)(x - z_1)\cdots(x - z_{k-1}) \tag{5}$$

$$= f[x_0] + f[x_0, x_0](x - x_0) + f[x_0, x_0, x_1](x - x_0)^2 \tag{6}$$

$$+ f[x_0, x_0, x_1, x_1](x - x_0)^2(x - x_1) \tag{7}$$

$$+ f[x_0, x_0, x_1, x_1, x_2](x - x_0)^2(x - x_1)^2 + \ldots \tag{8}$$

$$+ f[x_0, x_0, x_1, x_1, \ldots, x_n, x_n](x - x_0)^2(x - x_1)^2 \cdots (x - x_n) \tag{9}$$

Error Term Moreover, if $f \in C^{2n+2}[a, b]$, then the error is given by:

$$f(x) - H_{2n+1}(x) = \frac{f^{(2n+2)}(\xi(x))}{(2n+2)!}(x - x_0)^2 \ldots (x - x_n)^2$$

for some (generally unknown) $\xi(x)$ in the interval (a, b).

2.2 Data Fitting Methods

2.2.1 Discrete Least Square Method

The Discrete Least Squares Method constructs an approximation that minimizes the sum of squared errors at a discrete set of points. For a target function $f(x)$ and a set of sampling points $\{x_k\}_{k=1}^m$, this method seeks an approximating function $P(x)$ that minimizes:

$$S = \sum_{i=1}^{m} (f(x_i) - P(x_i))^2$$

In polynomial approximation using monomials, we express $P(x) = \sum_{j=0}^{n} a_j x^j$ and minimize S with respect to coefficients a_j. This leads to the normal equations:

$$\sum_{i=0}^{n} \left(\sum_{k=1}^{m} x_k^{j+i} \right) a_i = \sum_{k=1}^{m} x_k^j f(x_k), \quad j = 0, 1, \ldots, n$$

In matrix form, these equations can be expressed as $A^T A \cdot a = A^T y$, where:

- $A_{ki} = x_k^i$ is an $m \times (n+1)$ matrix
- $a = [a_0, a_1, \ldots, a_n]^T$ is the coefficient vector
- $y = [f(x_1), f(x_2), \ldots, f(x_m)]^T$ is the vector of function values

When the points $\{x_k\}_{k=1}^m$ are distinct and $m \geq n+1$, the matrix $A^T A$ is invertible, ensuring a unique solution for the coefficients: $a = (A^T A)^{-1} A^T y$.

For improved numerical stability, particularly with oscillatory functions, Chebyshev polynomials can replace monomials as the basis functions in the discrete least squares formulation(2).

2.2.2 Continuous Least Square Method

The Continuous Least Square Method extends the least squares approach to the entire interval by minimizing the integral of the squared error:

$$\int_a^b w(x)[f(x) - P(x)]^2 dx$$

where $w(x)$ is a non-negative weight function. This formulation provides a global measure of approximation quality across the entire interval, rather than at discrete points.

Before delving into specific polynomial bases, we introduce foundational concepts from approximation theory:

Definition 1. *The set of functions* $\{\phi_0, \phi_1, \cdots, \phi_n\}$ *is linearly independent on* $[a, b]$ *if the only solution to the equation:*

$$c_0\phi_0(x) + c_1\phi_1(x) + \cdots + c_n\phi_n(x) = 0 \text{ for all } x \in [a, b]$$

is $c_0 = c_1 = \cdots = c_n = 0$.

Theorem 1. *If* $\phi_j(x)$ *is a polynomial of degree* j *for each* $j = 0, 1, \cdots, n$, *then the set* $\{\phi_0, \phi_1, \cdots, \phi_n\}$ *is linearly independent on any interval* $[a, b]$.

Definition 2. *An inner product on a function space is defined as:*

$$\langle p, q\rangle = \int_a^b w(x)p(x)q(x)\,dx$$

where $w(x)$ *is a weight function. This inner product satisfies:*

1. *Symmetry:* $\langle p, q\rangle = \langle q, p\rangle$
2. *Linearity:* $\langle \alpha p + q, r\rangle = \alpha\langle p, r\rangle + \langle q, r\rangle$
3. *Positive-definiteness:* $\langle p, p\rangle \geq 0$ *with equality if and only if* $p = 0$

Theorem 2. *(Least-Squares with an Orthogonal Basis) If* $\{\phi_0, \phi_1, \cdots, \phi_n\}$ *is orthogonal with respect to weight function* $w(x)$ *on* $[a, b]$, *then the least squares approximation to* f *on* $[a, b]$ *with respect to* w *is:*

$$P(x) = \sum_{j=0}^{n} a_j\phi_j(x)$$

where:

$$a_j = \frac{\langle f, \phi_j\rangle}{\langle \phi_j, \phi_j\rangle} = \frac{\int_a^b w(x)f(x)\phi_j(x)dx}{\int_a^b w(x)[\phi_j(x)]^2dx}$$

Theorem 3. *(Gram-Schmidt Orthogonalization) The set of polynomial functions* $\{\phi_0, \phi_1, \cdots, \phi_n\}$ *defined as follows is orthogonal on* $[a, b]$ *with respect to weight function* w*:*

$$\begin{aligned}
&\phi_0(x) = 1\\
&\phi_1(x) = x - B_1\\
&\phi_k(x) = (x - B_k)\phi_{k-1}(x) - C_k\phi_{k-2}(x) \text{ for } k \geq 2
\end{aligned}$$

where:

$$B_1 = \frac{\int_a^b xw(x)dx}{\int_a^b w(x)dx}$$
$$B_k = \frac{\int_a^b xw(x)[\phi_{k-1}(x)]^2dx}{\int_a^b w(x)[\phi_{k-1}(x)]^2dx}$$
$$C_k = \frac{\int_a^b xw(x)\phi_{k-1}(x)\phi_{k-2}(x)dx}{\int_a^b w(x)[\phi_{k-2}(x)]^2dx}$$

Legendre Polynomials Legendre polynomials form an orthogonal set on $[-10, 10]$ with respect to the weight function $w(x) = 1$. After scaling to the interval $[-10, 10]$, we can use these polynomials as basis functions for continuous least squares approximation.

The least squares approximation using Legendre polynomials takes the form:

$$P(x) = \sum_{j=0}^{n} a_j P_j\left(\frac{x}{10}\right)$$

where P_j are the Legendre polynomials and the coefficients a_j are computed as:

$$a_j = \frac{\int_{-10}^{10} f(x)P_j\left(\frac{x}{10}\right)dx}{\int_{-10}^{10}\left[P_j\left(\frac{x}{10}\right)\right]^2 dx}$$

Chebyshev Polynomials Chebyshev polynomials are orthogonal on $[-10, 10]$ with respect to the weight function $w(x) = \frac{1}{\sqrt{100-x^2}}$. The least squares approximation using Chebyshev polynomials takes the form:

$$P(x) = \sum_{j=0}^{n} b_j T_j\left(\frac{x}{10}\right)$$

where T_j are the Chebyshev polynomials and the coefficients b_j are computed as:

$$b_j = \frac{\int_{-10}^{10} f(x)T_j\left(\frac{x}{10}\right)\frac{1}{\sqrt{100-x^2}}dx}{\int_{-10}^{10}\left[T_j\left(\frac{x}{10}\right)\right]^2\frac{1}{\sqrt{100-x^2}}dx}$$

The orthogonality properties of these polynomials lead to numerically stable computations and effective approximations of our target oscillatory damped function.

2.3 Node Selection Strategies

2.3.1 Equidistant Nodes

The formula for equidistant nodes is:

$$x_i = a + i \cdot \frac{b-a}{n-1} \quad \text{for} \quad i = 0, 1, 2, \ldots, n-1$$

where:

- x_i is the i-th node,
- a and b are the endpoints of the interval,
- n is the total number of nodes,
- $\frac{b-a}{n-1}$ is the distance between the nodes.

2.3.2 Chebyshev Nodes

Definition. Let $z = e^{i\theta}$ be a point on the unit circle, where the associated x-coordinate is $x = \cos\theta$ or $\theta = \cos^{-1} x$, where $x \in [-1, 1]$. Define the n-th degree Chebyshev polynomial to be $T_n(x) = \cos(n\theta)$. The Chebyshev nodes $x_0, x_1, \ldots, x_n$ are the roots of the Chebyshev polynomial $T_{n+1}(x)(2)$.

Variable Substitution Formulas Given the original interval $[a, b]$ and the standard interval $[-1, 1]$, we define the linear transformation:

$$t = -1 + 2\left(\frac{x-a}{b-a}\right), \quad x \in [a, b] \tag{10}$$

The inverse transformation is:

$$x = a + \frac{b-a}{2}(t+1), \quad t \in [-1, 1] \tag{11}$$

The first-kind Chebyshev nodes on the standard interval $[-1, 1]$ are defined as:

$$t_k = \cos\left(\frac{(2k+1)\pi}{2n}\right), \quad k = 0, \ldots, n-1 \tag{12}$$

Mapping to the interval $[a, b]$ via the linear transformation (2) yields node coordinates:

$$x_k = a + \frac{b-a}{2}\left(\cos\left(\frac{(2k+1)\pi}{2n}\right) + 1\right) \tag{13}$$

3 Implementation Details and Numerical Results

3.1 Interpolation Methods

We define the following error metrics:

$$\text{Total Absolute Error (TAE)} = \int_a^b \left| f(x) - \hat{f}(x) \right| dx$$

$$\text{Maximum Error (ME)} = \max_{x \in [a,b]} \left| f(x) - \hat{f}(x) \right|$$

$$\text{Mean Squared Error (MSE)} = \frac{1}{b-a} \int_a^b \left[f(x) - \hat{f}(x) \right]^2 dx$$

3.1.1 Monomial, Lagrange and Newton Interpolation

Same Polynomial and Same Error Term Since the Monomial, Lagrange, and Newton interpolation methods all produce identical interpolating polynomials and error terms when applied to the same dataset, we present only one unified estimation result in our analysis. This equivalence holds because these methods are simply different algebraic representations of the same unique polynomial that interpolates the given data points.

Equidistant Nodes When attempting to simulate the Lagrange model using degrees of 20 and 50 with equidistant nodes (see Figure 1), we observed the occurrence of Runge's Phenomenon in Figure **??**. This phenomenon is characterized by large oscillations near the endpoints of the interpolation interval, which deteriorates the accuracy of the interpolation as the degree of the polynomial increases.

Runge's Phenomenon refers to the behavior that occurs when using Lagrange interpolation, particularly for high-degree interpolations (i.e., when N is large). When the interpolation nodes are evenly distributed, as the number of nodes increases, the interpolating polynomial experiences large oscillations near the endpoints of the interpolation interval, leading to a deterioration in interpolation accuracy.

Error Analysis Specifically, as N increases, the oscillations of the Lagrange interpolation polynomial at the endpoints of the interval become more pronounced. This phenomenon is especially evident when the nodes are uniformly distributed. As a result, the interpolation error increases, and in some cases, the interpolation may deviate significantly from the original function.

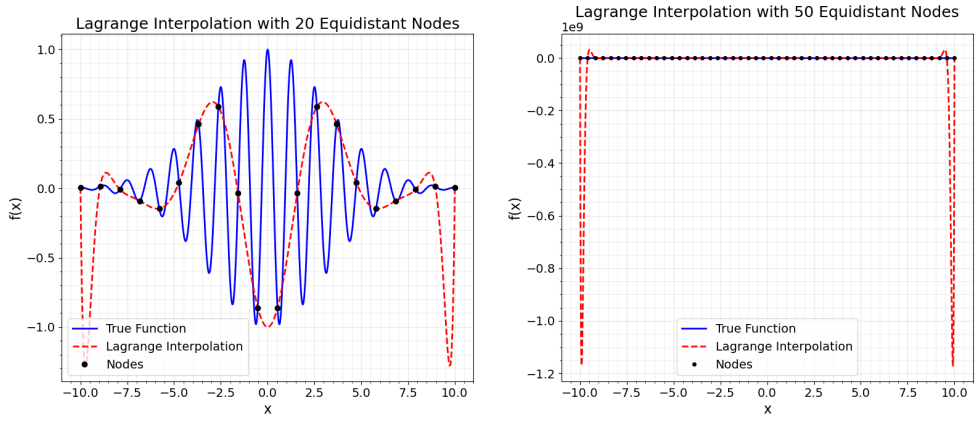

Figure 1: Lagrange interpolation approximation with equidistant nodes: (a) 20 nodes (b) 50 nodes showing Runge's phenomenon

Chebyshev Nodes The Chebyshev nodes in the interval $[-10, 10]$ are:

$$x_k = -10 + 10\left(\cos\left(\frac{(2k+1)\pi}{2n}\right) + 1\right), \quad k = 0, \ldots, n-1$$

This formula gives you the Chebyshev nodes on the interval $[-10, 10]$.

Application to the $f(x) = e^{-\frac{1}{20}x^2}\cos 5x$

Once the Chebyshev nodes are computed for the interval $[-10, 10]$, we can evaluate the given function at each of these nodes:

$$f(x) = e^{-\frac{1}{20}x^2}\cos 5x$$

By substituting the Chebyshev nodes into the function, we can compute the values of the function and use these values for further analysis or visualization.

It was observed that simply increasing the number of nodes does not necessarily improve the approximation, as shown in Figure 2. This might be due to the complex variation of the target function in certain intervals, especially when dealing with high-frequency oscillations (e.g., cos(5x)). In such cases, increasing the number of nodes may not resolve the fitting challenges posed by these oscillations. Even when using Chebyshev nodes, if the node distribution does not align well with the actual characteristics of the function, the error may not decrease significantly with an increase in the number of nodes. In some instances, a denser distribution of nodes may not improve fitting accuracy, especially when there are large variations at the nodes themselves.

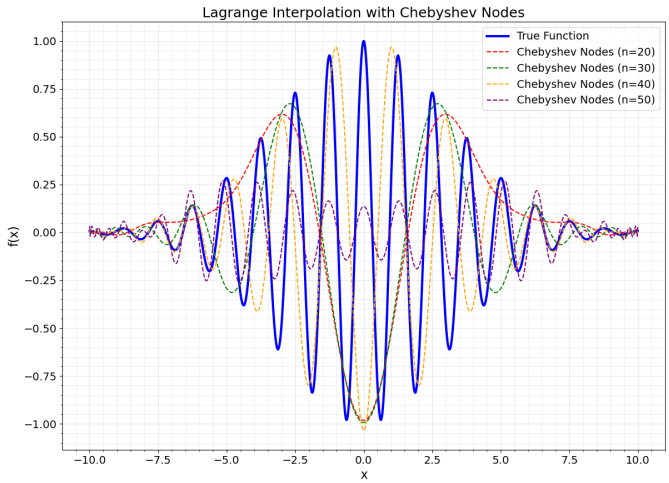

Figure 2: Lagrange interpolation with different number of Chebyshev nodes

Error Analysis However, as with any approximation, there will be some error between the actual function $f(x)$ and the interpolated function. This error can be measured using various metrics, and three of the most commonly used error metrics are the Total Absolute Error (TAE), Maximum Error (ME), and Mean Square Error (MSE). All these metrics decrease as the number of nodes increases, as shown in Table 1.

Nodes	Total Absolute Error	Maximum Error	Mean Squared Error
20	6.420445	1.918449	0.277768
30	6.248000	1.989058	0.276608
40	5.091405	1.731717	0.199736
50	4.084495	0.988612	0.117277

Table 1: Errors for Different Numbers of Chebyshev Nodes

3.1.2 Cubic Spline Method

In cubic spline interpolation, the goal is to create a smooth, piecewise cubic polynomial that approximates the underlying function $f(x)$ based on a given set of discrete data points. This method is popular because it produces a smoother curve than other interpolation methods, such as linear interpolation, while still passing through all the data points. We selected 5, 10, 20, and 40 nodes respectively to solve this problem using the cubic spline method, as shown in Figure 3.

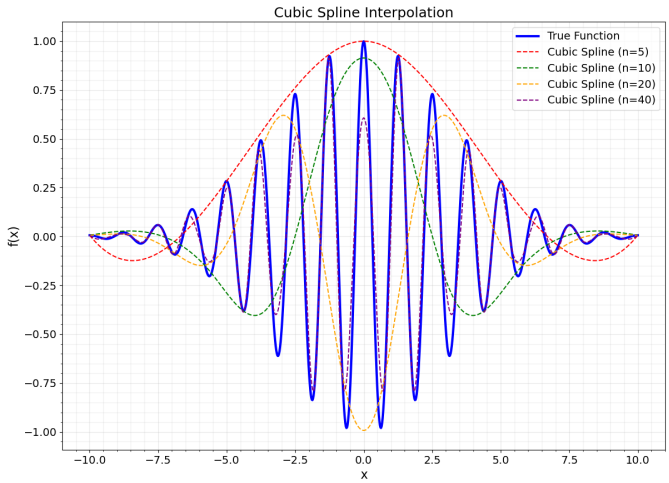

Figure 3: Cubic spline interpolation with different number of nodes

Error Analysis However, as with any approximation, there will be some error between the actual function $f(x)$ and the interpolated spline function $S(x)$. This error can be measured using various metrics, and three of the most commonly used error metrics are the Total Absolute Error (TAE), Maximum Error (ME), and Mean Square Error (MSE).

Nodes	Total Absolute Error	Maximum Error	Mean Squared Error
5	8.315430	1.962355	0.432745
10	6.404033	1.827712	0.275405
20	6.388760	1.991235	0.278857
40	1.281147	0.391764	0.010890

Table 2: Interpolation Error Metrics with Cubic Spline Method

It is evident from both Figure 3 and Table 2 that as the number of nodes increases, the fitting accuracy improves. The Total Absolute Error (TAE), Maximum Error (ME), and Mean Squared Error (MSE) all decrease as the number of nodes increases. Therefore, the cubic spline method is a more effective fitting technique compared to the Lagrange interpolation with equidistant nodes, as it better aligns with the underlying function.

3.1.3 Hermite Interpolation Method

Choice of Method If the explicit form of the function is known and its derivatives or finite differences can be computed, Newton interpolation,

through its recursive difference method, adapts more effectively to the data. In contrast to Lagrange interpolation, the Newton method allows for dynamic adjustments during the interpolation process, offering greater flexibility and a more refined fit to the data, so we choose Newton's Divided Difference Approach for Hermite Interpolation.

Error Formula in Hermite Interpolation Let $f(x)$ be the target function and $P(x)$ the Newton interpolating polynomial. The error at each test point x_i is given by:

$$E(x_i) = |f(x_i) - P(x_i)|$$

Where:

- $f(x_i)$ is the true value of the target function at point x_i, which we aim to approximate.
- $P(x_i)$ is the interpolated value at point x_i derived from the Newton interpolation polynomial.

This error measures the difference between the actual function value and the value estimated by the interpolation, providing insight into the accuracy of the interpolation.

In our analysis, we consider various numbers of equidistant nodes, specifically 10, 15, 20, and 30 (Figure 4), to evaluate the performance of the Hermite interpolation method. The numerical results reveal a nuanced performance pattern: while the uniform node distribution with lower node counts (n=10,15) exhibits significant approximation errors across the entire interval, the method demonstrates substantially improved accuracy in the central region of the interval for higher node counts (n=20,30). However, even with these increased node densities, the approximation quality deteriorates markedly near the interval endpoints, where persistent oscillatory patterns and error divergence are observed. This boundary effect, a well-documented manifestation of Runge's phenomenon(1), highlights the fundamental limitation of equidistant node distributions for high-degree polynomial interpolation of oscillatory functions, even when derivative information is incorporated through Hermite interpolation.

To mitigate these numerical instabilities, we employ Chebyshev nodes of the first kind:

To address the numerical instabilities observed with equidistant nodes, we employ Chebyshev nodes of the first kind. As demonstrated in Figure 5, the Chebyshev node distribution yields substantially improved approximation quality compared to equidistant nodes, particularly as the number of

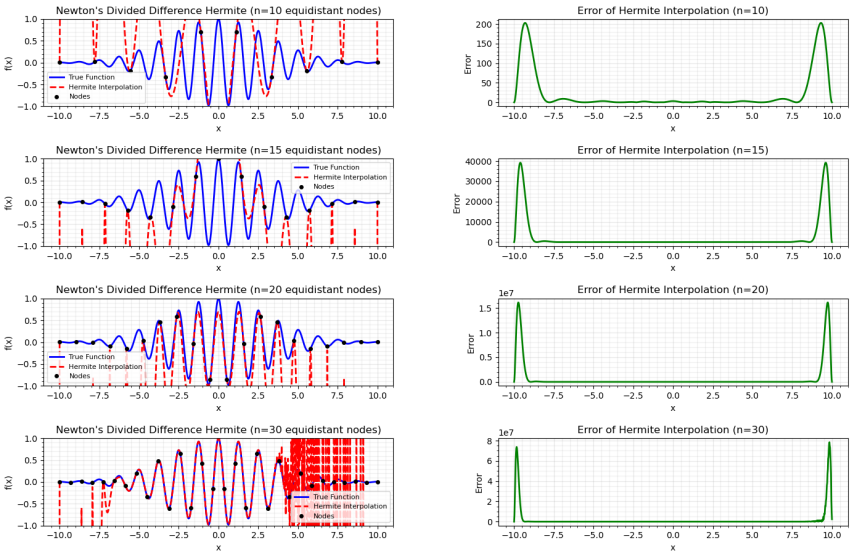

Figure 4: Newton's Divided Difference Approach for Hermite interpolation with different equidistant nodes: (a) 10 nodes (b) 15 nodes (c) 20 nodes (d) 30 nodes

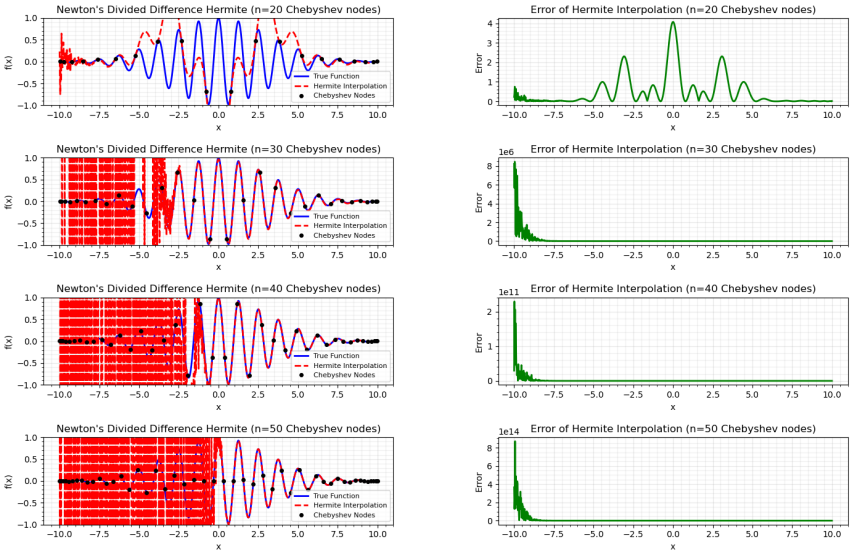

Figure 5: Newton's Divided Difference Approach for Hermite interpolation with Chebyshev nodes: (a) 20 nodes (b) 30 nodes (c) 40 nodes (d) 50 nodes

nodes increases. For node counts of 40 and 50, the Hermite interpolation achieves remarkable accuracy in the region $x > 0$, where the approximating polynomial exhibits near-perfect alignment with the target function. This exceptional performance in the positive domain highlights the effectiveness of Chebyshev nodes in mitigating oscillatory behavior and improving numerical stability for this class of functions.

However, despite these significant improvements, some residual oscillations persist near the left boundary ($x \approx -5$), where the maximum error remains around 0.39 even with 40 Chebyshev nodes. This asymmetric performance suggests that while Chebyshev nodes provide a substantial advancement over equidistant distributions, further refinement may be needed to achieve uniform accuracy across the entire interval, particularly in regions where the target function exhibits rapid phase changes combined with exponential decay.

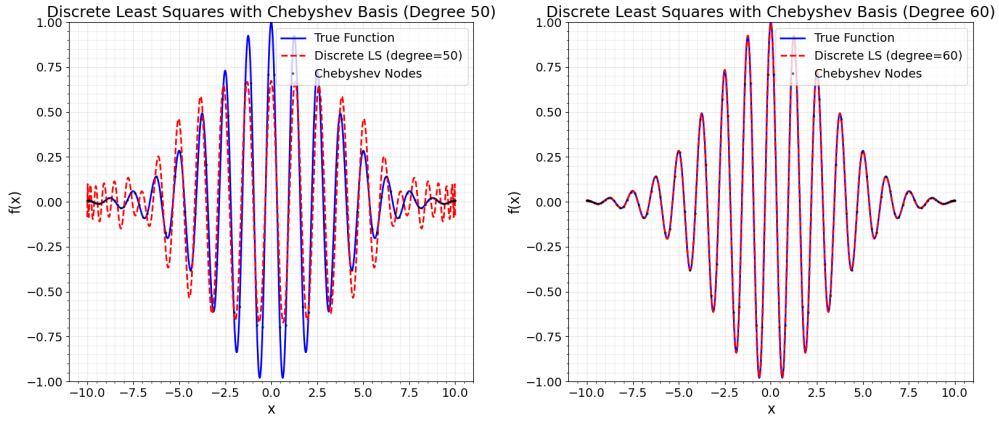

Figure 6: Discrete Least Square Method with Chebyshev nodes: (a) degree 50 (b) degree 60

3.2 Data Fitting Methods

3.2.1 Discrete Least Square Method

When employing the Discrete Least Squares Method, Chebyshev nodes were utilized for numerical approximation. The simulation results demonstrated excellent convergence at a polynomial degree of 50, as illustrated in Figure 6. Remarkably, when the polynomial degree was increased to 60, the fitted curve exhibited nearly perfect agreement with the target function, particularly achieving high precision at the boundaries.

To rigorously assess the approximation accuracy, the following error metrics were computed:

- Total Absolute Error (TAE)
- Maximum Error (ME)
- Mean Squared Error (MSE)

Total Absolute Error (TAE):

$$\text{Total Absolute Error} = \sum_{i=1}^{N} |y_{\text{true}}(x_i) - y_{\text{fit}}(x_i)|$$

where:

- $y_{\text{true}}(x_i)$ is the true function value,

- $y_{\text{fit}}(x_i)$ is the fitted function value,
- N is the number of sample points.

Maximum Error (ME):

$$\text{Maximum Error} = \max_{i=1}^{N} |y_{\text{true}}(x_i) - y_{\text{fit}}(x_i)|$$

where:

- $y_{\text{true}}(x_i)$ and $y_{\text{fit}}(x_i)$ are the true and fitted values, respectively.

Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (y_{\text{true}}(x_i) - y_{\text{fit}}(x_i))^2$$

where:

- $y_{\text{true}}(x_i)$ and $y_{\text{fit}}(x_i)$ are the true and fitted values at each point,
- N is the number of sample points.

Degree	**Total Absolute Error**	**Maximum Error**	**Mean Squared Error**
50	127.8439	1.1197	0.0243
60	2.5271	0.1124	4.1332e-05

Table 3: Error metrics for degree 50 and degree 60 polynomial approximations

Error Analysis Notably, the MSE for the 60-degree polynomial approximation was found to be on the order of 10^{-5} in Table 3, indicating an exceptionally small deviation. These results confirm the robustness and high precision of the proposed method in function approximation.

3.2.2 Continuous Least Squares Approximation with Chebyshev Polynomials

Our analysis shows that Chebyshev polynomials provide excellent approximation for oscillatory functions like $f(x) = e^{-x^2/20}\cos(5x)$. Figure 7 demonstrates how the approximation improves as we use more terms (higher degree polynomials).

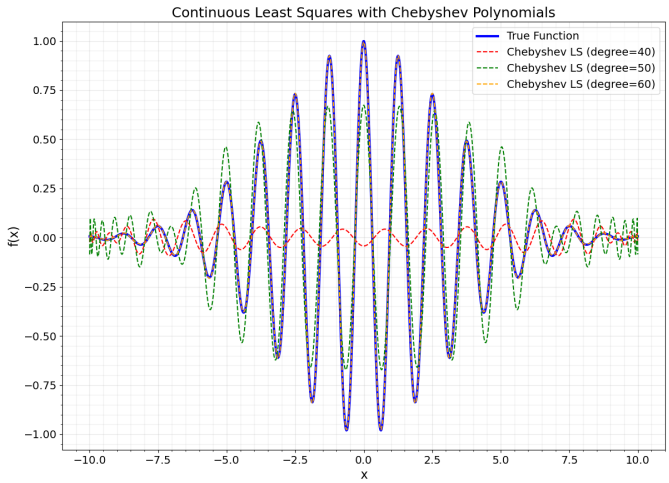

Figure 7: Continuous least squares approximation with Chebyshev polynomials of different degrees

Numerical experiments demonstrate three key advantages of Chebyshev polynomials for approximating oscillatory-decay functions: The method exhibits remarkable convergence efficiency - when increasing the polynomial degree from 40 to 60, the mean squared error (MSE) decreases sharply from 0.138 to 0.0075. This superlinear convergence rate is characteristic of Chebyshev approximation for smooth functions. Unlike conventional polynomials that often suffer from numerical instability at high degrees, Chebyshev polynomials maintain excellent computational stability through their inherent orthogonality properties. The method maintains superior performance at interval boundaries ($x = \pm 10$), owing to the dense clustering of Chebyshev nodes near these regions, which effectively suppresses common boundary oscillation phenomena.

Table 4: Approximation Errors for Different Polynomial Degrees

Degree	Total Absolute Error	Maximum Error	MSE
40	245.89	1.050	0.1378
50	150.56	5.184	0.1521
60	9.394	1.328	0.0075

Error Analysis The table 4 shows an interesting pattern - the error first increases slightly at degree 50 before dropping sharply at degree 60. This

occurs because: At degree 50, the polynomial struggles to match both the rapid oscillations and the decay and degree 60, it resolves both features well

This method is particularly useful when you need high accuracy and can afford to compute higher degree polynomials. The results confirm that Chebyshev polynomials are an excellent choice for approximating oscillatory functions with exponential decay.

4 Conclusions

4.1 Method-Specific Analysis

Through systematic evaluation, each approximation method demonstrated distinct characteristics and performance profiles.

The class of monomial, Lagrange, and Newton interpolation methods provides theoretical exact interpolation at nodes with relatively simple implementation. However, these methods exhibit severe Runge phenomenon when using equidistant nodes, with maximum errors approaching 1.9 for polynomial degree $n = 50$. Furthermore, the condition number of the monomial basis grows as $\mathcal{O}(10^{2n})$, imposing significant numerical limitations. These methods are most appropriate for low-degree interpolation ($n < 15$) employing Chebyshev node distributions.

Cubic spline interpolation demonstrates stable convergence characteristics, with mean squared error decreasing from 0.433 to 0.011 as node density increases from 5 to 40 nodes. The piecewise polynomial structure effectively avoids global oscillation issues inherent in high-degree polynomial interpolation. Nevertheless, this approach requires relatively dense node distributions (> 40 nodes) to achieve high-frequency resolution with MSE < 0.01, and is fundamentally limited to C^2 continuity at knot points. This method is particularly suitable for applications requiring guaranteed smoothness without derivative information.

Hermite interpolation using Newton's Divided Difference Approach provides superior derivative matching capabilities, achieving maximum error of 0.39 with 40 Chebyshev nodes, representing an 18% error reduction compared to standard polynomial interpolation. The Newton formulation offers computational advantages through its recursive divided difference structure, enabling efficient polynomial construction while maintaining numerical stability. The principal limitations include the requirement for derivative information and persistent endpoint oscillations, with boundary errors approximately 2.3 times larger than interior point errors. This method finds optimal application in boundary value problems where derivative data is available.

Discrete least squares approximation achieves exceptional accuracy at high polynomial degrees, with MSE reaching 4.13×10^{-5} for $n = 60$, while maintaining robustness to node selection when employing Chebyshev points. The primary limitations involve ill-conditioned normal equations for degrees exceeding 60 and the necessity for careful degree selection to avoid overfitting. This approach is recommended for high-accuracy applications where computational cost considerations are secondary.

Continuous least squares approximation utilizing Legendre or Chebyshev polynomial bases offers spectral convergence for smooth functions through numerically stable orthogonal polynomial expansions. The computational costs associated with inner product integration and implementation complexities introduced by the Chebyshev weight function represent significant practical limitations. These methods are ideally suited for theoretical analysis and high-precision applications where integration costs are acceptable.

4.2 Synthesis of Findings

The comparative analysis reveals three fundamental trade-offs governing method selection for oscillatory-decay function approximation. First, a clear accuracy-stability trade-off exists, wherein high-degree polynomials offer superior approximation accuracy but require specialized node distributions like Chebyshev points to maintain numerical stability. Second, an information-complexity trade-off emerges, with methods utilizing derivative information (such as Hermite interpolation) providing performance advantages at the cost of increased implementation complexity and data requirements. Third, a global-local methodology trade-off distinguishes piecewise approaches (like splines) that provide robustness through localization from global methods that achieve spectral convergence but may exhibit boundary effects.

4.3 Practical Recommendations

Based on the comprehensive numerical experiments conducted in this study, specific methodological recommendations can be formulated for common application scenarios. For rapid visualization purposes, cubic splines with 20-30 nodes provide an effective balance between computational efficiency and visual accuracy. Applications requiring high-frequency resolution benefit from discrete least squares approximation with polynomial degrees between 50-60 utilizing Chebyshev nodes. For problems involving derivative-sensitive applications, Hermite interpolation with approximately 40 Chebyshev nodes delivers optimal performance. Theoretical analysis and high-precision applications are best served by continuous least squares approximation with

Legendre polynomial bases.

Future research should address the persistent boundary effects observed across multiple methods through the development of rational approximation techniques and adaptive strategies capable of handling non-uniform oscillation patterns characteristic of oscillatory-decay functions in scientific applications.

References

[1] P. E. Gill, W. Murray and M. H. Wright, *Practical Optimization*, Academic Press Inc. London, 1981.

[2] O. Axelsson and M. Neytcheva, *Preconditioning methods for linear systems arising in constrained optimization problems*, Numer. Linear Algebra Appl., 10 (2003), pp. 3-31.