Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2025 (43)** : 1–13.
ISSN-0037-8712
doi:10.5269/bspm.68737

On some examples and counterexamples of the Runge phenomena and Chebyshev nodes

AMINE LAGHRIB, YASMINE EL MOUBARIKI, MOUNIA ALAOUI and HANAA MTAI

ABSTRACT: Runge's phenomenon appears in the interpolation process using uniformly placed nodes for some smooth function. An effective solution to this problem is the use of Chebychev roots as interpolation nodes. In this paper, we discuss the efficiency of these nodes and give many illustrated counterexamples of some differentiable and non-differentiable functions to show the limitation of the used Chebychev roots as nodes for Lagrange interpolation.

Key Words: Runge Phenomenon, Interpolation, counterexamples, differential function.

Contents

1. Introduction

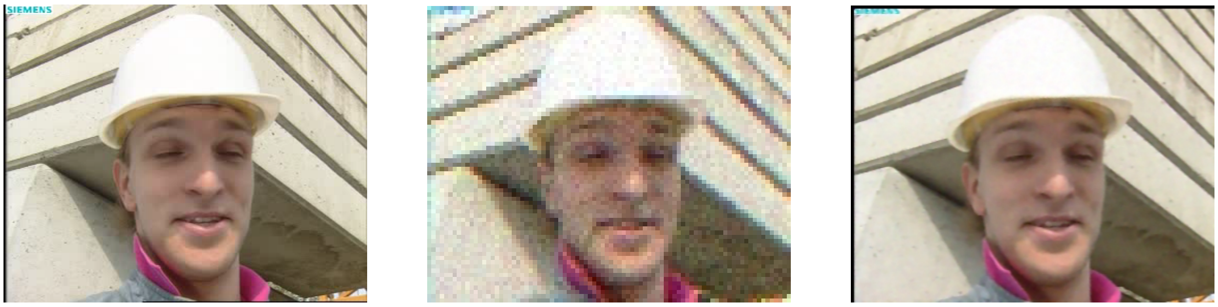
Faber's theorem is recognized as a fundamental result in numerical analysis, which highlights the non-convergence of polynomial interpolation for certain classes of functions, even when these functions are smooth. It is widely discussed in numerical analysis literature and provides key insights into the behavior of polynomial interpolations, especially regarding the convergence of the interpolated polynomial to the actual function. In essence, Faber's theorem illustrates that polynomial interpolation may not always converge to the true function, even if the function is regular (i.e., smooth), under certain conditions.

The counterintuitive nature of Faber's theorem lies in the fact that, in many practical cases, even for smooth functions, interpolation may still fail to converge. This phenomenon is particularly puzzling, as one would typically expect smooth functions to be well-approximated by polynomials. However, Faber's theorem shows that this expectation does not always hold. Importantly, the theorem's implications are not limited to smooth functions; it also applies to functions that are merely continuous or even just Lipschitz continuous. In these cases, the regularity conditions that would normally ensure convergence of the interpolation are no longer sufficient to guarantee convergence to the true function.

A well-known example of this behavior can be observed when interpolating the function $f(x) = \frac{1}{1+x^2}$. While this function is smooth and continuous, when we apply polynomial interpolation using equally spaced nodes, we observe the emergence of the Runge phenomenon. The Runge phenomenon refers to the drastic oscillations that appear at the edges of the interpolation interval as the number of interpolation points increases, leading to poor approximation near the boundaries.

In contrast, Chebyshev nodes have been shown to effectively mitigate the Runge phenomenon. The Chebyshev polynomial interpolation is known for its ability to avoid the large oscillations at the endpoints of the interval by placing interpolation points more densely near the boundaries, where interpolation errors tend to be larger. By using Chebyshev nodes, one can ensure that the sequence of interpolation polynomials converges uniformly to the true function, provided the function is sufficiently smooth.

2010 *Mathematics Subject Classification:* 35B40, 35L70.
Submitted June 29, 2023. Published September 01, 2025



Original image LR image Resized

Figure 1: The *Worker* image resizing using the Chebyshev interpolation.

However, despite the advantages of Chebyshev nodes, there are cases where their use does not guarantee uniform convergence. In this paper, we explore the ability of Chebyshev nodes to circumvent the Runge phenomenon and also present a counterexample where the choice of Chebyshev nodes is insufficient to achieve uniform convergence. This counterexample highlights the limitations of Chebyshev interpolation in specific scenarios, offering further insight into the conditions under which polynomial interpolation may fail to converge, even with Chebyshev nodes. In contrast, for non-differentiable functions, Chebyshev nodes can still yield satisfactory results, particularly in applications such as image resizing. This is notable because, in these cases, the conditions typically required for convergence, such as differentiability—are not satisfied. Despite the lack of smoothness or differentiability in the function, Chebyshev nodes still provide a robust interpolation strategy. This is especially useful in practical applications like image resizing, where the goal is often to approximate pixel values at non-integer locations, and the functions being interpolated may not be differentiable or smooth in the traditional sense.

The ability of Chebyshev nodes to perform well in these scenarios is a testament to their strength in reducing edge effects and improving the accuracy of interpolation, even when the standard convergence theorems do not apply. By concentrating interpolation points more densely near the boundaries of the interval, Chebyshev nodes mitigate the common pitfalls of polynomial interpolation, such as the Runge phenomenon. In the context of image processing [2,3,4], this translates to more accurate and visually pleasing results when resizing images, even when the underlying function, such as pixel intensities or color values does not exhibit the smoothness required by traditional interpolation theory. For example, consider the image resizing of the *Worker* image using Chebyshev nodes, as shown in Figure 1. This figure demonstrates how Chebyshev nodes are applied to resize the image while preserving key features and reducing common interpolation artifacts. Even though the function representing the pixel values is not necessarily smooth or differentiable, the Chebyshev interpolation effectively handles the resizing task, yielding a result that is visually accurate and free from the usual edge effects associated with other interpolation methods. This highlights the practical utility of Chebyshev nodes in non-differentiable interpolation tasks, particularly in image processing applications.

2. Tchebychev polynomial

The polynomial interpolation of Chebyshev, denoted as $P_n(x)$, for the function f at the points x_k, can be expressed in the following form:

$$P_n(x) = \sum_{i=0}^{n} c_i T_i(x) \tag{2.1}$$

The coefficients c_i are determined by the following theorem:

Theorem 1 *The coefficients c_i in formula (2.1) are explicitly given by:*

$$c_i = \frac{2}{n+1} \sum_{k=1}^{n+1} f(x_k) T_i(x_k),$$

where

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

Proof: If we assume that $f(x)$ is equal to $P_n(x)$ at the points x_k, then we obtain:

$$\begin{aligned} f(x_k) &= \sum_{i=0}^{n} c_i T_i(x_k) && (2.2)\\ &= \frac{1}{2}c_0T_0(x_k) + c_1T_1(x_k) + c_2T_2(x_k) + \cdots && (2.3) \end{aligned}$$

By multiplying equation (2.2) by $\frac{2}{n+1}T_j(x_k)$ and taking the sum, we obtain:

$$\frac{2}{n+1}\sum_{k=1}^{n+1} f(x_k)T_j(x_k) = \sum_{i=0}^{n} c_i \frac{2}{n+1}\sum_{k=1}^{n+1} T_i(x_k)T_j(x_k),$$

According to the formula of orthogonality, we have:

$$\sum_{k=1}^{n+1} T_i(x_k)T_j(x_k) = \begin{cases} 0 & i \neq j(\leq n) \\ n+1 & i = j = 0 \\ \frac{1}{2}(n+1) & 0 < i = j \leq n \end{cases}$$

We obtain finally that

$$c_j = \frac{2}{n+1}\sum_{k=1}^{n+1} f(x_k)T_j(x_k)$$

□

We consider the function f such as:

$$f(x) = \sin(\frac{\pi}{2}x).$$

We look-for the interpolate polynomial of Tchebychev $P_2(x)$ of the function f in nodes $\{\frac{\sqrt{3}}{2}, -\frac{\sqrt{3}}{2}, 0\}$. Let the polynomial $P_n(x)$ such that:

$$\begin{aligned} P_n(x) &= \sum_{i=0}^{n} c_i T_i(x) \\ &= \frac{1}{2}c_0T_0(x) + \sum_{i=1}^{n} c_i T_i(x) \end{aligned}$$

with

$$c_i = \frac{2}{n+1}\sum_{k=1}^{n+1} f(x_k)T_i(x_k)$$

such as x_k sare the polynomial zeroes of $T_{n+1}(x)$ given by

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

and we know that $x_1 = \frac{\sqrt{3}}{2}$, $x_2 = -\frac{\sqrt{3}}{2}$, and $x_3 = 0$ are the zeros of the Chebyshev polynomial $T_3(x)$. Therefore, the Chebyshev interpolation polynomial of the function f at the points $\{x_1, x_2, x_3\}$ is given by:

$$P_2(x) = \sum_{i=0}^{2} c_i T_i(x) \quad (2.4)$$

$$= \frac{c_0}{2} T_0(x) + \sum_{i=1}^{3} c_i T_i(x)$$

with

$$c_i = \frac{2}{3} \sum_{k=1}^{3} f(x_k) T_i(x_k)$$

For $i = 0$, we have:

$$\begin{aligned} c_0 &= \frac{2}{3} [f(x_1)T_0(x_1) + f(x_2)T_0(x_2) + f(x_3)T_0(x_3)] \\ &= \frac{2}{3} \left[\sin(\frac{\pi}{4}\sqrt{3}) + \sin(-\frac{\pi}{4}\sqrt{3}) + \sin(0)\right] \\ &= 0, \end{aligned}$$

For $i = 1$, we obtain:

$$\begin{aligned} c_1 &= \frac{2}{3} [f(x_1)T_1(x_1) + f(x_2)T_1(x_2) + f(x_3)T_1(x_3)] \\ &= \frac{2}{3} \left[\sin(\frac{\pi}{4}\sqrt{3})x_1 + \sin(-\frac{\pi}{4}\sqrt{3})x_2 + \sin(0)x_3\right] \\ &= \frac{2}{3} \left[\sin(\frac{\pi}{4}\sqrt{3})(\frac{\sqrt{3}}{2}) + \sin(-\frac{\pi}{4}\sqrt{3})(-\frac{\sqrt{3}}{2})\right] \\ &= \frac{2\sqrt{3}}{3} \sin(\frac{\pi}{4}\sqrt{3}), \end{aligned}$$

For $i = 2$, we can have:

$$\begin{aligned} c_2 &= \frac{2}{3} \left[\sin(\frac{\pi}{4}\sqrt{3})(2x_1^2 - 1) + \sin(-\frac{\pi}{4}\sqrt{3})(2x_2^2 - 1)\right] \\ &= \frac{2}{3} \left[\sin(\frac{\pi}{4}\sqrt{3})(2(\frac{\sqrt{3}}{2})^2 - 1) + \sin(-\frac{\pi}{4}\sqrt{3})(2(-\frac{\sqrt{3}}{2})^2 - 1)\right] \\ &= 0 \end{aligned}$$

We substitute c_1, c_2, c_3 in (2.4), we obtain

$$\begin{aligned} P_2(x) &= c_1 x \\ &= \frac{2\sqrt{3}}{3} \sin(\frac{\pi}{4}\sqrt{3})x. \end{aligned}$$

3. Main result

We start by announcing the following theorem, which give the expression of the Lagrange interpolation process of a smooth function f.

Theorem 2 *(Lagrange interpolation error)*
Let $f \in \mathcal{C}^{n+1}([a,b])$ and $a \leq x_0 < x_1 < x_2 < \ldots < x_n \leq b$ the interpolation nodes. Let P the Lagrange polynomial defined by

$$P(x_s) = f(x_s) \text{ for } s = 0, 1, \ldots, n.$$

Then the Lagrange polynomial error is then given by:

$$f(x) - P(x) = \frac{Q(x)}{(n+1)!} f^{(n+1)}(\eta) \tag{3.1}$$

where $Q(x) = \prod_{i=0}^{n} (x - x_i), a \leq \min(x_0, x) < \eta < \max(x, x_n) \leq b$.

We can see that the error depends on the regularity of the function f and also on the nodes x_i figured in $Q(x)$. In fact, the polynomial $Q(x) = (x - x_0)(x - x_1)\dots(x- \ x_{n+1})$ is of degree $(n+1)$ with dominant coefficient of x^{n+1} is 1. For a better choice of the nodes $\{x_i\}_{i=0,n}$ it is then convenient to take the roots of the polynomial $Q(x)$ such as:

$$\max_{x\in[a,b]} |Q(x)| \leq \max_{x\in[a,b]} |S(x)|, \quad \forall S \in \mathbb{P}_{n+1} \text{ and } S(x) = x^{n+1} + a_n x^n + \dots$$

One of the successful polynomial choice is the Chebyshev polynomials T_n (see [9] for more general proprieties of these polynomial) defined such as:

$$T_n(x) = \cos(n \arccos(x)), \quad x \in [-1,1], n \geq 0.$$

The roots of the polynomial T_n verifies : $T_n(x) = \cos(n \arccos(x)) = 0$, then we have $n\ arccos(x) = \frac{\pi}{2} + k\pi$, which implies that $\arccos(x) = \frac{\pi}{2n} + \frac{k\pi}{n}$ for $k = 0 \cdots n-1$. This conduct to the following roots of T_n:

$$x_k = \cos\left(\frac{2k+1}{2n}\pi\right); k = 0 \cdots n-1. \tag{3.2}$$

Proposition 1 1. *The Chebyshev polynomial $T_{n+1}(x)$ satisfies the condition that*

$$\max_{x\in[-1,1]} |T_{n+1}(x)| = 1.$$

Consequently, we have

$$\max_{x\in[-1,1]} |(x-x_0)(x-x_1)\cdots(x-x_n)| = \frac{1}{2^n}.$$

This leads to the following error estimate for the Chebyshev polynomial interpolation:

$$|f(x) - P(x)| \leq \frac{1}{(n+1)!}\frac{1}{2^n}\max_{x\in[-1,1]}\left|f^{(n+1)}(x)\right|.$$

2. *The roots of the Chebyshev polynomials are more densely distributed near the endpoints of the interval $[a,b]$. This specific distribution helps mitigate the Runge phenomenon, particularly reducing the edge effects that typically occur in polynomial interpolation near the boundaries of the interval [8].*

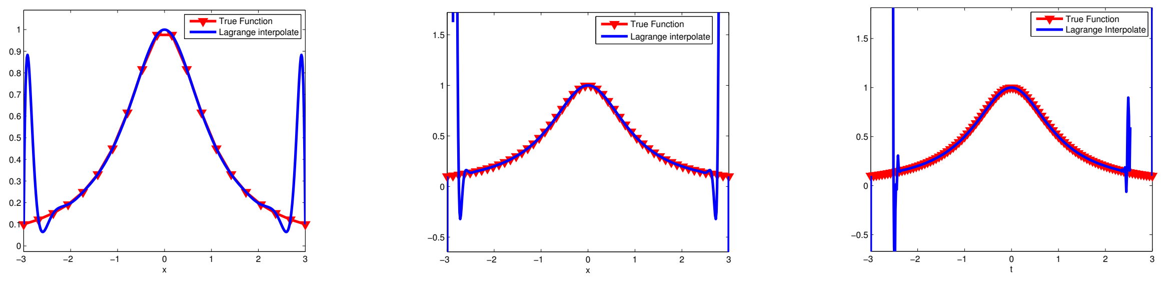
Proof: Let $(n+1)$ the points for the interpolation polynomial $P(x)$ are chosen especially at the roots of the Chebyshev polynomials $x_k = \cos\left[\frac{(2k+1)\pi}{2n}\right]$, then by the usual interpolation error, we have

$$\begin{aligned} f(x) - P(x) &= \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)\cdots(x-x_n) \\ &= \frac{f^{(n+1)}(\xi)}{(n+1)!}\frac{T_{n+1}(x)}{2^n} \end{aligned}$$

Then by introducing the absolute value in both sides we obtain

$$\begin{aligned} |f(x) - P(x)| &= \left|\frac{f^{(n+1)}(\xi)}{(n+1)!}\frac{T_{n+1}(x)}{2^n}\right| \\ &\leq \frac{\left\|f^{(n+1)}\right\|_\infty}{(n+1)!}\frac{|T_{n+1}(x)|}{2^n} \\ &\leq \frac{\left\|f^{(n+1)}\right\|_\infty}{2^n(n+1)!} \end{aligned}$$

Which concludes the proof of relation 1. For the second point it is consequence of the first point since the error decreases exponentially with the number of nodes n. □



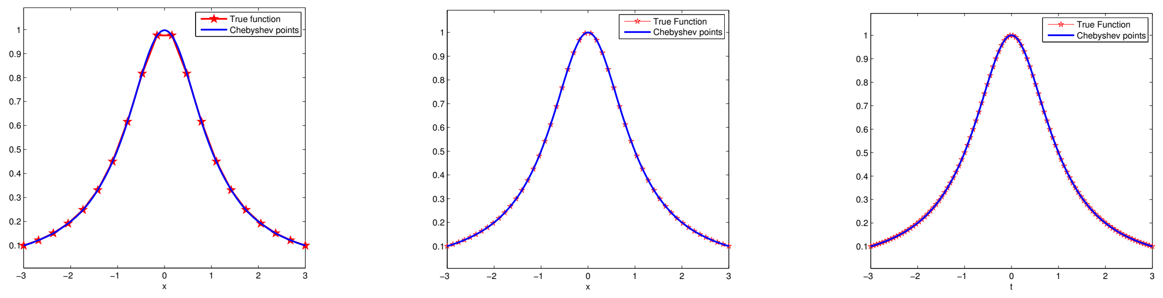
20 interpolation points 50 interpolation points 100 interpolation points

Figure 2: The evolution of the Lagrange approximate polynomials with respect to the number of nodes n. The approximate function is $f(x) = 1/(1+x^2)$ using uniformly placed nodes.

3.1. Numerical Examples

The famous example due to Carle Runge is known in the interpolation process. Indeed, by interpolating the function $f(x) = 1/(1+x^2)$ over the interval $[-3, 3]$ with uniformly spaced nodes (the space between the nodes is fixed by $h = \dfrac{b-a}{n}$), the sequence of interpolating Lagrange polynomials diverges (the error tends to ∞). This can be confirmed by adding more interpolation points, which makes the approximation not appropriate. See Figure 1 to look the behavior of the Lagrange interpolate polynomial with respect to evenly spaced nodes number. The error near the bounds of the interval is outstanding, while the approximation looks much better in the interior of the interval.

The Runge phenomenon can be avoided using the Chebyshev nodes. Moreover, the use of large nodes can enhance the approximation. In fact, we can obtain the convergence to the function f earlier using only 20 points as shown in Figure 2. These plots show how interpolating at the roots of T_{20} can efficiently eliminates the bad behavior at the extremities of the interval $[-3, 3]$.



20 interpolation points 50 interpolation points 100 interpolation points

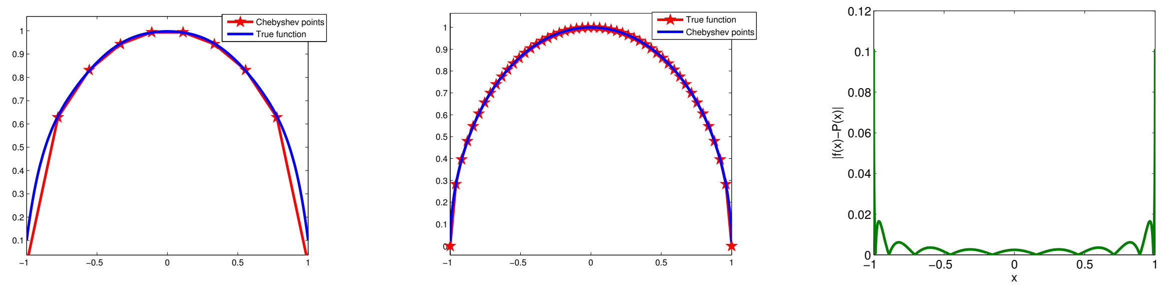
Figure 3: The evolution of the Lagrange approximate polynomials with respect to the number of nodes n. The approximate function is $f(x) = 1/(1+x^2)$ using roots of Chebyshev polynomials as nodes.

3.2. Examples of non-differentiable functions

Taking into account the Proposition 1, the interpolate function f using the Chebyshev nodes must be smooth enough ($f \in \mathcal{C}^{n+1}([a, b])$). Effectively, for non-differentiable function, the Chebyshev nodes fails to give a better approximation to the function f. To illustrate this limitation, we Consider the function $f(x) = \sqrt{1-x^2}$ on the interval $[-1, 1]$. This function represents a semicircle centered at the origin with a radius of 1. Since the function has a sharp corner at $x = \pm 1$, it is not differentiable at those points. Let's attempt to interpolate this function using Chebyshev polynomial interpolation. We'll choose a relatively low degree for simplicity, such as $n = 4$ and we will use $n = 10$ and $n = 20$ in the numerical approximation. The Chebyshev nodes for $n = 4$ are: $x_0 = \cos(\frac{\pi}{8}) \approx 0.9239$ $x_1 = \cos(\frac{3\pi}{8}) \approx 0.3827$ $x_2 = \cos(\frac{5\pi}{8}) \approx -0.3827$ $x_3 = \cos(\frac{7\pi}{8}) \approx -0.9239$ Now, let's compute the Chebyshev interpolating polynomial $P_n(x)$ of degree $n = 4$ using these nodes. We have:

$$P_4(x) = c_0T_0(x) + c_1T_1(x) + c_2T_2(x) + c_3T_3(x) + c_4T_4(x), \tag{3.3}$$

where $T_i(x)$ represents the Chebyshev polynomial of degree i and c_i are the coefficients determined by the function values at the nodes. The function values at the Chebyshev nodes are: $f(x_0) \approx 0.3827$ $f(x_1) \approx 0.9239$ $f(x_2) \approx 0.9239$ $f(x_3) \approx 0.3827$ By computing the interpolation polynomial using these function values, we obtain: $P_4(x) = 0.6545x^4 - 0.8314x^2 + 0.3660$. If we compare this polynomial to the function $f(x) = \sqrt{1-x^2}$, we can see that it fails to accurately represent the sharp corner at $x = \pm 1$. See Figure 3 for such comparison. It seems that for $n = 50$, the approximation is adequate, however, when we plot the absolute error $|f(x) - P(x)|$ between the function f and P the error increases near the nodes -1 and 1, where the function in non-differentiable. The interpolation polynomial smooths



20 interpolation points　　50 interpolation points　　Absolute error of interpolation

Figure 4: The evolution of the Lagrange approximate polynomials with respect to the number of nodes n. The approximate function is $f(x) = \sqrt{1-x^2}$ using roots of Chebyshev polynomials as nodes.

out the corner, resulting in a polynomial that does not converge to $f(x)$ as x approaches ± 1. This failure to converge is a consequence of the non-differentiability of the function at the interpolation points. Chebyshev polynomial interpolation assumes certain smoothness properties of the function, and when those properties are violated, the convergence of the interpolation process is compromised. Therefore, in this demonstration, Chebyshev polynomial interpolation fails to accurately approximate the non-differentiable function $f(x) = \sqrt{1-x^2}$. The interpolation polynomial cannot capture the sharp corner of the function, highlighting the limitations of Chebyshev interpolation in such cases.

If the function has a rough allure, we can detect the so-called Gibbs phenomena, which is very similar to the Runge phenomena detected previously. We consider another example of the interpolating problem of the indicator function δ over the interval $[-1, 1]$ at 20, 50 and 100 Chebyshev nodes. The function δ is defined by:

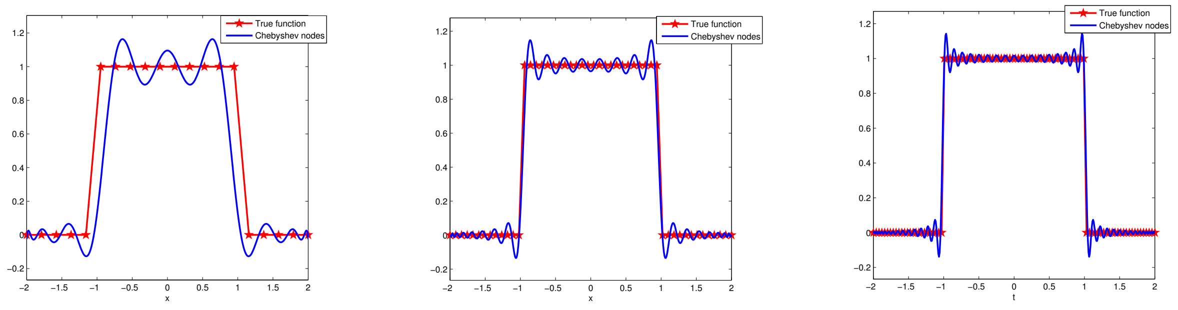
$$\delta(x) := \begin{cases} 1 & \text{if } x \in [-1, 1] \ , \\ 0 & \text{if } x \notin [-1, 1] \ . \end{cases}.$$

Figure 5 illustrates the Lagrange polynomial interpolation using three different numbers of interpolation points. As observed, even with a larger number of points, the Gibbs phenomenon persists, and the error between the true function and its approximation remains significant.

This demonstrates once again that Chebyshev polynomial interpolation fails to accurately approximate the non-differentiable function δ. The Chebyshev interpolation polynomial cannot capture the sharp corner of the function, as guaranteed convergence requires sufficient differentiability of the function being interpolated.

3.3. Numerical Counterexamples

In this subsection, we provide an example illustrating that the choice of Chebyshev polynomial nodes may not always be sufficient to ensure the convergence of polynomial interpolation, even when the function



20 interpolation points   50 interpolation points   100 interpolation points

Figure 5: The evolution of the Lagrange approximate polynomials with respect to the number of nodes n. The approximate function is the indicator function δ using roots of Chebyshev polynomials as nodes.

f is smooth enough.

To begin with, consider the function $f(x) = \sqrt{x}$ over the interval $[0, 1]$. This function is smooth and differentiable over the interval $(0, 1]$, but we will show that even with Chebyshev polynomial interpolation, the convergence may fail. The purpose of Chebyshev polynomial interpolation is to approximate the function $f(x)$ using a polynomial $P_n(x)$ of degree n that passes through the Chebyshev nodes. The Chebyshev nodes are specifically chosen to reduce the impact of Runge's phenomenon, but they do not guarantee perfect convergence for all smooth functions.

The Chebyshev nodes for a polynomial of degree n are given by the following formula:

$$x_i = \cos\left(\frac{(2i+1)\pi}{2n+2}\right), \quad \text{for} \quad i = 0, 1, 2, \ldots, n.$$

These nodes are distributed more densely near the endpoints of the interval $[-1, 1]$, which can improve the approximation properties of the polynomial. Now, let us consider the case where $n = 4$. The Chebyshev nodes for $n = 4$ are calculated as:

$$x_0 = \cos\left(\frac{\pi}{10}\right) \approx 0.9511, \quad x_1 = \cos\left(\frac{3\pi}{10}\right) \approx 0.5878, \quad x_2 = \cos\left(\frac{5\pi}{10}\right) = 0,$$

and

$$x_3 = \cos\left(\frac{7\pi}{10}\right) \approx -0.5878, \quad x_4 = \cos\left(\frac{9\pi}{10}\right) \approx -0.9511.$$

Thus, we have five Chebyshev nodes: x_0, x_1, x_2, x_3, x_4 for this example. These nodes are placed on the interval $[-1, 1]$, and we are interested in approximating the function $f(x) = \sqrt{x}$ on the sub-interval $[0, 1]$.

Now, we can compute the Chebyshev interpolating polynomial $P_4(x)$ of degree $n = 4$ using the Chebyshev nodes and the values of the function $f(x)$ at these nodes. The Chebyshev interpolating polynomial is a linear combination of Chebyshev polynomials $T_i(x)$ of degree i, and it can be written as:

$$P_4(x) = c_0T_0(x) + c_1T_1(x) + c_2T_2(x) + c_3T_3(x) + c_4T_4(x),$$

where $T_i(x)$ are the Chebyshev polynomials of the first kind, and the coefficients c_i are determined by the values of the function $f(x)$ at the nodes. To compute the coefficients c_i, we evaluate the function $f(x) = \sqrt{x}$ at each Chebyshev node:

$$f(x_0) = \sqrt{0.9511} \approx 0.9752, \quad f(x_1) = \sqrt{0.5878} \approx 0.7661, \quad f(x_2) = \sqrt{0} = 0,$$

and

$$f(x_3) = \sqrt{0.5878} \approx 0.7661, \quad f(x_4) = \sqrt{0.9511} \approx 0.9752.$$

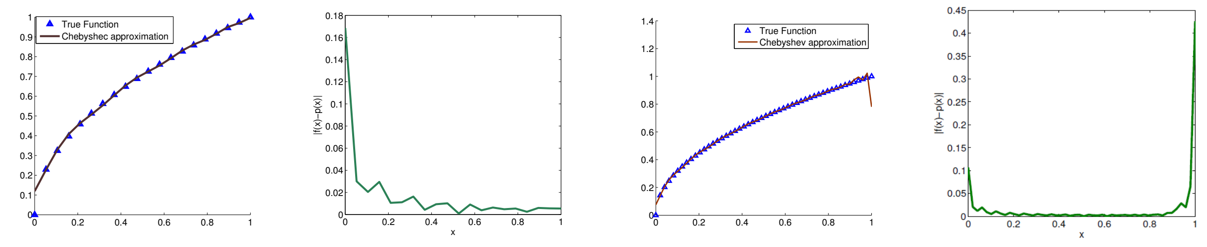
Thus, we have the following set of function values:

$$f(x_0) \approx 0.9752, \quad f(x_1) \approx 0.7661, \quad f(x_2) = 0, \quad f(x_3) \approx 0.7661, \quad f(x_4) \approx 0.9752.$$

With these function values, we can now compute the Chebyshev interpolation polynomial $P_4(x)$. The polynomial is constructed using the Chebyshev polynomials $T_0(x), T_1(x), T_2(x), T_3(x), T_4(x)$, which are standard polynomials defined in terms of cosines. For example, $T_0(x) = 1$, $T_1(x) = x$, and so on. The coefficients c_0, c_1, c_2, c_3, c_4 are obtained by solving a system of linear equations derived from the interpolation conditions.

To better illustrate the interpolation, we also compute the interpolating polynomial for larger numbers of nodes, such as $n = 20$ and $n = 50$. By plotting the original function $f(x) = \sqrt{x}$ along with its Chebyshev interpolating polynomials for these values of n, we can visually assess the performance of the interpolation. The corresponding plots are shown in Figure , where we observe the behavior of the interpolated function as the number of interpolation points increases.

In summary, this example demonstrates that even though the function $f(x) = \sqrt{x}$ is smooth and differentiable, the interpolation using Chebyshev polynomials does not always converge, particularly when the function has a singularity at one of the interpolation points. The failure of convergence is more noticeable as the number of interpolation points increases, despite the function being smooth over the interval.

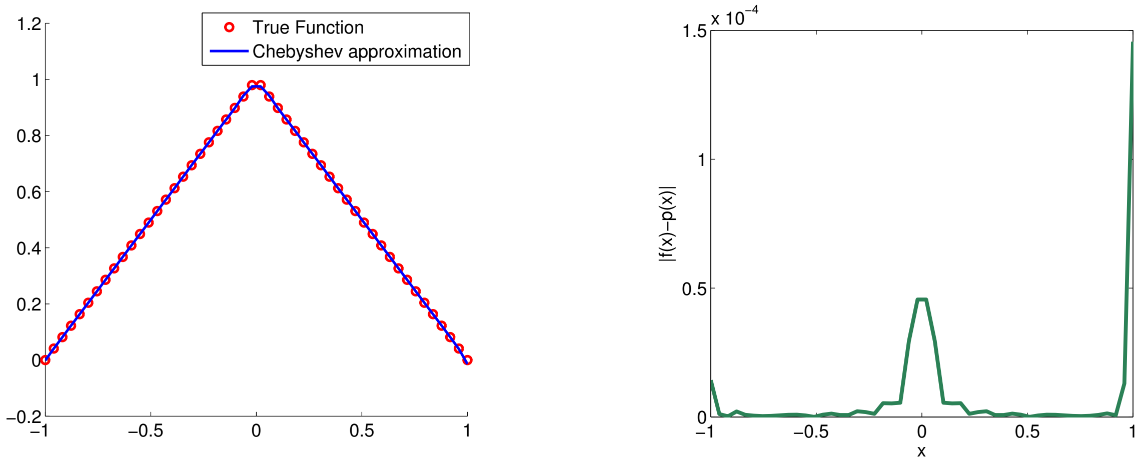


20 interpolation points Absolute error 50 interpolation points Absolute error

Figure 6: The evolution of the Chebyshev approximate polynomials with respect to the number of nodes n and the associated errors. The approximate function is the smooth function $\sqrt{x}$ using roots of Chebyshev polynomials as nodes.

If we compare this polynomial to the function $f(x) = \sqrt{x}$, we can observe that Chebyshev polynomial interpolation fails to accurately approximate the function when the number of nodes increases, which not endure the convergence. The interpolation polynomial deviates from the true function, resulting in a poor approximation. This demonstrates a case where Chebyshev polynomial interpolation fails to converge even when the function being interpolated is differentiable. The failure occurs due to the specific arrangement of the interpolation nodes and the inherent limitations of polynomial approximation methods.

This next example illustrated a non-differentiable function where the Chebyshev interpolation works. Let's consider the function $f(x) = 1 - |x|$ on the interval $[-1, 1]$. By considering a specific degree for the interpolating polynomial, such as $n = 4$. We compute the Chebyshev nodes x_i for $n = 4$ using the formula $x_i = \cos\left(\frac{(2i+1)\pi}{2(n+1)}\right)$. The Chebyshev nodes for $n = 4$ are: $x_0 = \cos\left(\frac{\pi}{10}\right) \approx 0.9511$, $x_1 = \cos\left(\frac{3\pi}{10}\right) \approx 0.5878$, $x_2 = \cos\left(\frac{5\pi}{10}\right) = 0$, $x_3 = \cos\left(\frac{7\pi}{10}\right) \approx -0.5878$, $x_4 = \cos\left(\frac{9\pi}{10}\right) \approx -0.9511$. We have then the function values $f(x_i)$ at the Chebyshev nodes. For example: $f(x_0) = 1 - |0.9511| = 1 - 0.9511 = 0.0489$, $f(x_1) = 1 - |0.5878| = 1 - 0.5878 = 0.4122$, $f(x_2) = 1 - |0| = 1$, $f(x_3) = 1 - |-0.5878| = 1 - 0.5878 = 0.4122$, $f(x_4) = 1 - |-0.9511| = 1 - 0.9511 = 0.0489$. We compute the Chebyshev interpolating polynomial $P_n(x)$ of degree $n = 50$ using the function values at the nodes and we plot this approximation in Figure . By comparing the interpolation results of Chebyshev polynomial interpolation and the original function $f(x)$, we will likely observe that the Chebyshev polynomial interpolation provides an accurate approximation of the function $f(x) = 1 - |x|$ near the corner at $x = 0$ even if the function is not-differentiable. Chebyshev nodes are specifically designed to minimize oscillations and provide better convergence for non-differentiable functions with sharp features too. This example showcases how Chebyshev polynomial interpolation can provide more accurate results for non-differentiable functions with sharp corners, such as $f(x) = 1 - |x|$.



50 interpolation points

Absolute error using $n = 50$

Figure 7: The approximate function $1 - |x|$ using roots of Chebyshev polynomials as nodes.

The last example is an image processing application (resizing). To resize the image, we map the coordinates of the original grid $[0, m - 1]$ to the new grid $[0, m' - 1]$ using the scaling factor:

$$x' = \frac{m'}{m} \cdot x, \quad y' = \frac{n'}{n} \cdot y$$

For example, resizing an image from $m = 100 \times n = 100$ to $m' = 200 \times n' = 200$. After determining the mapping of coordinates, Chebyshev interpolation is performed along both the horizontal and vertical directions. For each coordinate, the Lagrange basis polynomial $L_i(x)$ is used:

$$P_n(x) = \sum_{i=0}^{n-1} f(x_i) L_i(x)$$

where $f(x_i)$ are the values of the function at the Chebyshev nodes and $L_i(x)$ are the Lagrange basis polynomials. The interpolation is performed in two steps:

- Horizontal Interpolation: Interpolate the image along the rows using Chebyshev interpolation for the horizontal direction.
- Vertical Interpolation: After horizontal interpolation, interpolate along the columns using Chebyshev interpolation for the vertical direction.

In the following, we resize four real images with various sizes using Chebyshev interpolation and compare them to bicubic interpolation. Below, we show the resized image after applying Chebyshev interpolation with $m = 100$, $n = 100$, and resizing to $m' = 200$, $n' = 200$. The images in Figure 8 illustrate the result obtained after applying Chebyshev interpolation and also bicubic for resizing. In these cases, we aimed for a resolution of four times the initial pixels. Chebyshev interpolation, by its design, provides a smoother and less oscillatory result compared to other interpolation methods like bicubic interpolation. The resized images are compared, demonstrating how Chebyshev interpolation results in a smoother resizing compared to other method. The Chebyshev nodes ensure that edge effects are minimized, and the resizing is uniform and accurate. The use of Chebyshev interpolation for image resizing provides an efficient and accurate method to avoid common interpolation issues such as the Runge phenomenon. By

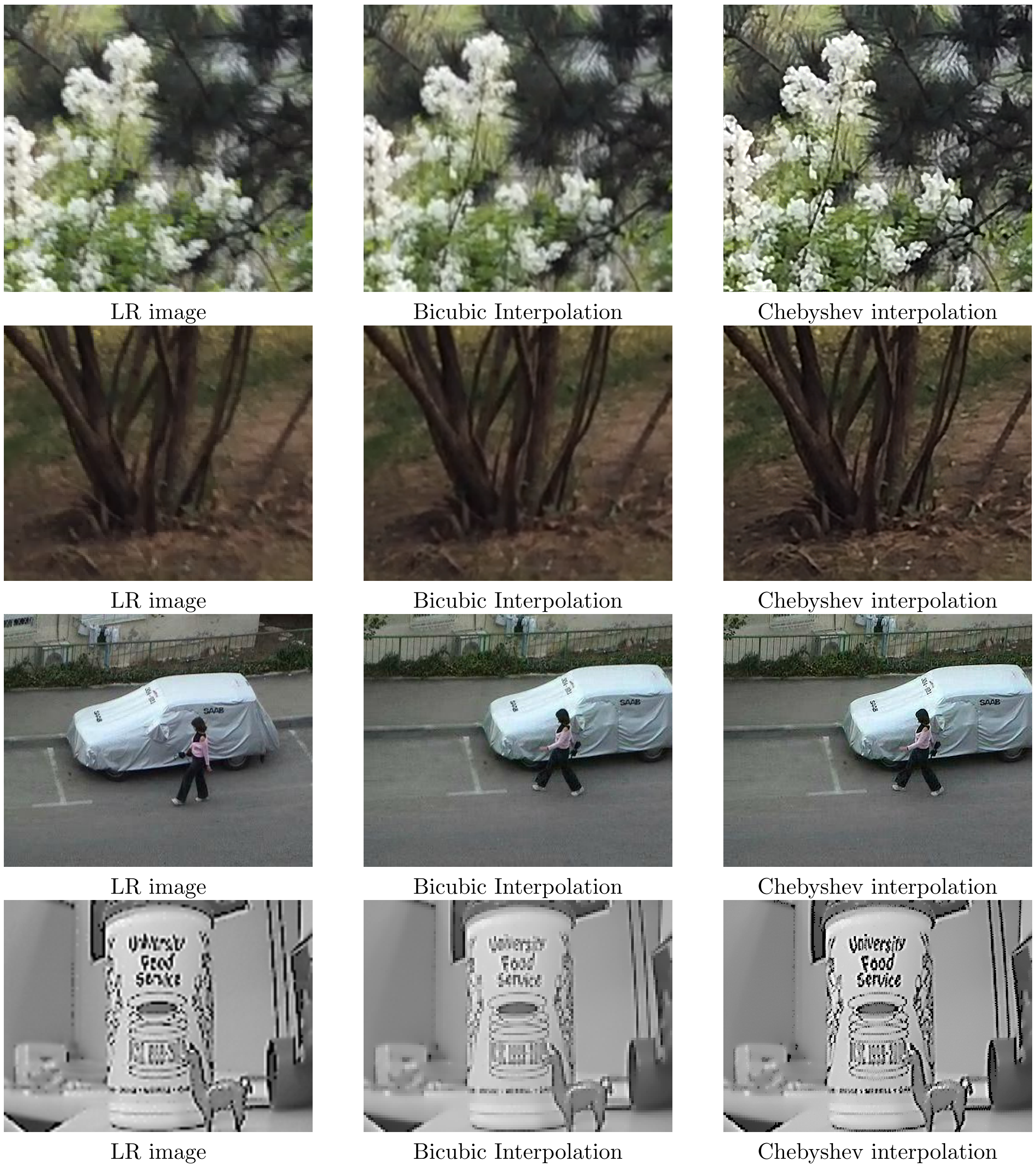


Figure 8: Resized four selected real images using Chebyshev interpolation and compared to bicubic interpolation.

using Chebyshev nodes, we achieve a smoother resizing process, especially near the boundaries of the image. This method can be applied effectively in practical scenarios requiring high-quality image resizing.

However, when the decimation factor is high and compared to bicubic interpolation, which is well-known for its high-quality results in terms of preserving sharp edges and minimizing the smoothing of image details, the Chebyshev interpolation can sometimes lead to slightly less sharp features in regions with high contrast, see Figure 9 for some illustration. This happens because Chebyshev interpolation uses the roots of the Chebyshev polynomials to sample data points, which, although more evenly distributed, can still cause some loss of detail in complex regions of the image. For a more comprehensive comparison, we can also show details resized images using bicubic interpolation and compared to Chebyshev interpolation in Figure 10, which will highlight the differences in how each method handles fine details and edge preservation.

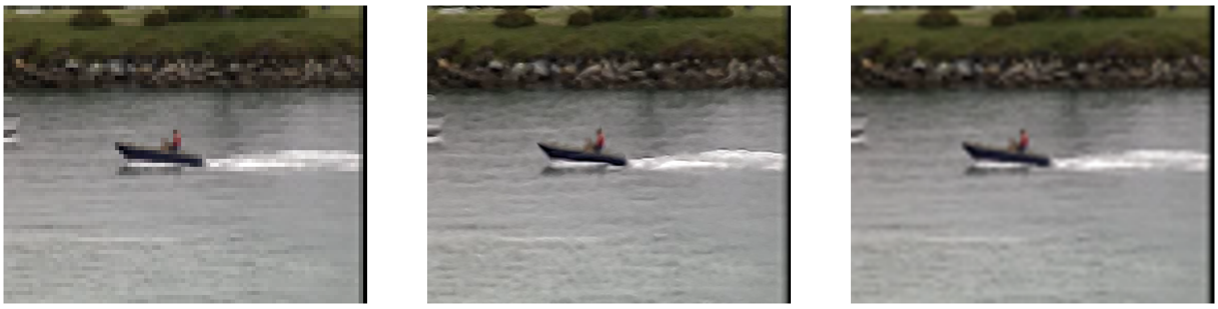

LR image Bicubic Interpolation Chebyshev interpolation

Figure 9: Resized of real *Boat* image using Chebyshev interpolation and compared to bicubic interpolation.

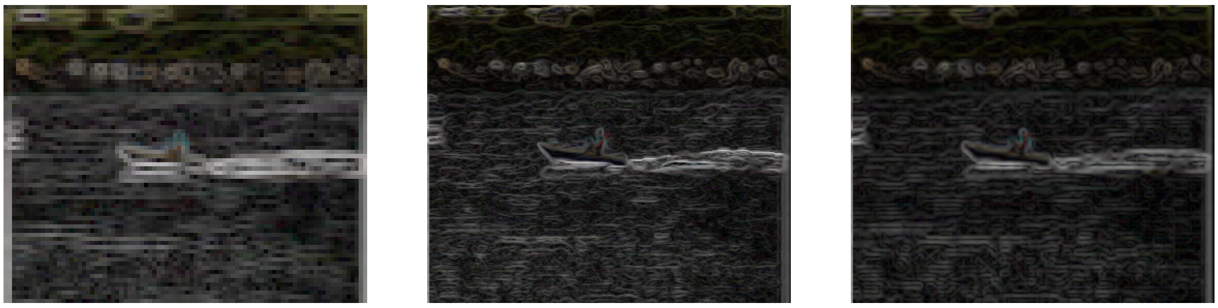

LR image Bicubic Interpolation Chebyshev interpolation

Figure 10: Image details of resized *Boat* image using Chebyshev interpolation and compared to bicubic interpolation.

4. Conclusion

In summary, this paper addresses the well-known Runge's phenomenon, which occurs during the interpolation process when uniformly spaced nodes are used to approximate smooth functions. To overcome this challenge, the paper suggests using Chebyshev roots as interpolation nodes, a method that has proven to effectively mitigate the Runge phenomenon. The paper investigates the efficiency of Chebyshev nodes by presenting a series of counterexamples that cover both differentiable and non-differentiable functions. These examples highlight the limitations of Chebyshev roots in achieving uniform convergence when used for Lagrange interpolation. Furthermore, the paper explores the application of Chebyshev interpolation in practical scenarios, such as image resizing. The results demonstrate that while Chebyshev nodes provide satisfactory outcomes in non-differentiable cases, particularly in image resizing tasks, they may still face challenges under certain conditions. By examining these counterexamples, the paper offers valuable insights into the strengths and weaknesses of Chebyshev polynomial interpolation in real-world applications. These counterexamples can also serve to better explain certain behaviors in image restoration, particularly when estimating missing pixels [10]. This could serve as a potential avenue for further investigation in future work.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. L. N. Trefethen. *Approximation Theory and Approximation Practice, Extended Edition.* SIAM, 2019.
2. A. Laghrib, A. Ghazdali, A. Hakim, S. Raghay. *A multi-frame super-resolution using diffusion registration and a nonlocal variational image restoration. Computers & Mathematics with Applications*, 72(9), 2535-2548, 2016.
3. M. Nachaoui, A. Laghrib. *An improved bilevel optimization approach for image super-resolution based on a fractional diffusion tensor. Journal of the Franklin Institute*, 359(13), 7165-7195, 2022.
4. L. Afraites, A. Hadri, A. Laghrib. *A denoising model adapted for impulse and Gaussian noises using a constrained-PDE. Inverse Problems*, 36(2), 025006, 2020.
5. J. P. Boyd and J. R. Ong. Exponentially-convergent strategies for defeating the Runge Phenomenon for the approximation of non-periodic functions, part two: Multi-interval polynomial schemes and multidomain Chebyshev interpolation. *Applied Numerical Mathematics*, 61(4):460–472, 2011.
6. J. P. Boyd and J. R. Ong. Whitney edge elements and the Runge phenomenon. *Journal of Computational and Applied Mathematics*, 427(1):115–127, 2023.
7. E. Cheney. Chebyshev methods in numerical approximation (martin avery snyder). *SIAM Review*, 9(2):264, 1967.
8. B. Fornberg and J. Zuev. The runge phenomenon and spatially variable shape parameters in rbf interpolation. *Computers & Mathematics with Applications*, 54(3):379–398, 2007.
9. J. C. Mason and D. C. Handscomb. *Chebyshev polynomials.* Chapman and Hall/CRC, 2002.
10. A. Laghrib, L. Afraites, A. Hadri, M. Nachaoui. *A non-convex pde-constrained denoising model for impulse and gaussian noise mixture reduction. Inverse Problems and Imaging*, 17(1), 23-67, 2023.

EMI, Department of Mathematics,

ENSA Khouribga, Université Sultan Moulay Slimane,

Morocco,

E-mail address: `a.laghrib@usms.ma`