*Research Paper*
*Computational Methods for Differential Equations*
http://cmde.tabrizu.ac.ir
Vol. *, No. *, *, pp. 1-14
DOI:10.22034/cmde.2026.66564.3119

# A novel, fast and accurate numerical method for three-dimensional elasticity analysis of plates

**Mahdi Adineh**[1] **and Fahimeh Akhavan Ghassabzade**[2,*]

[1]Department of Mechanical Engineering, University of Gonabad, Gonabad, 9691957678, Iran.

[2]Department of Mathematics, Faculty of Sciences, University of Gonabad, Gonabad, Iran.

**Abstract**

The finite difference (FD) method and the generalised differential quadrature (GDQ) method are numerical techniques grounded in discretising differential equations by approximating derivatives as weighted sums of function values at specific grid points within the solution domain. However, these two methods differ significantly in terms of the number of grid points required to achieve accurate solutions and the precision of the results. The GDQ method has been extensively utilised in engineering problems, such as solving differential equations derived from plate equilibrium equations. Numerous studies have demonstrated its efficiency and accuracy. In comparison to many other numerical methods, GDQ provides greater precision and faster computation.
In this paper, a novel, fast, and accurate numerical approach based on the finite difference method is developed to analyse three-dimensional elasticity in plates. This method combines the finite difference method with Lagrange interpolation to create a new algorithm for the analysis of three-dimensional elasticity in plates. A complex test case involving three-dimensional equations, boundary conditions with an elastic foundation, and a material with gradually varying properties has been selected to validate the new method. The results are compared with those from the published literature. The method exhibits remarkable superiority in performance regarding speed and simplicity when compared to the GDQ method. The numerical results illustrate the method's efficiency. It seems that applying the approach used in this study to other numerical methods could also enhance the performance of those methods.

**Keywords.** Numerical method, finite difference, interpolation, GDQ, plate.
**2010 Mathematics Subject Classification.** 65Mxx, 41A05.



## 1. Introduction

The Differential Quadrature Method (DQM) is a numerical technique for solving differential equations. It was introduced by the late Richard Bellman and his associates in the 1970s [8]. Since then, this numerical method has been widely employed in various engineering problems [3–6, 9]. For example, nonlinear bending and static analyses of annular sectors and rectangular plates with variable thickness have been conducted using this approach [3, 4]. Large-deflection behaviour of radially functionally graded sector plates resting on elastic foundations has also been examined [5], along with the thermo-mechanical nonlinear bending response of moderately thick functionally graded material plates [6]. Furthermore, the method has been extended to the three-dimensional bending analysis of multi-directional functionally graded annular sector thick plates [9].

This numerical method typically solves problems with fewer nodal points and higher accuracy compared to other similar numerical methods, which has led its proponents to consider it a suitable alternative to methods such as FDM and FEM [8].

What could be the primary reason behind such characteristics in this numerical method has been less explored. It is known that the GDQ method utilises all points in the domain to approximate derivatives, whereas methods like FDM or FEM rely only on neighbouring points for this purpose. This study focuses on employing an approach

Received: 27 March 2025; Accepted: 06 June 2026.
∗ Corresponding author. Email: akhavan_ gh@yahoo.com.

inspired by the GDQ method, where an estimation of the solution is incorporated into the formulation. This idea was first introduced in [2]. In this study, for the first time, the idea mentioned in reference [2] has been developed to solve an engineering problem. For this purpose, initially, the numerical method is explained with a clearer expansion of the subject, and then the method is developed for three-dimensional equations. Additionally, its convergence and accuracy are evaluated using the published data.

The core idea is that GDQ is built upon interpolation techniques (here, Lagrange interpolation), which inherently estimate the shape of the solution from the outset. Specifically, by using Lagrange interpolation, it assumes that the solution can be well approximated by a polynomial function. Since in many engineering problems (particularly in plate analysis, as examined in this study), solutions indeed closely resemble such functions, GDQ provides accurate results with fewer nodal points.

To investigate this hypothesis, a new approach is introduced in which Lagrange interpolation is first used to interpolate function values at specific points. Then, instead of relying on neighbouring points for differentiation (as in FDM), points derived from this interpolation near the differentiation location are utilised. This leads to a new formulation that simplifies the process compared to GDQ. To validate the accuracy of this method, stresses and bending behaviour of a functionally graded plate resting on an elastic foundation are analysed using three-dimensional elasticity theory. The results are compared with those published in previous studies, showing good agreement. This formulation can also produce the coefficients of the conventional finite difference method with slight modifications to some parameters.

Based on these findings, it can be concluded that the key feature of GDQ lies in its interpolation and solution estimation capabilities. This insight, along with the approach used in this research, could be further explored in future studies to incorporate interpolation into other numerical methods, such as FEM or Isogeometric Analysis, to enhance their accuracy and reduce required nodal points.

This paper is organised as follows: The new numerical method is formulated in section 2. Section 3 presents the problem statement. The results and discussions are presented in section 4. The calculation of the FD coefficient using the proposed method is presented in section 5. The execution time required for the code based on the method used in this research is studied in section 6. Finally, section 7 presents the conclusions of the study and summarises the findings.

## 2. New numerical method

In this method, function derivatives are approximated using finite differences, with the key difference being that the function is modelled as a polynomial. This polynomial assumption significantly reduces the number of grid points needed for problem solving.

In conventional finite difference methods, reducing the distance between grid points enhances the accuracy of derivative approximations. However, this also increases the total number of grid points required, resulting in higher computational costs. In contrast, the current method utilises a technique which allows for derivative calculations that do not depend on the spacing between grid points. As a result, accurate derivative estimates can be obtained even with fewer grid points.

The core idea behind this method is to utilise interpolation techniques at dependent points to compute function values at these locations, thereby reducing the required number of grid points. The classical backward finite difference approximation for the first derivative is given by (see Diagram (a) in Figure 1):

$$\frac{df}{dx}(x_{i+1}) \approx \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}, \tag{2.1}$$

By applying a Taylor series expansion of $f(x_{i+1})$ around $x_i$, we obtain

$$f(x_{i+1}) = f(x_i) + \triangle x.f'(x_i) + O(\triangle x^2),$$

where $x = x_{i+1} - x_i$. Hence,

$$\frac{f(x_{i+1}) - f(x_i)}{\triangle x} = f'(x_i) + O(\triangle x).$$

Therefore, the backward finite difference approximation is first-order accurate, i.e., its truncation error is O($\triangle x$).

In the new method, we select a point $x_i < x_i + \triangle x < x_{i+1}$ close to the point $x_i$, then calculate the first-order derivative using the following relation:

$$\frac{df}{dx} \approx \frac{f(x_i + \triangle x) - f(x_i)}{\triangle x}, \tag{2.2}$$

The figure below illustrates a sample distribution of grid points ($x_i$ points).

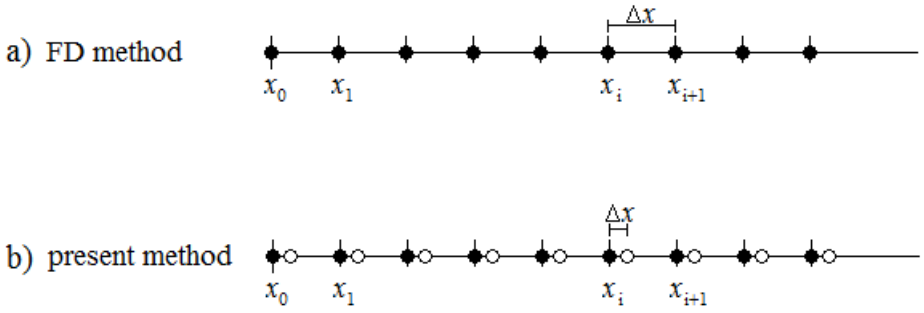


FIGURE 1. Function values at the white nodes are calculated using the interpolation of the function passing through the primary black nodes.

The function value at the point $x_i + \triangle x$ is obtained through interpolation based on the values at the grid points, which means it does not introduce any additional degrees of freedom or unknowns into the system. Since the function values at these points depend on the grid points, they are referred to as dependent grid points, represented by hollow circles in Diagram b of Figure 1, while the primary grid points are shown with solid circles. It is important to note that the value of $\triangle x$ can be assumed to be very small to improve the accuracy of estimating relation (2.2).

To interpolate the function values at these dependent grid points, various interpolation techniques can be utilised. In this study, we use Lagrange interpolation. In Lagrange interpolation, to estimate the interpolating function for N+1 distinct points, the following polynomial is used:

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

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

Therefore, to find the function value at the dependent grid points, the following relation is used:

$$f(x_i + \triangle x) = \sum_{k=0}^{N} f(x_k) L_{N,k}(x_i + \triangle x), \tag{2.3}$$

$$L_{N,k}(x_i + \triangle x) = \prod_{\substack{j=0 \\ j \neq k}}^{N} \frac{(x_i + \triangle x - x_j)}{(x_k - x_j)}. \tag{2.4}$$

By combining Equations (2.2) and (2.3), the formulation of the method is expressed in the following form:

$$\frac{df}{dx}(x_i) \approx \frac{f(x_i + \triangle x) - f(x_i)}{\triangle x} \approx \frac{\sum_{k=0}^{N} f(x_k) L_{N,k}(x_i + \triangle x) - \sum_{k=0}^{N} f(x_k) L_{N,k}(x_i)}{\triangle x} \tag{2.5}$$

$$= \frac{1}{\triangle x}\sum_{k=0}^{N} f(x_k)\big(L_{N,k}(x_i+\triangle x) - f(x_k)L_{N,k}(x_i)\big), \qquad i = 1, 2, ..., N-1.$$

Now, by substituting $L_{N,k}$ from the Equation (2.4) in Equation (2.5), we have:

$$\frac{df}{dx}(x_i) \approx \frac{1}{\triangle x}\sum_{k=0}^{N} f(x_k)\Big(\prod_{\substack{j=0\\ j\neq k}}^{N}\frac{(x_i+\triangle x - x_j)}{(x_k-x_j)} - \prod_{\substack{j=0\\ j\neq k}}^{N}\frac{x_i-x_j}{x_k-x_j}\Big), \tag{2.6}$$

where $\triangle x$ is a small constant, for example, $10^{-10}$. In the above relationship, $\prod_{\substack{j=0\\ j\neq k}}^{N}\frac{x_i-x_j}{x_k-x_j}$ can be simplified as follows:

$$\prod_{\substack{j=0\\ j\neq k}}^{N}\frac{x_i-x_j}{x_k-x_j} = \prod_{\substack{j=0\\ j\neq k}}^{N}\frac{x_k-x_j}{x_i-x_j} = 1, \qquad i = k,$$

$$\prod_{\substack{j=0\\ j\neq k}}^{N}\frac{x_i-x_j}{x_k-x_j} = \frac{x_i-x_0}{x_k-x_0}\times\frac{x_i-x_1}{x_k-x_1}\times\cdots\times\frac{x_i-x_i}{x_k-x_i}\times\cdots\times\frac{x_i-x_N}{x_k-x_N} = 0, \qquad i \neq k.$$

So, by expressing the results in matrix form, we obtain

$$F^{(1)} \approx [C^1].f,$$

where $F^{(1)} = [\frac{df}{dx}(x_0)\ \frac{df}{dx}(x_1)\cdots\frac{df}{dx}(x_N)]^T, f = [f(x_0)\ f(x_1)\cdots f(x_N)]^T$ and $[C^1]$ is a $N\times N$ matrix that :

$$[\mathbf{C^1}] = \begin{pmatrix} c^1_{11} & c^1_{12} & \cdots & c^1_{1N}\\ c^1_{21} & c^1_{22} & \cdots & c^1_{2N}\\ \vdots & \vdots & \ddots & \vdots\\ c^1_{N1} & c^1_{N2} & \cdots & c^1_{NN}\end{pmatrix},$$

where

$$c_{i,k} = \begin{cases} \frac{1}{\triangle x}\prod_{\substack{j=0\\ j\neq k}}^{N}\frac{x_i+\triangle x-x_j}{x_k-x_j}, & i\neq k,\\ \\ \frac{1}{\triangle x}\prod_{\substack{j=0\\ j\neq k}}^{N}\frac{x_i+\triangle x-x_j}{x_k-x_j} - 1, & i = k.\end{cases}$$

Now, we obtain the weighting coefficients of the second and higher order derivatives. From the definition of the differential operator, we have

$$\frac{d^2f}{dx^2} = \frac{d}{dx}(\frac{df}{dx}).$$

Let

$$\frac{d^2f}{dx^2}(x_i) \approx \sum_{j=1}^{N} c^2_{ij}.f(x_j), \qquad i = 1, 2, \cdots, N, \tag{2.7}$$

when the above method is applied to the right side of the Equation (2.7) twice, we get

$$\frac{d^2f}{dx^2}(x_i) = \sum_{k=1}^{N} c^1_{ik}.f_x(x_k) = \sum_{k=1}^{N} c^1_{ik}(\sum_{j=1}^{N} c^1_{kj}.f(x_j)) = \sum_{j=1}^{N}(\sum_{k=1}^{N} c^1_{ik}c^1_{kj}).f(x_j)), \tag{2.8}$$

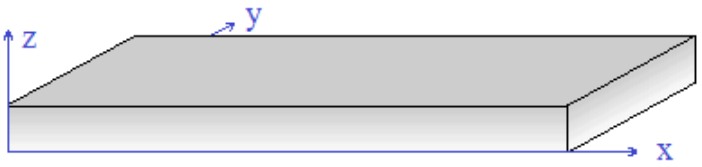


FIGURE 2. Functionally graded rectangular plate.

for $i = 1, 2, \cdots, N$ . By comparing Equations (2.7) and (2.8), we obtain

$$c^2_{ij} = \sum_{k=1}^{N} c^1_{ik}.c^1_{kj}. \tag{2.9}$$

Now, if we define one matrix by $[C^2] = [c^2_{ik}]_{N\times N}$, then Equation (2.9) gives

$$[C^2] = [C^1].[C^1].$$

Therefore, the second derivative of the function at nodal points can be calculated using the following matrix relationship:

$$F^{(2)} \approx [C^1].[C^1].f = [C^2]f.$$

In the same manner, for the n-th derivative:

$$F^{(n)} \approx [C^n]f,$$
$$[C^n] = [C^1][C^{n-1}].$$

The distribution of domain points is determined using a Chebyshev polynomial based on the given equation, which shows good agreement with Lagrange interpolation [12].

$$l_i = 0.5L(1 - \frac{\cos(i-1)\times\pi}{N-1}), \qquad i = 1, 2, ..., N. \tag{2.10}$$

In the equation above, $L$ indicates the domain length.

## 3. Problem statement

The plate under investigation in this study is a rectangular plate made of functionally graded materials (FGM) in which material properties vary through the thickness direction, as shown in Figure 2. To simulate a rectangular plate, the equations of elasticity can be utilised in Cartesian coordinates. Also, the equations can be written in cylindrical coordinates, and then, by using appropriate parameterization, a rectangular plate can be simulated. The relevant relationships are as follows [10].

Strain-Displacement Relations in Cartesian Coordinates:

$$\varepsilon_{rr} = \frac{\partial u_r}{\partial r},$$
$$\varepsilon_{r\theta} = \frac{1}{2}(\frac{1}{r}\frac{\partial u_r}{\partial \theta} + \frac{\partial u_\theta}{\partial r} - \frac{u_\theta}{r}),$$
$$\varepsilon_{rz} = \frac{1}{2}(\frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r}),$$
$$\varepsilon_{\theta\theta} = \frac{1}{r}\frac{\partial u_\theta}{\partial \theta} + \frac{u_r}{r},$$
$$\varepsilon_{\theta z} = \frac{1}{2}(\frac{\partial u_\theta}{\partial z} + \frac{1}{r}\frac{\partial u_\theta}{\partial \theta}),$$
$$\varepsilon_{zz} = \frac{\partial u_z}{\partial z},$$

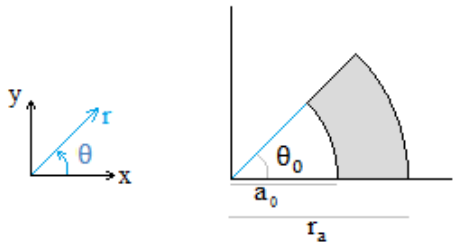


FIGURE 3. Geometric parameters of the plate in cylindrical coordinates.

where $\varepsilon_{ij}$ are strain components and $u_r, u_\theta$ and $u_z$ are displacements in r, $\theta$ and z directions respectively. In this study, the stress–strain relations are based on Hooke's law for elastic materials [7] :

$$\begin{aligned}
\varepsilon_{rr} &= \frac{1}{E}[\sigma_r - \upsilon(\sigma_\theta + \sigma_z)],\\
\varepsilon_{\theta} &= \frac{1}{E}[\sigma_\theta - \upsilon(\sigma_r + \sigma_z)],\\
\varepsilon_{z} &= \frac{1}{E}[\sigma_z - \upsilon(\sigma_r + \sigma_\theta)],\\
\sigma_{rz} &= 2G\varepsilon_{r\theta}, \quad \sigma_{rz} = 2G\varepsilon_{rz}, \quad \sigma_{\theta,z} = 2G\varepsilon_{\theta z},
\end{aligned}$$

where $\sigma_{ij}$ are stress components, E is the modulus of elasticity and $\upsilon$ is the Poisson's ratio, which are material properties and $G = \dfrac{E}{2(1+\upsilon)}$. Equilibrium equations in the absence of body forces can be written as:

$$\begin{aligned}
\frac{\partial \sigma_r}{\partial r} + \frac{1}{r}\frac{\partial \sigma_{\theta r}}{\partial \theta} + \frac{\partial \sigma_{zr}}{\partial z} + \frac{\sigma_r - \sigma_\theta}{r} &= 0,\\
\frac{\partial \sigma_{r\theta}}{\partial r} + \frac{1}{r}\frac{\partial \sigma_{\theta}}{\partial \theta} + \frac{\partial \sigma_{z\theta}}{\partial z} + \frac{2\sigma_{r\theta}}{r} &= 0,\\
\frac{\partial \sigma_{rz}}{\partial r} + \frac{1}{r}\frac{\partial \sigma_{\theta z}}{\partial \theta} + \frac{\partial \sigma_{z}}{\partial z} + \frac{\sigma_{rz}}{r} &= 0.
\end{aligned}$$

Boundary Conditions for simply supported edges and elastic foundation used in this study are:

$$\begin{aligned}
r = a_0, r_a, \quad &\longrightarrow \quad \sigma_r = u_\theta = u_z = 0,\\
\theta = 0, \theta_0, \quad &\longrightarrow \quad \sigma_\theta = u_r = u_z = 0,\\
z = 0, \quad &\longrightarrow \quad \sigma_z = k_w u_z - k_{sr}\frac{\partial^2 u_z}{\partial r^2} - k_{s\theta}(\frac{1}{r}\frac{\partial u_z}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u_z}{\partial \theta^2}), \sigma_{xz} = \sigma_{yz} = 0,\\
z = h, \quad &\longrightarrow \quad \sigma_z = 0, \sigma_{rz} = \sigma_{rz} = 0.
\end{aligned}$$

The plate generated in cylindrical coordinates is a circular sector, which can be transformed into a rectangular plate by assigning a sufficiently large value to the inner radius and a very small value to the sector angle. This approach is a common method for analysing rectangular plates using cylindrical coordinates. For example, by assigning $a_0 = 1000, \theta = 0.003, r_a = 1001$, and $z = 0.1$, a rectangular plate with a length of 3, a width of 1, and a thickness of 0.1 is simulated (See Figure 3). The material distribution used in this study is according to the following relation:

$$P = P_m(\frac{z}{h})^{n_z} + P_c(1 - (\frac{z}{h})^{n_z}),$$

where $P$ represents any material property of the plate. Modulus of elasticity for aluminium is $E_m = 70GPa$ and for alumina is $E_c = 380GPa$. The Poisson ratio for both materials is $\nu_m = \nu_c = 0.3$. The nondimensional parameters

used in this study are as follows:

$$\sigma_r^* = -\frac{h^2}{qa^2}\sigma_r(\frac{a}{2}, \frac{a_0\theta_0}{2}, 0),$$
$$\sigma_\theta^* = -\frac{h^2}{qa^2}\sigma_\theta(\frac{a}{2}, \frac{a_0\theta_0}{2}, 0),$$
$$\sigma_{r\theta}^* = -\frac{h^2}{qa^2}\sigma_{r\theta}(0, 0, 0),$$
$$W^* = \frac{100D_0}{qa^4}W(\frac{a}{2}, \frac{a_0\theta_0}{2}, \frac{h}{2}),$$
$$D_0 = \frac{E_C h^3}{12(1-\upsilon^2)},$$
$$K_0 = \frac{K_w a^4}{E_0 h^3},$$
$$J_0 = \frac{K_{sr}b^2}{E_0h^3} = \frac{K_{s\theta}a^2}{E_0h^3},$$
$$E_0 = 1.0GPa, \qquad \upsilon = 0.3.$$

## 4. Results and discussions

As described in the introduction of the proposed method, calculating derivatives involves using a small auxiliary parameter ($\triangle x$ in Equation (2.6)), which must be chosen carefully to ensure accuracy. For convenience, we refer to this parameter simply as delta. A smaller delta value generally provides a more precise approximation of the derivative at the desired point; however, very small values can cause numerical instability or round-off errors. Therefore, a numerical test was conducted in Figures 4 -7, examining a plate under specific conditions. A particular delta value was assigned each time, and the results were plotted in graphs. As shown, large and very small delta values produce unsuitable responses; for example, in this study, a delta value of $10^{-16}$ led to divergent answers. The inappropriate ranges also showed that slight changes in delta caused significant variations in numerical results, whereas in the suitable delta range, variations had little effect on the outcomes. Referring to Figures 4 -7, a delta value of $10^{-10}$ was examined for all examples in this research. The figures suggest that there is a broad, clear range for selecting the delta parameter. To evaluate the accuracy and efficiency of the proposed formulation, it was employed to solve the problem outlined in section 3. The results are presented in Tables 1 and 2. Table 1 corresponds to a plate composed of isotropic material, where the material properties remain constant throughout the plate. In contrast, Table 2 presents a plate made of functionally graded material (FGM), in which the material properties vary from one point to another according to Equation (2.1). As observed, there is a very good agreement between the obtained results and the existing reports in the literature. Furthermore, the results from the numerical method presented in these examples are identical to those obtained using the GDQ method [1] up to four decimal places. It can be stated that the strength of the GDQ method, based on the results obtained, lies in its use of interpolation, such that when the same interpolation is applied to the current method, the derivative formulation of the FD method also produces similar accuracy to GDQ. This aspect highlights the advantage of the current method due to its simpler formulation.

## 5. Calculation of the FD coefficient using the proposed formulation

One of the key features of the proposed formulation is that if the distribution of nodal points is uniform and follows Equation (5.1), and the parameter delta is chosen to be equal to the distance between nodal points, as per Equation (5.2), the coefficient matrix of the current method becomes identical to that of the FD method, except at the endpoints of the interval. The coefficient matrix ($C^1$) for the first-order derivative of a 1D problem, for an interval length of L=1, is presented in Table 3.

$$x_i = \frac{L}{N-1}(i-1), \qquad i = 1, 2, \ldots, N \tag{5.1}$$

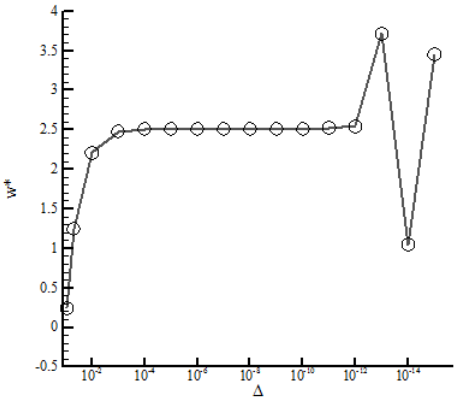


FIGURE 4. Sensitivity of the plate nondimensional deformation($w^*$) to the delta parameter. ($n_z = 1, K_0 = J_0 = 0$)

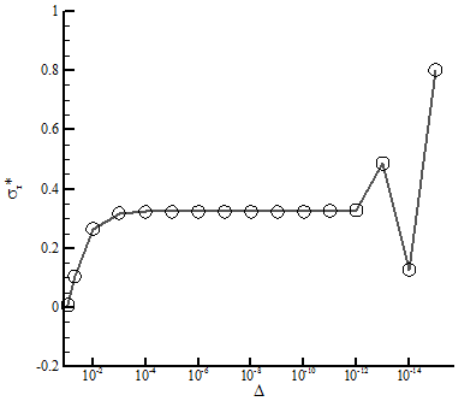


FIGURE 5. Sensitivity of the plate nondimensional stress ($\sigma_r^*$) to the delta parameter. ($n_z = 1, K_0 = J_0 = 0$)

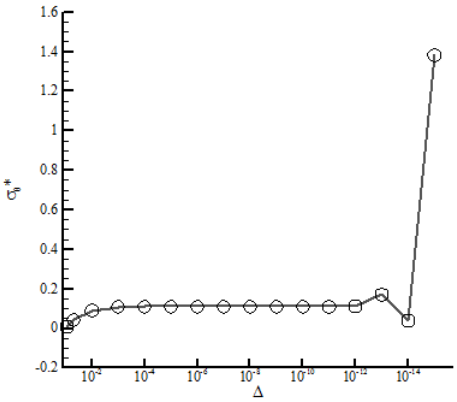


FIGURE 6. Sensitivity of the plate nondimensional stress ($\sigma_\theta^*$) to the delta parameter. ($n_z = 1, K_0 = J_0 = 0$)

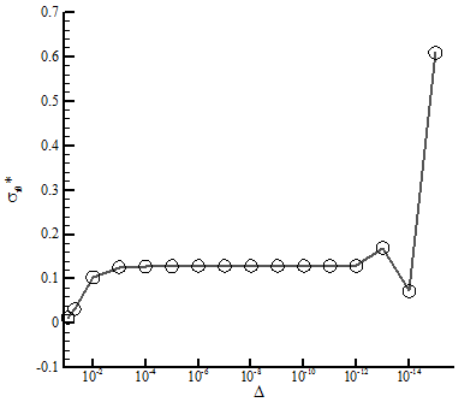


FIGURE 7. Sensitivity of the plate nondimensional stress ($\sigma_{r\theta}^*$) to the delta parameter. ($n_z = 1, K_0 = J_0 = 0$)

$$\triangle x = \frac{L}{N-1}. \tag{5.2}$$

For comparision, Table 4 presents the matrix of coefficients for the standard formulation conditions, i.e., nodes distributed according to Equation (2.10) and $\Delta = 10^{-10}$, and for the same interval length ($L = 1$).

TABLE 1. Isotropic plate ($n_z = 0$).

| $k_0$ | $J_0$ | | $w^*$ | $\sigma_r^*$ | $\sigma_\theta^*$ | $\sigma_{r\theta}^*$ |
|---|---|---|---|---|---|---|
| 0 | 0 | $7 \times 7 \times 7$ | 1.2416 | 0.7052 | 0.2310 | 0.2670 |
| | | $9 \times 9 \times 9$ | 1.2554 | 0.7155 | 0.2461 | 0.2670 |
| | | $11 \times 11 \times 11$ | 1.2547 | 0.7155 | 0.2444 | 0.2855 |
| | | $13 \times 13 \times 13$ | 1.2546 | 0.7150 | 0.2443 | 0.2860 |
| | | $15 \times 15 \times 15$ | 1.2545 | 0.7153 | 0.2445 | 0.2860 |
| | | ref[1] | 1.2545 | 0.7153 | 0.2445 | 0.286 |
| | | ref[11] | 1.2583 | 0.716 | 0.2447 | 0.289 |
| 100 | 0 | $7 \times 7 \times 7$ | 1.2097 | 0.6863 | 0.2238 | 0.2620 |
| | | $9 \times 9 \times 9$ | 1.2234 | 0.6965 | 0.2389 | 0.2781 |
| | | $11 \times 11 \times 11$ | 1.2227 | 0.6965 | 0.2372 | 0.2806 |
| | | $13 \times 13 \times 13$ | 1.2226 | 0.6961 | 0.2371 | 0.2811 |
| | | $15 \times 15 \times 15$ | 1.2226 | 0.6963 | 0.2372 | 0.2811 |
| | | ref[1] | 1.2226 | 0.6963 | 0.2375 | 0.2811 |
| | | ref[11] | 1.226 | 0.6969 | 0.2375 | 0.284 |
| 0 | 100 | $7 \times 7 \times 7$ | 1.1508 | 0.6516 | 0.2107 | 0.2526 |
| | | $9 \times 9 \times 9$ | 1.1643 | 0.6618 | 0.2258 | 0.2687 |
| | | $11 \times 11 \times 11$ | 1.1637 | 0.6617 | 0.2241 | 0.2712 |
| | | $13 \times 13 \times 13$ | 1.1635 | 0.6613 | 0.2240 | 0.2717 |
| | | $15 \times 15 \times 15$ | 1.1635 | 0.6615 | 0.2241 | 0.2717 |
| | | ref [1] | 1.1635 | 0.6615 | 0.2241 | 0.2717 |
| | | ref[11] | 1.1662 | 0.6618 | 0.2245 | 0.2744 |
| 100 | 100 | $7 \times 7 \times 7$ | 1.1232 | 0.6352 | 0.2045 | 0.2483 |
| | | $9 \times 9 \times 9$ | 1.1366 | 0.6453 | 0.2196 | 0.2644 |
| | | $11 \times 11 \times 11$ | 1.1359 | 0.6453 | 0.2179 | 0.2669 |
| | | $13 \times 13 \times 13$ | 1.1358 | 0.6449 | 0.2178 | 0.2674 |
| | | $15 \times 15 \times 15$ | 1.1358 | 0.6451 | 0.2179 | 0.2674 |
| | | ref[1] | 1.1358 | 0.6451 | 0.2179 | 0.2674 |
| | | ref[11] | 1.1382 | 0.6452 | 0.2183 | 0.27 |



## 6. Execution Time Analysis

This section examines the execution time required for the code based on the method used in this research. The specifications of the computer system used are as follows:
Processor: Intel® Core(TM) i7-4700MQ CPU @ 2.40GHz 2.40GHz
Installed memory (RAM): 16.0 GB
System type: 64-bit Operating System, x86-based processor
The table below presents the MATLAB codes for generating the first-order derivative coefficient matrix for a one-dimensional problem within the interval [0,1]. The methods used include the method proposed in this study and the GDQ method. The number of grid points in the codes corresponds to the parameter Nx+1. For example, Nx=7 implies that 8 grid points are considered for the length of the interval. The coefficient matrix c1 in the codes corresponds to the matrix $C^1$ in Equation (5.1). The remaining codes for solving equations using these two methods are identical, including the higher-order derivative coefficient matrices, which can be calculated by multiplying lower-order derivative coefficient matrices as per Equation (2.9).

As shown in Table 5, the codes for the new method are shorter. The codes were executed in MATLAB, and the execution time was measured using the tic and toc commands. The results for different numbers of grid points are presented in Table 6 and Figure 8. As observed, the execution time required for the new method is significantly less than that of the GDQ method, and this time difference increases as the number of grid points grows. Reducing the solution time can be particularly advantageous for problems that require multiple and repetitive calculations, such as optimization problems or training artificial intelligence models.

TABLE 2. FGM plate ($n_z = 1$).

| $k_0$ | $J_0$ | | $w^*$ | $\sigma_r^*$ | $\sigma_\theta^*$ | $\sigma_{r\theta}^*$ |
|---|---|---|---|---|---|---|
| 0 | 0 | $7 \times 7 \times 7$ | 2.4937 | 0.3211 | 0.1052 | 0.1208 |
| | | $9 \times 9 \times 9$ | 2.5134 | 0.3246 | 0.1117 | 0.1277 |
| | | $11 \times 11 \times 11$ | 2.5113 | 0.3245 | 0.1109 | 0.1289 |
| | | $13 \times 13 \times 13$ | 2.5110 | 0.3243 | 0.1108 | 0.1291 |
| | | $15 \times 15 \times 15$ | 2.5106 | 0.3244 | 0.1109 | 0.1291 |
| | | ref[1] | 2.5106 | 0.3244 | 0.1109 | 0.1291 |
| | | ref[11] | 2.5134 | 0.325 | 0.1111 | 0.1306 |
| 100 | 0 | $7 \times 7 \times 7$ | 2.3692 | 0.3041 | 0.0987 | 0.1164 |
| | | $9 \times 9 \times 9$ | 2.3893 | 0.3078 | 0.1052 | 0.1234 |
| | | $11 \times 11 \times 11$ | 2.3872 | 0.3076 | 0.1044 | 0.1245 |
| | | $13 \times 13 \times 13$ | 2.3869 | 0.3075 | 0.1043 | 0.1248 |
| | | $15 \times 15 \times 15$ | 2.3866 | 0.3075 | 0.1044 | 0.1248 |
| | | ref[1] | 2.3866 | 0.3075 | 0.1044 | 0.1248 |
| | | ref[11] | 2.3875 | 0.308 | 0.1047 | 0.1226 |
| 0 | 100 | $7 \times 7 \times 7$ | 2.1538 | 0.2751 | 0.0876 | 0.1085 |
| | | $9 \times 9 \times 9$ | 2.1748 | 0.2789 | 0.0942 | 0.1156 |
| | | $11 \times 11 \times 11$ | 2.1727 | 0.2787 | 0.0934 | 0.1167 |
| | | $13 \times 13 \times 13$ | 2.1725 | 0.2786 | 0.0933 | 0.117 |
| | | $15 \times 15 \times 15$ | 2.1722 | 0.2786 | 0.0934 | 0.117 |
| | | ref[1] | 2.1722 | 0.2786 | 0.0934 | 0.117 |
| | | ref[11] | 2.1703 | 0.2791 | 0.094 | 0.1182 |
| 100 | 100 | $7 \times 7 \times 7$ | 2.059 | 0.2622 | 0.0827 | 0.1051 |
| | | $9 \times 9 \times 9$ | 2.082 | 0.266 | 0.0893 | 0.1122 |
| | | $11 \times 11 \times 11$ | 2.0781 | 0.2659 | 0.0885 | 0.1134 |
| | | $13 \times 13 \times 13$ | 2.0779 | 0.2657 | 0.0884 | 0.1136 |
| | | $15 \times 15 \times 15$ | 2.0777 | 0.2658 | 0.0885 | 0.1136 |
| | | ref[1] | 2.0777 | 0.2658 | 0.0885 | 0.1136 |
| | | ref[11] | 2.0746 | 0.2663 | 0.0893 | 0.1148 |

TABLE 3. The coefficient matrix for the first-order derivative of a 1D problem. (In the form similar to the FD method).

| Number of nodes | $C^1$matrix |
|---|---|
| N=2 | $\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$ |
| N=3 | $\begin{bmatrix} -2 & 2 & 0 \\ 0 & -2 & 2 \\ 2 & -6 & 4 \end{bmatrix}$ |
| N=4 | $\begin{bmatrix} -3 & 3 & 0 & 0 \\ 0 & -3 & 3 & 0 \\ 0 & 0 & -3 & 3 \\ -3 & 12 & -18 & 9 \end{bmatrix}$ |
| N=5 | $\begin{bmatrix} -4 & 4 & 0 & 0 & 0 \\ 0 & -4 & 4 & 0 & 0 \\ 0 & 0 & -4 & 4 & 0 \\ 0 & 0 & 0 & -4 & 4 \\ 4 & -20 & 40 & -40 & 16 \end{bmatrix}$ |
| N=6 | $\begin{bmatrix} -5 & 5 & 0 & 0 & 0 & 0 \\ 0 & -5 & 5 & 0 & 0 & 0 \\ 0 & 0 & -5 & 5 & 0 & 0 \\ 0 & 0 & 0 & -5 & 5 & 0 \\ 0 & 0 & 0 & 0 & -5 & 5 \\ -5 & 30 & -75 & 100 & -75 & 25 \end{bmatrix}$ |

TABLE 4. The coefficient matrix for the first-order derivative of a 1D problem. (In the standard form).

| Number of nodes | $C^1$matrix |
|---|---|
| N=2 | $\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$ |
| N=3 | $\begin{bmatrix} -3 & 4 & -1 \\ -1 & 0 & 1 \\ 1 & -4 & 3 \end{bmatrix}$ |
| N=4 | $\begin{bmatrix} -6.3333 & 8 & -6.3333 & 1 \\ -2 & 0.6667 & 2 & -0.6667 \\ 0.6667 & -2 & -0.6667 & 2 \\ -1 & 2.6667 & -8 & 6.3333 \end{bmatrix}$ |
| N=5 | $\begin{bmatrix} -11 & 13.6569 & -4 & 2.3431 & -1 \\ -3.4142 & 1.4142 & 2.8284 & -1.4142 & 0.5858 \\ 1 & -2.8284 & 0 & 2.8284 & -1 \\ -0.5858 & 1.4142 & -2.8284 & -1.4142 & 3.4142 \\ 1 & -2.3431 & 4 & -13.6569 & 11 \end{bmatrix}$ |
| N=6 | $\begin{bmatrix} -17 & 20.9443 & -5.7889 & 3.0557 & -2.2111 & 1 \\ -5.2361 & 2.3416 & 4 & -1.7889 & 1.2361 & -0.5528 \\ 1.4472 & -4 & 0.3416 & 3.2361 & -1.7889 & 0.7639 \\ -7639 & 1.7889 & -3.2361 & -0.3416 & 4 & -1.4472 \\ 0.5528 & -1.2361 & 1.7889 & -4 & -2.3416 & 5.2361 \\ -1 & 2.2111 & -3.0557 & 5.7889 & -20.9443 & 17 \end{bmatrix}$ |

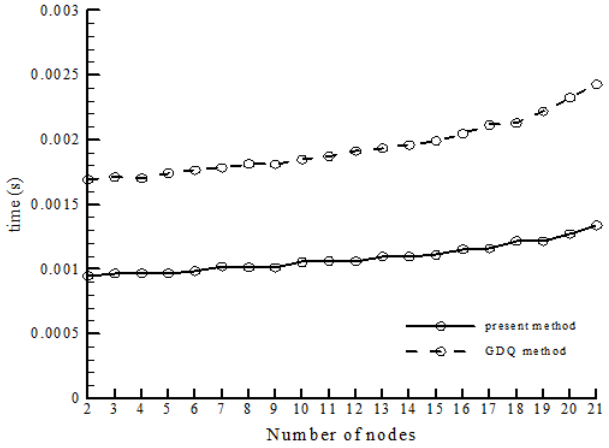


FIGURE 8. The execution time required for the MATLAB code based on the present method and the GDQ method

Table 7 shows the matrix $C^1$ calculated by both methods for 2 to 6 grid points, which are observed to be very close to each other.

## 7. CONCLUSIONS

In this study, inspired by the generalised differential quadrature (GDQ) method, which has gained significant attention from researchers in recent years for solving engineering equations, including those related to elasticity theory, an attempt was made to derive a new formulation by incorporating interpolation within the finite difference formulation and develop it for a three-dimensional elasticity problem. The results obtained indicate:
The proposed formulation is simpler than the GDQ method, yet it achieves accuracy very close to that of the GDQ

TABLE 5. Matlab codes for generating the $C^1$ matrix based on the present method and the GDQ method.

Present method

```
clear all
clc
tic
lengthx=1;
Nx=5;
deltax=-1*1e-10;
for iNx=1:Nx+1
x(iNx)=lengthx*0.5*(1-cos((iNx-1)*pi/(Nx
    )));
end
c1=zeros(Nx+1);
for j=1:Nx+1
    xplus=x(j)+deltax;
for k=1:Nx+1
    c1(j,k)=1;
for i=1:Nx+1
    if i==k
        c1(j,k)=c1(j,k);
    else
        c1(j,k)=c1(j,k)*(xplus-x(i))/(x(
            k)-x(i));
    end
end
end
end
c1=c1-eye(Nx+1);
c1=(1/deltax)*c1;
toc
 break
```

GDQ method

```
clear all
clc
tic
lengthx=1;
Nx=17;
for iNx=1:Nx+1
x(iNx)=lengthx*0.5*(1-cos((iNx-1)*pi/(Nx
    )));
end
c1=zeros(Nx+1);
for i=1:Nx+1
    for j=1:Nx+1
      R1(i)=1;
    for k=1:Nx+1
       if k==i
           R1(i)=R1(i);
       else
           R1(i)=R1(i)*(x(i)-x(k));
       end
    end
      R2(j)=1;
    for kk=1:Nx+1
       if kk==j
           R2(j)=R2(j);
       else
           R2(j)=R2(j)*(x(j)-x(kk));
       end
    end
      c1(i,j)=R1(i)/((x(i)-x(j))*R2(j));
    end
end
for i=1:Nx+1
    c1(i,i)=0;
    for j=1:Nx+1
        if i==j
            c1(i,i)=c1(i,i);
        else
    c1(i,i)=c1(i,i)-c1(i,j);
        end
    end
end
toc
break
```

TABLE 6. The execution time required for the MATLAB code based on the present method and the GDQ method.

| c | present method | GDQ |
|---|---|---|
| 2 | 0.000948 | 0.001694 |
| 3 | 0.000967 | 0.001714 |
| 4 | 0.00097 | 0.001702 |
| 5 | 0.00097 | 0.001743 |
| 6 | 0.000985 | 0.001764 |
| 7 | 0.00102 | 0.001784 |
| 8 | 0.001014 | 0.001815 |
| 9 | 0.001011 | 0.001809 |
| 10 | 0.001055 | 0.001846 |
| 11 | 0.001064 | 0.001873 |
| 12 | 0.001062 | 0.001911 |
| 13 | 0.0011 | 0.001935 |
| 14 | 0.001098 | 0.001959 |
| 15 | 0.001111 | 0.001992 |
| 16 | 0.001154 | 0.002048 |
| 17 | 0.001158 | 0.002115 |
| 18 | 0.001218 | 0.002131 |
| 19 | 0.001216 | 0.002219 |
| 20 | 0.001273 | 0.002326 |
| 21 | 0.001338 | 0.00243 |

TABLE 7. $C^1$ Matrix calculated by present method and GDQ method

| Number of nodes | $c^1$ matrix | |
|---|---|---|
| | Present method | GDQ method |
| N=2 | $\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$ | $\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$ |
| N=3 | $\begin{bmatrix} -3 & 4 & -1 \\ -1 & 0 & 1 \\ 1 & -4 & 3 \end{bmatrix}$ | $\begin{bmatrix} -3 & 4 & -1 \\ -1 & 0 & 1 \\ 1 & -4 & 3 \end{bmatrix}$ |
| N=4 | $\begin{bmatrix} -6.3333 & 8 & -2.6667 & 1 \\ -2 & 0.6667 & 2 & -0.6667 \\ 0.6667 & -2 & -0.6667 & 2 \\ -1 & 2.6667 & -8 & 6.3333 \end{bmatrix}$ | $\begin{bmatrix} -6.3333 & 8 & -2.6667 & 1 \\ -2 & 0.6667 & 2 & -0.6667 \\ 0.6667 & -2 & -0.6667 & 2 \\ -1 & 2.6667 & -8 & 6.3333 \end{bmatrix}$ |
| N=5 | $\begin{bmatrix} -11 & 13.6569 & -4 & 2.3431 & -1 \\ -3.4142 & 1.4142 & 2.8284 & -1.4142 & 0.5858 \\ 1 & -2.8284 & 0 & 2.8284 & -1 \\ -0.5858 & 1.4142 & -2.8284 & -1.4142 & 3.4142 \\ 1 & -2.3431 & 4 & -13.6569 & 11 \end{bmatrix}$ | $\begin{bmatrix} -11 & 13.6569 & -4 & 2.3431 & -1 \\ -3.4142 & 1.4142 & 2.8284 & -1.4142 & 0.5858 \\ 1 & -2.8284 & 0 & 2.8284 & -1 \\ -0.5858 & 1.4142 & -2.8284 & -1.4142 & 3.4142 \\ 1 & -2.3431 & 4 & -13.6569 & 11 \end{bmatrix}$ |
| N=6 | $\begin{bmatrix} -17 & 20.9443 & -5.7889 & 3.0557 & -2.2111 & 1 \\ -5.2361 & 2.3416 & 4 & -1.7889 & 1.2361 & -0.5528 \\ 1.4472 & -4 & 0.3416 & 3.2361 & -1.7889 & 0.7639 \\ -7639 & 1.7889 & -3.2361 & -0.3416 & 4 & -1.4472 \\ 0.5528 & -1.2361 & 1.7889 & -4 & -2.3416 & 5.2361 \\ -1 & 2.2111 & -3.0557 & 5.7889 & -20.9443 & 17 \end{bmatrix}$ | $\begin{bmatrix} -17 & 20.9443 & -5.7889 & 3.0557 & -2.2111 & 1 \\ -5.2361 & 2.3416 & 4 & -1.7889 & 1.2361 & -0.5528 \\ 1.4472 & -4 & 0.3416 & 3.2361 & -1.7889 & 0.7639 \\ -7639 & 1.7889 & -3.2361 & -0.3416 & 4 & -1.4472 \\ 0.5528 & -1.2361 & 1.7889 & -4 & -2.3416 & 5.2361 \\ -1 & 2.2111 & -3.0557 & 5.7889 & -20.9443 & 17 \end{bmatrix}$ |



method. Additionally, the number of nodes required for the solution is reduced.
Using the new formulation, it is possible to generate the required coefficients for the derivative matrix of the finite difference method, with only minor changes in the constants, except for endpoint nodes. This research and the new formulation contribute to a better understanding of how the GDQ method operates. It appears that the approach introduced in this study, as it has improved the performance of the finite difference method, could also be effective in formulating and enhancing the performance of other numerical methods, including the finite element method, which could be a topic for future research.

## DECLARATION OF COMPETING INTEREST

The authors declare that there is no conflict of interest.

## Authors contributions

All authors contributed equally and significantly in writing this article. All authors read and approved the final manuscript.

## References


[1] M. Adineh and M. Kadkhodayan, *Three-dimensional thermo-elastic analysis of multi-directional functionally graded rectangular plates on elastic foundation*, Acta Mechanica, *228* (2017), 881–899.

[2] M. Adineh, *Thermomechanical static and dynamic analysis of multidirectional functionally graded skew plates on elastic foundation based on three-dimensional theory of elasticity*, Phd dissertation, Ferdowsi University of Mashhad, 2017.

[3] F. Alinaghizadeh and M. Shariati, *Geometrically non-linear bending analysis of thick two-directional functionally graded annular sector and rectangular plates with variable thickness resting on non-linear elastic foundation*, Composites Part B: Engineering, *86* (2016), 61–83.

[4] F. Alinaghizadeh and M. Shariati, *Static analysis of variable thickness two-directional functionally graded annular sector plates fully or partially resting on elastic foundations by the GDQ method*, Journal of the Brazilian Society of Mechanical Sciences and Engineering, *37* (2015), 1819–38.

[5] F. Alinaghizadeh and M. Kadkhodayan, *Large deflection analysis of moderately thick radially functionally graded annular sector plates fully and partially rested on two-parameter elastic foundations by GDQ method*, Aerospace Science and Technology, *39* (2014), 260–71.

[6] F. Alinaghizadeh and M. Kadkhodayan, *Investigation of nonlinear bending analysis of moderately thick functionally graded material sector plates subjected to thermomechanical loads by the GDQ method*, Journal of engineering mechanics, *140* (2014), 04014012.

[7] K. Asemi, M. Salehi, and M. Akhlaghi, *Post-buckling analysis of FGM annular sector plates based on three-dimensional elasticity graded finite elements*, International Journal of Non-Linear Mechanics, *67* (2014), 164–177.

[8] C.W Bert and M. Malik, *Differential quadrature method in computational mechanics: a review*, Appl. Mech. Rev. Jan, *49*(1) (1996), 1–28.

[9] M. Livani, *Three-dimensional bending analysis of multi-directional functionally graded annular sector thick plates*, Aerospace Knowledge and Technology Journal, *9* (2020), 113–124.

[10] V. Tahouneh and M. H. Yas, *3-D free vibration analysis of thick functionally graded annular sector plates on Pasternak elastic foundation via 2-D differential quadrature method*, Acta Mechanica, *223*(9) (2012), 1879–1897.

[11] H. T. Thai and D. H. Choi, *A refined plate theory for functionally graded plates resting on elastic foundation*, Compos. Sci. Technol., *71*(16) (2011), 1850–1858.

[12] L. N. Trefethen, *Approximation Theory and Approximation Practice*, Society for Industrial and Applied Mathematics, 2012.