Numerical Methods for Partial Differential Equations

WILEY

RESEARCH ARTICLE OPEN ACCESS

A Flexible Derivation Approach for the Numerical Solution of Partial Differential Equations

Nadaniela Egidi | Josephin Giacomini | Pierluigi Maponi

School of Science and Technology–Mathematics Division, University of Camerino, Camerino, Macerata, Italy

Correspondence: Nadaniela Egidi (nadaniela.egidi@unicam.it)

Received: 28 May 2025 | **Revised:** 28 May 2025 | **Accepted:** 17 October 2025

Funding: The authors received no specific funding for this work.

Keywords: boundary value problems | graph routing problems | numerical differentiation

ABSTRACT

We propose a new method for the numerical solution of boundary value problems associated to partial differential equations. This method is based on standard approximation techniques, like numerical differentiation of univariate functions and curve interpolation, so it can be easily generalized to high-dimensional problems. However, the concrete implementation of this method requires the proper solution of a routing problem for the graph associated with the discretized domain $\mathcal{N}$. This graph routing problem has an immediate solution when $\mathcal{N}$ has a grid structure. Instead, when $\mathcal{N}$ has a mesh structure or is given by sparse points, it is possible to take advantage of methods for two classical graph routing problems, that is, the Chinese postman problem and the Eulerian path problem. However, it will be shown that these problems do not provide a sufficiently satisfactory solution, hence, further study for non-structured grids is needed. A numerical experiment shows the effectiveness of the proposed method in the case of $\mathcal{N}$ with a grid structure.

1 | Introduction

Real-world phenomena can usually be properly simplified and described through boundary or initial-boundary value problems [1, 2]. A particularly important example is given by Climate Change, indeed water is the primary medium through which Climate Change influences the ecosystem of Earth, and the fluid dynamic equations can be used to model the hydrogeological risks [3, 4] and the hydraulic risk [5]. Another important geological process is the diffusion of underground thermal energy, which must be considered in the analysis of geothermal exchangers through forced-convective heat transfer [6, 7], [8], and in the optimal installation of several geothermal exchangers [9]. The study of the flow of water in porous media is important for the management of water resources [10], and also for the extraction of espresso coffee, which is regulated by a percolation model in a porous medium given by roasted and ground coffee [11–13]. Medicine provides other important examples, where the Navier-Stokes equations for non-Newtonian fluids and the interaction between fluid and boundary are used for the hemodynamic analysis of the human cardiovascular system [14]. Moreover, differential equations can be used to simulate human respiratory function [15], to model cancer growth in the human body [16, 17], and for pharmaceutical drug design [18, 19].

These problems give an extremely restricted subset of examples where differential equations play a fundamental role. However, they have a clear applied relevance, with an impact on the strategic fields of energy, medicine, environment, and sustainable development. All the above-described real-world phenomena have models that do not usually admit analytical solutions, thus, they undergo numerical methods for approximating

the involved partial differential equations. In particular, these methods have to deal with different difficulties such as nonlinear differential equations, domains with complex shapes, and domains with moving boundaries. Hence we need to develop new methods for the numerical solution of differential equations and/or refine the existing methods such as finite difference methods [20, 21], finite element methods [22, 23], finite volume methods [24, 25], spectral methods [26–28] and meshfree methods [29, 30].

In this paper, we propose a new method for the numerical solution of boundary value problems for partial differential equations. Of course, as with other similar methods, it is applicable also to initial-boundary value problems after a semi-discrete approximation in time. This method is essentially based on two main elements: the numerical differentiation of univariate functions and the curvilinear interpolation. The wide choice of efficient methods for solving such elementary problems constitutes a strong point of the proposed method and allows its easy implementation in computer codes. In particular, for the numerical differentiation of univariate functions, we have considered and analysed a spectral method based on the integral formulation of the derivative operator [31–33], [34]. The other fundamental element of the method, that is, the interpolation curve, has to be adapted to the spatial structure of the discretization points, which generally fall into the following cases: grid, mesh or a set of sparse points. In the case of the grid structure, we have considered the interpolation curves obtained by spline functions. This choice gives satisfactory results and can be immediately generalized to other different types of curves, with a higher regularity, such as the radial basis functions. The cases of meshes and sets of sparse points have been analysed by using classical methods to solve graph routing problems, like the Cinese Postman problem [35] and the Eulerian Path problem [36]. Unfortunately, the interpolating curves obtained from the solution of these problems are not satisfactory. So, it is necessary to refine the obtained interpolating curves or study other techniques. Thus, the cases of meshes and sets of sparse points are a work in progress. Instead, in the case of grids, the proposed method has been tested by considering various partial differential equations.

The paper is organized as follows. Section 2 presents the method for the numerical differentiation of univariate functions and the use of this method to approximate the partial derivatives of a multivariate function. Section 3 gives the implementation of the aforementioned numerical derivation method in the solution of partial differential equations. Section 4 describes the results obtained in a numerical experiment with the proposed method in the case of grids. Section 5 provides some conclusions and remarks for future investigations.

2 | Numerical Derivation in the Univariate and Multivariate Cases

We start by introducing the main notation. Let $\mathbb{N}$ be the set of natural numbers, $\mathbb{R}$ be the set of real numbers and $\mathbb{R}^N$ be the N-dimensional real Euclidean space. An element $\mathbf{x} \in \mathbb{R}^N$ is a column vector $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$, where the superscript T denotes the transposition operator. We denote with $\mathbb{R}^{M\times N}$ the space of real matrices having M rows and N columns. For $\mathbf{x}$, $\mathbf{y} \in \mathbb{R}^N$, $\|\mathbf{x}\|_1$ denotes the 1-norm of $\mathbf{x}$, $\mathbf{x}^T\mathbf{y}$ denotes the Euclidean scalar product of $\mathbf{x}$ and $\mathbf{y}$, and

$$\mathbf{x} * \mathbf{y} = (x_1 y_1, x_2 y_2, \ldots, x_N y_N)^T \in \mathbb{R}^N, \tag{1}$$

is the element-wise multiplication of $\mathbf{x}$ and $\mathbf{y}$.

For integers j, k, and $N > 0$, we define

$$\gamma_j = (2j-1)\frac{\pi}{2}, \tag{2}$$

$$c_{j,k,N} = \cos\left(\gamma_j \frac{k}{N}\right), \quad s_{j,k,N} = \sin\left(\gamma_j \frac{k}{N}\right). \tag{3}$$

Let $\mathbf{z} = (z_1, z_2, \ldots, z_N)^T \in \mathbb{R}^N$, for $p = 1, 2, \ldots, N$,

$$\left(FST^{(3)}(\mathbf{z})\right)_p = \frac{1}{\sqrt{N}}\left(2\sum_{q=1}^{N-1} z_q s_{p,q,N} + (-1)^{p-1} z_N\right), \tag{4}$$

$$\left(FCT^{(2)}(\mathbf{z})\right)_p = \frac{1}{\sqrt{N}}\sum_{q=1}^{N} z_q c_{q,p-1,N}, \tag{5}$$

$$\left(FCT^{(3)}(\mathbf{z})\right)_p = \frac{1}{\sqrt{N}}\left(z_1 + 2\sum_{q=2}^{N} z_q c_{p,q-1,N}\right), \tag{6}$$

are the components of the discrete sine transform of type III of $\mathbf{z}$, the discrete cosine transform of type II of $\mathbf{z}$ and the discrete cosine transform of type III of $\mathbf{z}$, respectively [37].

For the reader's convenience, we briefly describe a numerical derivation method for smooth functions $f : \Omega \to \mathbb{R}$, where $\Omega \subseteq \mathbb{R}^n$, since it is a fundamental element in the approximation procedure discussed in Section 3. A detailed description of these methods can be found in [33, 38]. In particular, the case $n = 1$ is reported in Section 2.1 and it provides the basis for the multivariate case $n > 1$, described in Section 2.2.

2.1 | Numerical Derivation in the Univariate Case

Given a sufficiently smooth function $G : [0, 1] \to \mathbb{R}$, the following Algorithm 1 computes the numerical derivative of G by using the properties of the singular value expansion of the Fredholm integral operator associated with the first order derivative [32]; see [38], [39] for a detailed derivation of Algorithm 1.

We consider the following points in $[0, 1]$,

$$\mathcal{T}(N) = \left\{ t_k = k\delta \in [0,1] \ : \ \delta = \frac{1}{N-1}, \right.$$
$$\left. k = 0, 1, \ldots, N-1 \right\}, \tag{7}$$

where we suppose $N \geq 9$; from the knowledge of $G_k = G(t_k)$, $t_k \in \mathcal{T}(N)$, $k = 0, 1, \ldots, N-1$, the following algorithm gives the approximation D_k of $G'(t_k)$.

Algorithm 1 (Numerical computation of the first-order derivative).

INPUT: $N \geq 9$ *and* $G_k = G(t_k)$, $t_k \in \mathcal{T}(N)$, $k = 0, 1, \ldots, N-1$.
OUTPUT: the approximation D_k *of* $G'(t_k)$, $k = 0, 1, \ldots, N-1$.

1. *Compute* $\delta = 1/(N-1)$, $\boldsymbol{\beta}, \mathbf{c}_j, \mathbf{s}_j \in \mathbb{R}^N$, $j = 1, 2$,

$$\beta_1 = 1, \quad \beta_l = 2,\ l = 2, 3, \ldots, N, \tag{8}$$

$$\mathbf{c}_j = (c_{1,j,N}, c_{2,j,N}, \ldots, c_{N,j,N})^T,$$
$$\mathbf{s}_j = (s_{1,j,N}, s_{2,j,N}, \ldots, s_{N,j,N})^T. \tag{9}$$

2. *Construct* $\mathbf{a}^{i,c} = (a_1^{i,c}, a_2^{i,c}, \ldots, a_N^{i,c})^T$, $\mathbf{a}^{j,s} = (a_1^{j,s}, a_2^{j,s}, \ldots, a_N^{j,s})^T \in \mathbb{R}^N$, $i = 0, 1, 2$, $j = 1, 2$, *in the following way. The components* $a_l^{i,c}$, $l = 1, 2, \ldots, N$, *and* $i = 0, 1, 2$, *are all zero except the following*

$$a_1^{i,c} = \begin{cases} \frac{1}{12\delta}\left(-25G_0 + 16G_3 - 3G_4\right), & i = 0, \\ -\frac{1}{4\delta}G_0, & i = 1, \\ \frac{1}{12\delta}G_0, & i = 2, \end{cases} \tag{10}$$

$$a_2^{i,c} = \begin{cases} \frac{1}{24\delta}\left(-10G_1 + G_4\right), & i = 0, \\ \frac{10}{6\delta}G_1, & i = 1, \\ \frac{1}{24\delta}G_1, & i = 2, \end{cases}$$

$$a_3^{i,c} = \begin{cases} \frac{5}{12\delta}G_2, & i = 1, \\ -\frac{35}{24\delta}G_2, & i = 2, \end{cases} \tag{11}$$

$$a_4^{2,c} = -\frac{5}{24\delta}G_3, \quad a_{N-3}^{2,c} = \frac{5}{24\delta}G_{N-4}, \tag{12}$$

$$a_{N-2}^{i,c} = \begin{cases} -\frac{5}{12\delta}G_{N-3}, & i = 1, \\ \frac{35}{24\delta}G_{N-3}, & i = 2, \end{cases} \tag{13}$$

$$a_{N-1}^{i,c} = \begin{cases} \frac{5}{12\delta}G_{N-2}, & i = 0, \\ \frac{10}{6\delta}G_{N-2}, & i = 1, \\ -\frac{1}{24\delta}G_{N-2}, & i = 2, \end{cases} \quad a_N^{i,c} = \begin{cases} \frac{25}{24\delta}G_{N-1}, & i = 0, \\ \frac{1}{8\delta}G_{N-1}, & i = 1, \\ -\frac{1}{24\delta}G_{N-1}, & i = 2. \end{cases} \tag{14}$$

Among the components $a_l^{j,s}$, $l = 1, 2, \ldots, N$, *and* $j = 1, 2$, *those non-zero are*

$$a_l^{1,s} = \begin{cases} \frac{14}{6\delta}G_l, & l = 1 \ \vee \ l = N-2, \\ \frac{13}{12\delta}G_l, & l = 2 \ \vee \ l = N-3, \\ \frac{4}{6\delta}G_l, & 3 \le l \le N-4, \\ \frac{1}{8\delta}G_l, & l = N-1, \\ -\frac{4}{3\delta}G_{l-4} + \frac{1}{4\delta}G_{l-5}, & l = N, \end{cases} \tag{15}$$

$$a_l^{2,s} = \begin{cases} -\frac{1}{24\delta}G_l, & l = 1 \ \vee \ l = N-2 \ \vee \ l = N-1, \\ -\frac{37}{24\delta}G_l, & l = 2 \ \vee \ l = N-3, \\ -\frac{7}{24\delta}G_l, & l = 3 \ \vee \ l = N-4, \\ -\frac{1}{12\delta}G_l, & 4 \le l \le N-5, \\ -\frac{1}{12\delta}G_{l-5}, & l = N. \end{cases} \tag{16}$$

3. *For* $i = 0, 1, 2$, *and* $j = 1, 2$, *compute*

$$\mathbf{b}^{i,c} = FCT^{(3)}\left(\mathbf{a}^{i,c}\right), \qquad \mathbf{b}^{j,s} = FST^{(3)}\left(\mathbf{a}^{j,s}\right). \tag{17}$$

4. *Compute*

$$\mathbf{b} = \mathbf{b}^{0,c} + \sum_{j=1,2}\left(\mathbf{b}^{j,c} * \mathbf{c}_j + \mathbf{b}^{j,s} * \mathbf{s}_j\right). \tag{18}$$

5. *Compute* $\mathbf{D} = (D_0, D_1, \ldots, D_{N-1})^T \in \mathbb{R}^N$,

$$\mathbf{D} = FCT^{(2)}(\mathbf{b}) * \boldsymbol{\beta}. \tag{19}$$

2.2 | Numerical Derivation in the Multivariate Case

We briefly describe the generalisation of the numerical derivation method in Section 2.1 to the multivariate case ($n > 1$).

Let $f : \Omega \subseteq \mathbb{R}^n \to \mathbb{R}$ be a sufficiently smooth function, let $\mathbf{x} \in \Omega$ and $\mathbf{h} = (h_1, h_2, \ldots, h_n) \in \mathbb{R}^n$ such that $\mathbf{x} + t\mathbf{h} \in \Omega$ for every $t \in [0, 1]$. Then the first-order derivatives of f satisfy

$$f(\mathbf{x} + t\mathbf{h}) = f(\mathbf{x}) + \sum_{i=1}^{n} h_i \int_0^t \frac{\partial f}{\partial x_i}(\mathbf{x} + s\mathbf{h})ds, \ t \in [0, 1], \tag{20}$$

see [40] for details, the function $F(t) = f(\mathbf{x} + t\mathbf{h})$, $t \in [0, 1]$, satisfies

$$F(t) = F(0) + \int_0^t F'(s)ds, \ t \in [0, 1], \tag{21}$$

and an approximation of $F'(t)$, $t \in \mathcal{T}(N)$, $N \ge 9$, can be computed by Algorithm 1 if F is known on $\mathcal{T}(N)$.

Formulas (20) and (21) together with Algorithm 1 may be used to compute the partial derivatives of f along rectilinear segments in Ω. However, a more flexible approach can be obtained by considering a generic continuously differentiable function $\mathbf{g} = (g_1, g_2, \ldots, g_n)^T : [0, 1] \to \Omega$ and by defining $G(t) = f(\mathbf{g}(t))$, $t \in [0, 1]$. Indeed, we have

$$G(t) = G(0) + \int_0^t G'(s)ds, \qquad t \in [0, 1], \tag{22}$$

or equivalently

$$f(\mathbf{g}(t)) = f(\mathbf{g}(0)) + \sum_{i=1}^{n} \int_0^t g_i'(s) \frac{\partial f}{\partial x_i}(\mathbf{g}(s))ds, \qquad t \in [0, 1]. \tag{23}$$

We suppose to know f on $\mathcal{P}(N) = \{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{N-1}\} \subset \Omega$, and to have a function $\mathbf{g} : [0, 1] \to \Omega$ such that $\mathbf{g}(t_l) = \mathbf{p}_l$, $t_l \in \mathcal{T}(N)$, $l = 0, 1, \ldots, N-1$. Then, for the corresponding function $G(t) = f(\mathbf{g}(t))$, $t \in [0, 1]$, we have

$$\sum_{i=1}^{n} g_i'(t_l) \frac{\partial f}{\partial x_i}(\mathbf{p}_l) = G'(t_l), \ l = 0, 1, \ldots, N-1. \tag{24}$$

Hence, by computing $d_l \approx G'(t_l)$ with Algorithm 1, the numerical derivatives $D_i(\mathbf{p}_l) \approx \frac{\partial f}{\partial x_i}(\mathbf{p}_l)$, $i = 1, 2, \ldots, n$, can be computed by imposing that

$$\sum_{i=1}^{n} g_i'(t_l) D_i(\mathbf{p}_l) = d_l, \ l = 0, 1, \ldots, N-1. \tag{25}$$

The accuracy of the numerical derivatives computed by (25) increases as N increases, in particular, in Theorem 1 of [38], we proved that $d_l = G'(t_l) + O(1/N^4)$, $l = 0, 1, \ldots, N-1$.

However, linear system (25) does not give an immediate approximation of the derivatives of the function f at a point $\mathbf{p} \in \mathcal{P}(N)$ when $n > 1$; in fact at each point $\mathbf{p}_l$, we have n unknowns, $D_i(\mathbf{p}_l) \approx \frac{\partial f}{\partial x_i}(\mathbf{p}_l)$, $i = 1, 2, \ldots, n$, and only one equation which involves these unknowns. In particular, each equation in (25) involves the scalar product of the gradient ∇f with the tangent vector

$$\boldsymbol{\tau}(t) = \left(g_1'(t), g_2'(t), \ldots, g_n'(t)\right)^T, \tag{26}$$

of the curve $\mathbf{g}(t)$ at the discretization points $t \in \mathcal{T}(N)$. Thus, for each $\mathbf{p} \in \mathcal{P}(N)$, the derivatives of f at $\mathbf{p}$ can be computed when we know at least n curves like $\mathbf{g}$ interpolating $\mathbf{p}$ or a unique curve $\mathbf{g}$ passing through $\mathbf{p}$ at least n times. More in detail, we have the following theorem.

Theorem 1. *Given the data set $f(\mathcal{N})$, corresponding to a function $f : \Omega \to \mathbb{R}$ sufficiently smooth and to a set $\mathcal{N} \subset \Omega$ having finite cardinality equal to $N_{\mathcal{N}}$. Supposing that*:

1. *there exist J curves*

$$\mathbf{g}_j = (g_{1,j}, g_{2,j}, \ldots, g_{n,j})^T : [0,1] \to \Omega, \quad j = 1, 2, \ldots, J, \tag{27}$$

and J subsets $\mathcal{P}_j \subseteq \mathcal{N}$, such that $\mathbf{g}_j$ interpolates the points in $\mathcal{P}_j$ at the N_j nodes in $\mathcal{T}_j$, where

$$\mathcal{T}_j = \left\{ \frac{k}{N_j - 1} \ : \ k = 0, 1, \ldots, N_j - 1 \right\} \subset [0,1];$$

2. *there exist the tangent vectors*

$$\boldsymbol{\tau}_j(t) = \left(g_{1,j}'(t), g_{2,j}'(t), \ldots, g_{n,j}'(t)\right)^T \in \mathbb{R}^n, \qquad t \in [0,1], \tag{28}$$

of the curves $\mathbf{g}_j$, $j = 1, 2, \ldots, J$;

3. *for each point $\mathbf{p} \in \mathcal{N}$, the set*

$$J_{\mathbf{p}} = \{ j \in \mathbb{N} \ : \ 1 \le j \le J, \ \mathbf{p} \in \mathcal{P}_j \} \tag{29}$$

is not empty; note that this implies that for $j \in J_{\mathbf{p}}$, $\mathbf{p} \in \mathcal{N}$, the curve $\mathbf{g}_j$ interpolates $\mathbf{p}$, and the set

$$\mathcal{T}_{j,\mathbf{p}} = \{ t \in \mathcal{T}_j \ : \ \mathbf{g}_j(t) = \mathbf{p} \} \tag{30}$$

is not empty;

4. *for each $\mathbf{p} \in \mathcal{N}$, the set*

$$\left\{ \boldsymbol{\tau}_j(t) \in \mathbb{R}^n \ : \ j \in J_{\mathbf{p}}, \ t \in \mathcal{T}_{j,\mathbf{p}} \right\}$$

spans $\mathbb{R}^n$.

Then, for $\mathbf{p} \in \mathcal{N}$, $j \in J_{\mathbf{p}}$ and $\bar{t} \in \mathcal{T}_{j,\mathbf{p}}$, the numerical derivatives of f, $D_i(\mathbf{p}) \approx \frac{\partial f}{\partial x_i}(\mathbf{p})$, $i = 1, 2, \ldots, n$, are well-defined as the solution of the linear system:

$$\sum_{i=1}^{n} g_{i,j}'(\bar{t}) D_i(\mathbf{p}) = d_j(\bar{t}), \ \bar{t} \in \mathcal{T}_{j,\mathbf{p}}, \ j \in J_{\mathbf{p}}, \tag{31}$$

where $d_j(\bar{t})$ is a numerical approximation of $G_j'(\bar{t})$.

Proof. The numerical derivatives $D_i(\mathbf{p}) \approx \frac{\partial f}{\partial x_i}(\mathbf{p})$, $i = 1, 2, \ldots, n$, of f at points $\mathbf{p} \in \mathcal{N}$ can be computed by the following elementary steps.

i. For $j = 1, 2, \ldots, J$, compute

$$d_j(t) \approx G_j'(t), \ t \in \mathcal{T}_{j,\mathbf{p}}, \ \mathbf{g}_j(t) = \mathbf{p} \in \mathcal{P}_j \subset \mathcal{N}.$$

ii. For $j = 1, 2, \ldots, J$, and $i = 1, 2, \ldots, n$, compute

$$g_{i,j}'(t), \ t \in \mathcal{T}_j.$$

iii. For $\mathbf{p} \in \mathcal{N}$, compute $D_i(\mathbf{p})$, $i = 1, 2, \ldots, n$, as the unique solution of the linear system (31), which is obtained from (25) with $\mathbf{g}_j$, $j \in J_{\mathbf{p}}$ and $\bar{t} \in \mathcal{T}_{j,\mathbf{p}}$. In fact, from hypothesis 4., the rows of the coefficient matrix in the linear system (31) span $\mathbb{R}^n$, which implies that the rank of this matrix is equal to n (the number of unknowns) and the solution is unique. We note that, regarding this last point, when the cardinality of $J_{\mathbf{p}}$ is greater than n the solution of (25) is intended as the unique least-square solution. □

Remark 1. Theorem 1 gives a constructive procedure to practically compute the partial derivative of a multivariate function f at points $\mathcal{N} \subset \Omega$. In particular, when the curves $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, are obtained, the numerical derivatives $d_j(t), j \in J_p, t \in \mathcal{T}_{j,\mathbf{p}}$ of univariate functions $G_j(t)$ can be computed by using Algorithm 1 with input N_j and

$$f(\mathcal{P}_j) = \left\{ G_j(t_k) \ : \ t_k \in \mathcal{T}_j, \ k = 0, 1, \ldots, N_j - 1 \right\}.$$

In particular, when Algorithm1 is used, the requirement $N_j \ge 9$ must be added in the hypotheses of the Theorem 1.

Note that the same procedure can be applied to the first-order derivatives in place of f for the computation of second-order derivatives. For derivations of order greater than one, we do not need additional conditions, the same $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, can be used and so the same coefficient matrices are obtained in linear systems (31).

In the following, the quality of the approximated derivatives, computed by the procedure described in Theorem 1, is shown in a simple example.

Example 1. We consider a two-dimensional case $n = 2$, where $\Omega = \mathbb{R} \times \mathbb{R}^+$ and $f(\mathbf{x}) = \log(x_1^2 + x_2 + 1)$, $\mathbf{x} \in \Omega$. The function f is sampled at the points in

$$\mathcal{N} = \left\{ \mathbf{p}_{l,m} = \left(t_l - \frac{s_m}{3}, \frac{e^{t_l}}{2} + \frac{s_m}{4} \right)^T \in \Omega \ : \ t_l = l/(N_1 - 1), \right.$$
$$l = 0, 1, \ldots, N_1 - 1, \ s_m = m/(N_2 - 1),$$
$$\left. m = 0, 1, \ldots, N_2 - 1 \right\}, \tag{32}$$

see Figure 1 for a pictorial description of such points. The hypotheses of Theorem 1 are satisfied by the following $J = N_1 + N_2$ curves $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, equal to the cubic splines interpolating the following sets of points

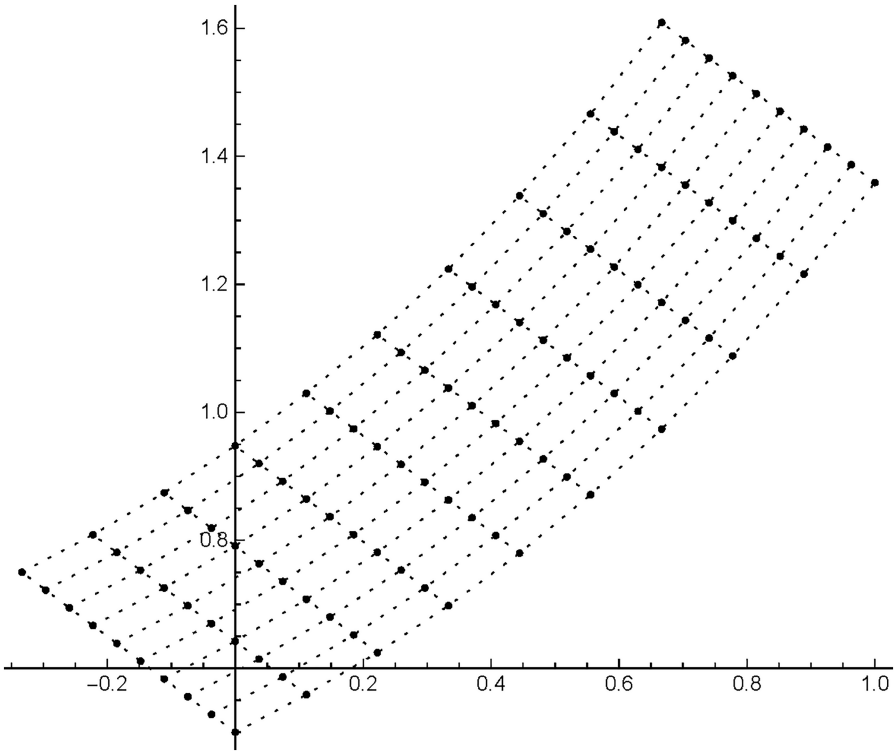

FIGURE 1 | Example 1, the points in $\mathcal{N}$ given in (32) where $N_1 = N_2 = 10$ and the $J = 20$ cubic splines $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, with nodes $\mathcal{T}(10)$.

TABLE 1 | Example 1, the mean square error E of the computed numerical derivatives of f in Example 1. The notation $x(y)$ stays for $x \cdot 10^y$.

$N_1 = N_2$	E
10	9.9 (−5)
20	3.7 (−6)
40	1.8 (−7)
80	1.0 (−8)

$$\mathcal{P}_{1,l} = \{\mathbf{p}_{l,m} \in \mathcal{N} \ : \ m = 0, 1, \ldots, N_2 - 1\}, \quad l = 0, 1, \ldots, N_1 - 1,$$

$$\mathcal{P}_{2,m} = \{\mathbf{p}_{l,m} \in \mathcal{N} \ : \ l = 0, 1, \ldots, N_1 - 1\}, \quad m = 0, 1, \ldots, N_2 - 1,$$

at nodes in $\mathcal{T}_2$ and $\mathcal{T}_1$, respectively. Table 1 shows the mean square error E in the numerical derivatives of f at points of $\mathcal{N}$, for $N_1 = N_2 = 10, 20, 40, 80$, computed with the procedure illustrated in the Theorem 1 and the Algorithm 1.

3 | Numerical Solution of Partial Differential Equations

The numerical derivation method described in the previous section can be used to implement a procedure for solving partial differential equations. We propose an easy implementation of this method that in principle can deal with different spatial organisation of the discretisation points $\mathcal{N}$, that is, grids, meshes, and sparse points, supposing that there exist J curves $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, satisfying the hypotheses of Theorem 1.

For the sake of simplicity, we consider the following linear second-order equation:

$$\sum_{i,j=1}^{n} a_{i,j}(\mathbf{x}) \frac{\partial^2 u}{\partial x_i \partial x_j}(\mathbf{x}) + \sum_{i=1}^{n} b_i(\mathbf{x}) \frac{\partial u}{\partial x_i}(\mathbf{x}) + c(\mathbf{x}) u(\mathbf{x}) = \mathrm{f}(\mathbf{x}), \ \mathbf{x} \in \Omega, \tag{33}$$

where $a_{i,j}$, b_i, c, $\mathrm{f} : \Omega \subset \mathbb{R}^n \to \mathbb{R}$, $i, j = 1, 2, \ldots, n$, are known functions and $u : \Omega \to \mathbb{R}$ is the unknown solution. The following boundary condition is prescribed to the function u:

$$u(\mathbf{x}) = \mathrm{f}_D(\mathbf{x}), \ \mathbf{x} \in B_D, \tag{34}$$

$$\frac{\partial u}{\partial \underline{\hat{n}}}(\mathbf{x}) = \mathrm{f}_N(\mathbf{x}), \ \mathbf{x} \in B_N, \tag{35}$$

where B_D and B_N are subsets of the boundary $\partial\Omega$ of Ω, such that $B_D \cup B_N = \partial\Omega$, $\underline{\hat{n}}(\mathbf{x})$ denotes the outward unit normal vector to $\partial\Omega$ at $\mathbf{x} \in B_N$, and $\mathrm{f}_D : B_D \to \mathbb{R}$, $\mathrm{f}_N : B_N \to \mathbb{R}$ are known functions. The set B_D is called the Dirichlet boundary of Ω, B_N is called its Neumann boundary, and when one of the sets B_D or B_N is empty the corresponding condition is dropped having in this way only the Neumann condition or the Dirichlet condition, respectively.

From now on, we suppose that problem (33), (34), (35) has a unique solution. The corresponding numerical solution can be obtained by an iterative approach applied to the linear system associated with the discretized problem, where different tentative approximate solutions are computed, and for each one of these solutions we need only to evaluate the residual of equations

(33), (34), (35). In details, according to the notation introduced in the previous sections, we denote with $\mathcal{N}$ the set of discretization points in Ω where we want to compute the approximate solution of problem (33), (34), (35). For $\mathbf{p} \in \mathcal{N}$, we denote with $U(\mathbf{p}) \approx u(\mathbf{p})$ a given approximate solution and with $D_i(\mathbf{p})$ and $D^2_{i,j}(\mathbf{p}), i, j = 1, 2, \ldots, n$, the approximations of the first-order and second-order derivatives, respectively, of the tentative solution $U(\mathbf{p})$, computed using the method described in Section 2.2, with curves $\mathbf{g}_j, j = 1, 2, \ldots, J$, satisfying the hypotheses of Theorem 1.

Let $\mathcal{N}_D = \mathcal{N} \cap B_D$ be the set of the discretization points belonging to the Dirichlet boundary B_D, and $\mathcal{N}_N = \mathcal{N} \cap B_N$ be the analogous set for the Neumann boundary. Let $\mathcal{N}_0 = \mathcal{N} \setminus (\mathcal{N}_D \cup \mathcal{N}_N)$ be the subset of internal discretization points. In order to compute the next tentative solution with an iterative method for linear systems, like GMRES, we only need to evaluate the residual vector having components $r(\mathbf{p})$, $\mathbf{p} \in \mathcal{N}$, that are obtained as follows:

$$r(\mathbf{p}) = \sum_{i,j=1}^{n} a_{i,j}(\mathbf{p}) D^2_{i,j}(\mathbf{p}) + \sum_{i=1}^{n} b_i(\mathbf{p}) D_i(\mathbf{p}) + c(\mathbf{p}) U(\mathbf{p}) - \mathrm{f}(\mathbf{p}), \ \mathbf{p} \in \mathcal{N}_0, \tag{36}$$

$$r(\mathbf{p}) = U(\mathbf{p}) - \mathrm{f}_D(\mathbf{p}), \ \mathbf{p} \in \mathcal{N}_D, \tag{37}$$

$$r(\mathbf{p}) = (D_1(\mathbf{p}), D_2(\mathbf{p}), \ldots, D_n(\mathbf{p}))^T \cdot \underline{\hat{n}}(\mathbf{p}) - \mathrm{f}_N(\mathbf{p}), \ \mathbf{p} \in \mathcal{N}_N. \tag{38}$$

Thus, the flexibility of the numerical derivation method described in Section 2.2 allows an immediate application in the solution of the above-described differential equation, and such procedure depends only on the initial choice of $\mathbf{g}_j, j = 1, 2, \ldots, J$, satisfying hypothesis of Theorem 1. In the following two remarks, we analyse how we can construct $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, depending on the type of the discretization points in $\mathcal{N}$.

Remark 2 (mesh and sparse points). When $\mathcal{N}$ has a mesh structure, the various discretization points in $\mathcal{N}$ have a natural list of connections provided by the edges of the mesh. Instead, when $\mathcal{N}$ is given by a sparse set of points in Ω the connections among these points are not provided, but can be computed by analyzing the properties of $\mathcal{N}$. For example, these connections can be obtained by a proximity analysis of points and joining directions.

So, in both cases, the set $\mathcal{N}$ can be equipped with a connection map among its points, and we can construct $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, satisfying the hypothesis of Theorem 1 by using such information.

Analyzing in detail the above described case, where one has only a connection map among the points in $\mathcal{N}$, from this information, we consider the (undirected) graph $G = (V, A)$ with vertices $V = \mathcal{N}$ and with arcs A the edges of the mesh or the connections established between the points of $\mathcal{N}$, in case of sparse points. We recall that the degree of a vertex is defined as the number of arcs joining this vertex with other vertices. We note that when the degree of each vertex of G is at least $2n - 1$ we can easily construct a unique curve $\mathbf{g} : [0, 1] \to \Omega$ covering all the arcs of G, and passing through each vertex at least n times, otherwise someone of the (at least) $2n - 1$ arcs incident this vertex is not covered. To this aim, the Chinese postman problem can be considered, that is: find the shortest path that visits every arc of the graph at least once, see [35] for details. In principle, from the solution of the Chinese postman problem, we can compute a curve $\mathbf{g}$ passing through the nodes $\mathcal{N}$, which will be organized according to this solution. In particular, the curve $\mathbf{g}$ can be computed by using cubic-spline interpolation or RBF interpolation techniques, and hence we can obtain the numerical derivatives from the procedure suggested in Theorem 1 and Algorithm 1. Unfortunately, all the hypotheses of Theorem 1 are satisfied from this curve $\mathbf{g}$ with the exception of hypothesis 4., which has to be checked since it depends on the vertices position. Hence this elegant solution is usually unable to provide satisfactory results without an ad-hoc refinement. This fact is shown by the following example about a simple graph and the corresponding solution of the Chinese postman problem, for which the corresponding $\mathbf{g}$ does not satisfy hypothesis 4. of Theorem 1.

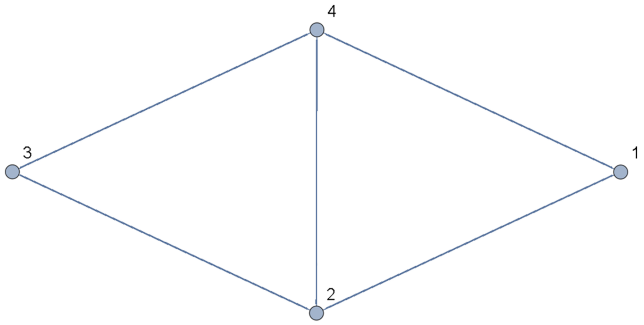

FIGURE 2 | The graph G of Example 2.

Example 2. Let $G = (V, A)$ be the undirected graph having vertices $V = \{1, 2, 3, 4\}$ and arcs $A = \{(1, 2), (1, 4), (2, 3), (2, 4), (3, 4)\}$, shown in Figure 2. For graph G, the solution of the Chinese postman problem is given by the path through the following vertices: $1, 4, 2, 4, 3, 2, 1$.

From this example, we can see that, also in the simple graph G shown in Figure 2, the solution of the corresponding Chinese postman problem gives a curve $\mathbf{g}$ with a sharp change in the direction of the curve; this can be seen from the part "$\ldots, 4, 2, 4, \ldots$", where the path from the vertex 4 goes to the vertex 2 and immediately goes back to the vertex 4. Of course, curves with sharp changes in the moving direction are not well suited for the numerical differentiation method described in Section 2.

A slightly better situation is obtained by considering the Eulerian paths for the graph G associated with $\mathcal{N}$. We note that an Eulerian path is a trail that visits every edge exactly once [36]. So, if G is an Eulerian graph, admitting in this way an Eulerian trail, we cannot obtain paths passing through a vertex and immediately going back along the same arc. This provides a relevant improvement in the path regularity. Unfortunately, even for the Eulerian graph, it is possible to obtain paths having an extremely large curvature since the path is not computed by taking into account the geometrical position of the points associated with the vertices of the graph.

Hence, when we have a mesh or a set of sparse points, the computation of a regular path $\mathbf{g}$, or a family of regular paths $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, passing through all the points of $\mathcal{N}$, as required by Theorem 1, needs further investigation and shall be discussed in a future study. This is the only point to be further investigated in order to apply the proposed method in these cases. Instead, when $\mathcal{N}$ has a grid structure, the solution is given by the following remark.

Remark 3 (grid). When $\mathcal{N}$ has a grid structure the various discretization points in $\mathcal{N}$ have a natural list of connections provided by the edges of the grid. In particular, we suppose that $\mathcal{N}$ is the set of the $N_1 \cdot N_2 \cdots N_n$ points of an n-dimensional grid; let

$$I = \left\{\mathbf{i} = (i_1, i_2, \ldots, i_n) \in \mathbb{N}^n : i_\alpha = 1, 2, \ldots, N_\alpha, \alpha = 1, 2, \ldots, n\right\}, \tag{39}$$

be the set of the indices for the points in $\mathcal{N}$, that is $\mathcal{N} = \{\mathbf{p}_\mathbf{i} \in \mathbb{R}^n, \mathbf{i} \in I\}$. For $\beta = 1, 2, \ldots, n$, we define

$$I_\beta = \left\{(i_1, i_2, \ldots, i_{\beta-1}, i_{\beta+1}, \ldots, i_n) \in \mathbb{N}^{n-1} \ : \ \mathbf{i} \in I\right\}, \tag{40}$$

$$\pi_\beta : I \to I_\beta, \qquad \pi_\beta(\mathbf{i}) = (i_1, i_2, \ldots, i_{\beta-1}, i_{\beta+1}, \ldots, i_n), \ \mathbf{i} \in I. \tag{41}$$

We define

$$\mathcal{P}(\beta; \mathbf{j}) = \left\{\mathbf{p}_\mathbf{i} \in \mathcal{N} \ : \ \mathbf{i} \in \ I, \ \pi_\beta(\mathbf{i}) = \mathbf{j}\right\}, \quad \mathbf{j} \in I_\beta, \ \beta = 1, 2, \ldots, n, \tag{42}$$

the subset of the points of $\mathcal{N}$ with constant indices $\mathbf{j} = (i_1, \ldots, i_{\beta-1}, i_{\beta+1}, \ldots, i_n) \in I_\beta$. When the grid $\mathcal{N}$ is smooth and regular, the curves $\mathbf{g}_{\beta;\mathbf{j}}$ through the points in $\mathcal{P}(\beta; \mathbf{j})$, $\mathbf{j} \in I_\beta$, $\beta = 1, 2, \ldots, n$, satisfy the hypotheses of Theorem 1. We note that the curves passing through such sets of points give an optimal solution satisfying Theorem 1, in fact they pass only one time on the arcs of G and each vertex is crossed exactly n times (counting also the starting and the ending points of the curves); so, each one of the linear systems (31) has exactly n equations and does not require a least square solution.

4 | Numerical Results

We present the results of a numerical experiment to test the performance of the proposed method. For the sake of brevity, we restrict our attention to the two-dimensional and three-dimensional problems. The two-dimensional problems considered in the experiment are: (1) the Poisson differential equation

$$\frac{\partial^2 u}{\partial x_1^2}(\mathbf{x}) + \frac{\partial^2 u}{\partial x_2^2}(\mathbf{x}) = \frac{x_1 x_2 (6 + 5x_1^2 + 5x_2^2)}{(1 + x_1^2 + x_2^2)^{3/2}}, \tag{43}$$

with solution $u(x_1, x_2) = x_1 x_2 \sqrt{x_1^2 + x_2^2 + 1}$; (2) the Helmholtz equation

$$\frac{\partial^2 u}{\partial x_1^2}(\mathbf{x}) + \frac{\partial^2 u}{\partial x_2^2}(\mathbf{x}) + (x_1^2 + x_2^2) u(\mathbf{x}) = 0, \tag{44}$$

with solution $u(x_1, x_2) = \cos(x_1 x_2)$. Each one of these equations is considered in different domains and with different boundary conditions, where the right-hand sides f_D, f_N (see equations (34), (35)) are computed from the explicit solution of the corresponding differential equation. In the part B_N of the domain boundary, where the Neumann condition is prescribed, an explicit knowledge of the outward normal vector is assumed.

We consider equations (43), (44) defined in the following domains, see Figure 3 for a pictorial representation of these domains:

$$\Omega^{(1)} = \left\{\left(t, \frac{1}{2}\sin(2\pi t) + s\right)^T \ : \ t \in [0, 1], \ s \in [0, 1]\right\}, \tag{45}$$

$$\Omega^{(2)} = \left\{\left((s+1)\cos(\pi t), \left(s + 1 + \frac{1}{2} s \sin(\pi t)\right)\sin(\pi t)\right)^T \ : \ t \in [0, 1], \ s \in [0, 1]\right\}, \tag{46}$$

where the boundary conditions (34), (35) are relative to the following boundary subsets:

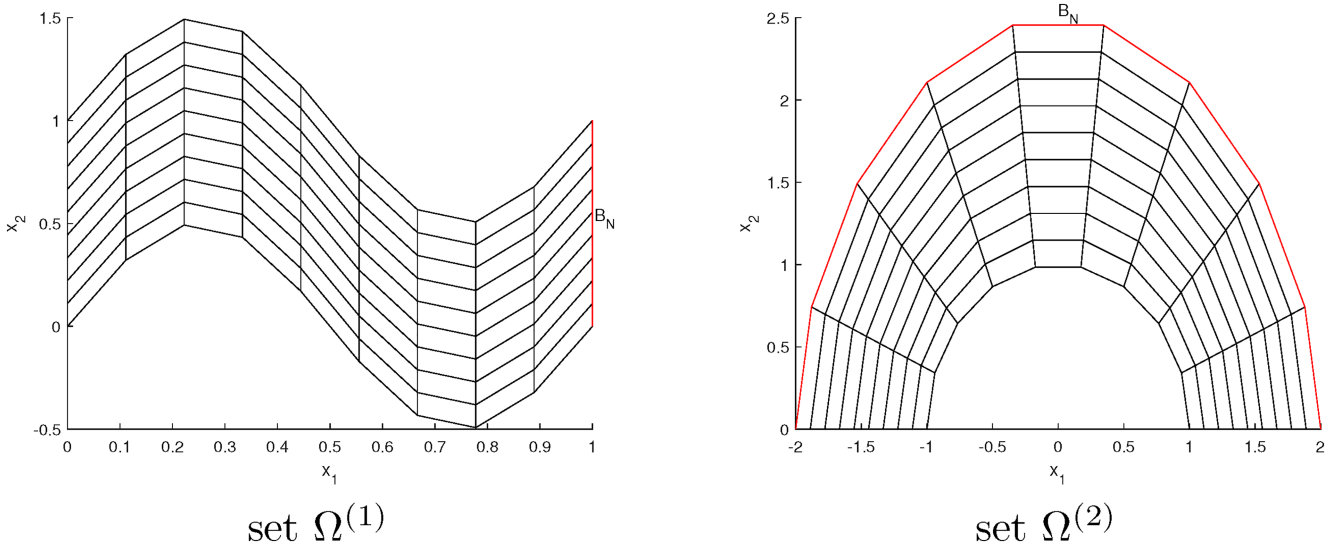

FIGURE 3 | The sets $\Omega^{(1)}$ and $\Omega^{(2)}$ with the corresponding points of $\mathcal{N}^{(1)}$ and $\mathcal{N}^{(2)}$ joined by edges; the red part of the boundary shows the set B_N where the Neumann condition is prescribed.

TABLE 2 | The mean square error E in the numerical solution of boundary value problems for equations (43), (44).

	N	Equation (43)	Equation (44)
$\Omega^{(1)}$	10	1.1(−2)	5.3(−3)
	20	8.5(−4)	7.2(−4)
	40	2.0(−5)	3.1(−5)
$\Omega^{(2)}$	10	2.4(−2)	2.4(0)
	20	1.3(−3)	6.6(−2)
	40	7.4(−5)	7.4(−4)

Note: The notation $x(y)$ stays for $x \cdot 10^y$.

$$B_N^{(1)} = \left\{ \left(t, \frac{1}{2}\sin(2\pi t) + s\right)^T \ : \ t = 1, \ s \in [0,1] \right\},$$
$$B_D^{(1)} = \partial\Omega^{(1)} \setminus B_N^{(1)}, \tag{47}$$
$$B_N^{(2)} = \left\{ \left((s+1)\cos(\pi t), \left(s + 1 + \frac{1}{2}s\sin(\pi t)\right)\sin(\pi t)\right)^T \ : \ t \in [0,1], \ s = 1 \right\},$$
$$B_D^{(2)} = \partial\Omega^{(2)} \setminus B_N^{(2)}. \tag{48}$$

We have considered different quadrilateral grids obtained by discretizing these domains with $N = N_1 = N_2$ steps along each direction. The edges of the grid have been used, as suggested by Remark 3, to obtain curves $\mathbf{g}_j$, $j = 1, 2, \ldots, J = 2N$, considered in Theorem 1; to this aim the cubic splines interpolating these sets of points are used. So, we have implemented the derivation method described in Section 2.2, and in turn the iterative solution of the discretization scheme, solved with GMRES with residuals computed by (36) to (38).

Table 2 shows the results of the numerical simulation for the two-dimensional problems. In particular, this table shows the mean square error in the numerical solution, that is

$$E = \sqrt{\frac{1}{|\mathcal{N}|}\sum_{\mathbf{p}\in\mathcal{N}} (U(\mathbf{p}) - u(\mathbf{p}))^2}, \tag{49}$$

where u is the exact solution, U is the approximated solution and $|\mathcal{N}|$ denotes the number of discretization points. We note that the solution of the linear system, obtained from the above described discretization scheme with residual vector given by (36) to (38), has been computed by the GMRES method [41] implemented in Matlab [42].

The three-dimensional problems considered in the experiment are: (1) the Laplace differential equation

$$\frac{\partial^2 u}{\partial x_1^2}(\mathbf{x}) + \frac{\partial^2 u}{\partial x_2^2}(\mathbf{x}) + \frac{\partial^2 u}{\partial x_3^2}(\mathbf{x}) = 0, \tag{50}$$

with solution $u(\mathbf{x}) = e^{-(x_1+x_2)}\cos(\sqrt{2}x_3)$; (2) the Helmholtz equation

$$\frac{\partial^2 u}{\partial x_1^2}(\mathbf{x}) + \frac{\partial^2 u}{\partial x_2^2}(\mathbf{x}) + \frac{\partial^2 u}{\partial x_3^2}(\mathbf{x}) + \frac{3}{(1+x_1^2+x_2^2+x_3^2)^2}u(\mathbf{x}) = \frac{6x_1x_2x_3}{(1+x_1^2+x_2^2+x_3^2)^{(3/2)}}, \tag{51}$$

with solution $u(\mathbf{x}) = \frac{x_1x_2x_3}{\sqrt{x_1^2+x_2^2+x_3^2+1}}$. As in the two-dimensional problems, the boundary data f_D, f_N are computed from the explicit solution of the corresponding differential equation. These equations are defined in the following domains, see Figure 4 for a pictorial representation of these domains,

$$\Omega^{(3)} = \left\{ \left(t, \frac{1}{2}\cos(\pi t) + s, r + \frac{rt}{2}\right)^T \ : \ t \in [0,1], \ s \in [0,1], \ r \in [0,1] \right\}, \tag{52}$$

$$\Omega^{(4)} = \left\{ \left((s+1)\cos(\pi t), \left(s + 1 + \frac{s}{2}\sin(\pi t)\right)\sin(\pi t) \times \left(1 + \frac{s}{2}\right), r(1+st)\right)^T : t \in [0,1], \ s \in [0,1], \ r \in [0,1]\}. \tag{53}$$

The boundary conditions are relative to the following boundary subsets:

$$B_N^{(3)} = \left\{ \left(t, \frac{1}{2}\cos(\pi t) + s, r + \frac{rt}{2}\right)^T : t = 0, \ s \in [0,1], \ r \in [0,1]\} \right\},$$
$$B_D^{(3)} = \partial\Omega^{(3)} \setminus B_N^{(3)}, \tag{54}$$

$$B_N^{(4)} = \left\{ \left((s+1)\cos(\pi t), \left(s + 1 + \frac{s}{2}\sin(\pi t)\right)\sin(\pi t) \times \left(1 + \frac{s}{2}\right), r(1+st)\right)^T : t \in [0,1], \ s = 0, \ r \in [0,1] \right\},$$
$$B_D^{(4)} = \partial\Omega^{(4)} \setminus B_N^{(4)}. \tag{55}$$

We have considered different hexahedral grids obtained by discretizing these domains with $N = N_i$, $i = 1, 2, 3$, steps in each direction. Also in the three-dimensional problems, Remark 3 has been followed to satisfy the hypotheses of Theorem 1, that is to obtain curves $\mathbf{g}_j$, $j = 1, 2, \ldots, J = 3N^2$. To this aim, the cubic splines interpolating these sets of discretisation points of the grid lines are used. So, the proposed discretization scheme has been implemented using the derivation method described in Section 2.2.

Table 3 shows the results of the numerical simulation for the three-dimensional problems providing the mean square error (49) in the numerical solution. Also in this case, the solution of the discretization scheme (36) to (38) has been computed by the GMRES method [41] implemented in Matlab [42].

From Tables 2 and 3, we can observe a quite satisfactory performance of the proposed method, also in consideration of previous works on similar problems [43–45]. An exception is given by Equation (51), which is numerically solved with a mean square error $E < 6(-2)$ but this error does not decrease when the discretization parameters increase. The strength of the proposed method, when compared to the methods presented in [43, 44] and [45], is the simplicity of the various elements that compose it: curve interpolation, numerical derivation of univariate functions, numerical solution of linear systems. The modular structure of the method facilitate its improvement procedures and its adaptation to different application problems; in addition, its structure seems to be highly suitable for parallel computations. However,

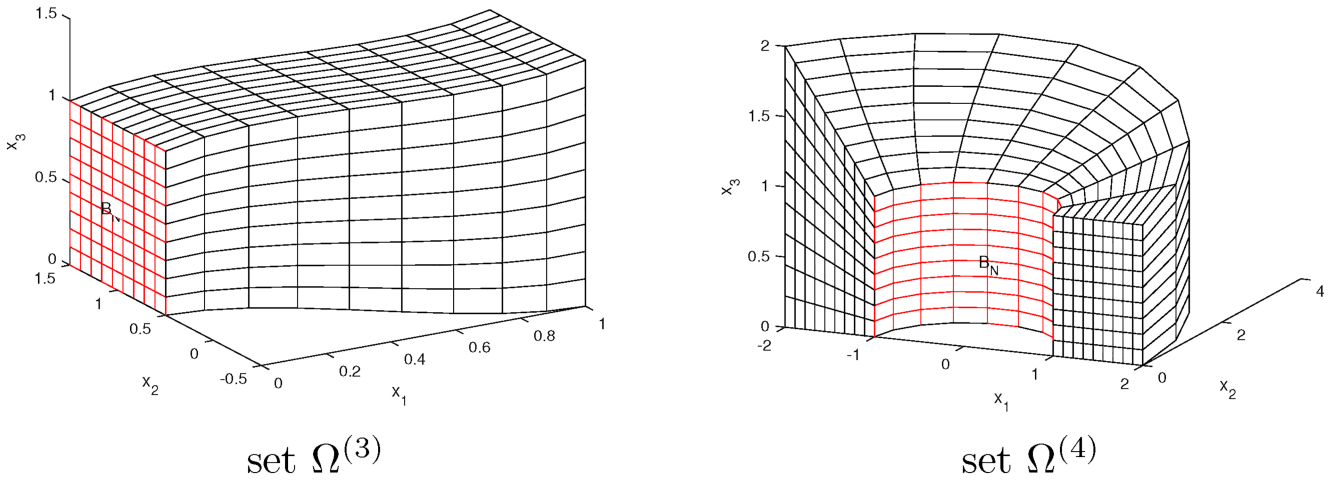

FIGURE 4 | The sets of points in $\Omega^{(3)}$ and $\Omega^{(4)}$; the red part of the boundary shows the set B_N where the Neumann condition is prescribed.

TABLE 3 | The mean square error E in the numerical solution of boundary value problems for equations (50), (51).

	N	Equation (50)	Equation (51)
$\Omega^{(3)}$	10	2.2(−4)	8.7(−3)
	20	1.1(−5)	9.3(−3)
	40	5.8(−7)	9.6(−3)
$\Omega^{(4)}$	10	1.9(−3)	5.2(−2)
	20	1.5(−4)	5.6(−2)
	40	9.3(−6)	5.8(−2)

Note: The notation $x(y)$ stays for $x \cdot 10^y$.

the convergence properties of the proposed method need further investigation, as well as the efficient computation of smooth curves $\mathbf{g}_j, j = 1, 2, \ldots, J$, in the case of discretization points $\mathcal{N}$ having the structure of meshes and sparse points. In particular, this last advance would allow exploiting the flexibility of the proposed method to obtain general algorithms for the numerical solution of partial differential equations.

5 | Conclusions

We have proposed a numerical method for the solution of partial differential equations. This method is based on a numerical differentiation procedure that makes use of standard approximation techniques like curve interpolation, numerical derivation of univariate functions, and numerical solution of linear systems. So, the proposed method can be easily generalized to the solution of boundary value problems in complex domains or in high-dimensional spaces. However, the concrete implementation of this method is based on the construction of curves $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, considered in Theorem 1; these curves can be easily obtained in the case of the discretized domain $\mathcal{N}$ having a grid structure. On the other hand, in the cases of meshes and sparse points, the computation of such curves can be in principle connected to routing problems on the graph G associated with $\mathcal{N}$. Standard problems like the Chinese postman problem or the Eulerian path are analyzed with a negative outcome, if we do not perform some refinement on the obtained solution. The main difficulty with such graph problems is their independence of the position of vertices with the consequent low geometric regularity of the corresponding solution.

The paper has also presented a numerical test of the proposed method, where the numerical solutions of boundary value problems, for elliptic differential equations in two-dimensional and three-dimensional domains, are considered. The results obtained in this numerical experiment are satisfactory and they encourage us to deepen the study of this method. In particular, different aspects of the proposed method deserve further analysis. The first and most important one is the computation of curves $\mathbf{g}_j$, $j = 1, 2, \ldots, J$, in the case of meshes and sparse points through specific graph routing problems or direct curve-fitting approaches. Another important study is the analysis of the convergence properties of the proposed method.

Acknowledgments

Nadaniela Egidi, Josephin Giacomini and Pierluigi Maponi are members of the Gruppo Nazionale Calcolo Scientifico–Istituto Nazionale di Alta Matematica (GNCS-INδAM).

This research has been accomplished within Rete ITaliana di Approssimazione (RITA), the thematic group on "Approximation Theory and Applications" of the Italian Mathematical Union.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data are available in the article.

References

1. A. Ü. Keskin, *Boundary Value Problems for Engineers With MATLAB Solutions* (Springer Nature, 2019).

2. A. Slavova, ed., *Proceedings of New Trends in the Applications of Differential Equations in Sciences - NTADES2023, Saints Constantine and Helena, Bulgaria, July 17-20*. Springer Proceedings in Mathematics & Statistics, vol. 449 (Springer Nature, 2024).

3. Q. Xu, Y. Shi, J. L. Bamber, C. Ouyang, and X. X. Zhu, "Large-Scale Flood Modeling and Forecasting With Floodcast," *Water Research* 264 (2024): 122162.

4. E. Nadaniela, G. Eleonora, G. Josephin, M. Pierluigi, and S. Lorenza, "A Numerical Solution of Richards Equation: A Simple Method Adaptable in Parallel Computing," *International Journal of Computer Mathematics* 97, no. 1–2 (2020): 2–17.

5. A. K. H. Dewedar, D. Palumbo, and M. Pepe, "Hydraulic Risk Assessment on Historic Masonry Bridges Using Hydraulic Open-Source Software and Geomatics Techniques: A Case Study of the "Hannibal Bridge" Italy," *Remote Sensing* 16, no. 16 (2024): 2994.

6. C. Zhu, C. Quan, P. Shi, B. Li, J. Zhang, and M. Li, "Analysis and Optimization of a Medium-Depth Ground Source Heat Pump Heating Systems With Heat Storage and Borehole Heat Exchangers," *Energy Reports* 11 (2024): 1455–1471.

7. N. Egidi, J. Giacomini, and P. Maponi, "Mathematical Model to Analyze the Flow and Heat Transfer Problem in u-Shaped Geothermal Exchangers," *Applied Mathematical Modelling* 61 (2018): 83–106.

8. J. Giacomini, C. Invernizzi, P. Maponi, et al., "Testing a Model of Flow and Heat Transfer for u-Shaped Geothermal Exchangers," *Advanced Modeling and Analysis A* 55, no. 3 (2018): 151–157.

9. N. Egidi, J. Giacomini, and P. Maponi, "Inverse Heat Conduction to Model and Optimise a Geothermal Field," *Journal of Computational and Applied Mathematics* 423 (2023): 114957.

10. J. S. Mustafa and D. K. Mawlood, "Mathematical Modelling for Groundwater Management for Multilayers Aquifers (Erbil Basin)," *Ain Shams Engineering Journal* 15, no. 7 (2024): 102781.

11. K. M. Moroney, W. T. Lee, S. B. G. O'Brien, F. Suijver, and J. Marra, "Coffee Extraction Kinetics in a Well Mixed System," *Journal of Mathematics in Industry* 7, no. 1 (2024): 3.

12. S. Angeloni, J. Giacomini, P. Maponi, et al., "Computer Percolation Models for Espresso Coffee: State of the Art, Results and Future Perspectives," *Applied Sciences* 13 (2023): 2688.

13. J. Giacomini, P. Maponi, and A. Perticarini, "CMMSE: A Reduced Percolation Model for Espresso Coffee," *Journal of Mathematical Chemistry* 61, no. 3 (2023): 520–538.

14. D. N. Ku, "Blood Flow in Arteries," *Annual Review of Fluid Mechanics* 29 (1997): 399–434.

15. I. Tominec, P.-F. Villard, E. Larsson, V. Bayona, and N. Cacciani, "An Unfitted Radial Basis Function Generated Finite Difference Method Applied to Thoracic Diaphragm Simulations," *Journal of Computational Physics* 469 (2022): 111496.

16. F. Schettini, M. V. De Bonis, C. Strina, et al., "Computational Reactive-Diffusive Modeling for Stratification and Prognosis Determination of Patients With Breast Cancer Receiving Olaparib," *Scientific Reports* 13, no. 1 (2023): 11951.

17. A. A. I. Quiroga, G. A. Torres, D. Fernández, and C. V. Turner, "Nonlinear Optimization for a Tumor Invasion PDE Model," *Computational and Applied Mathematics* 37, no. 1 (2018): 485–499.

18. M. C. Delfour and A. Garon, "Quadratic Ode and PDE Models of Drug Release Kinetics From Biodegradable Polymers," in *System Modeling and Optimization*, ed. D. Hömberg and F. Tröltzsch (Springer Berlin Heidelberg, 2013), 13–24.

19. E. Boger and O. Wigström, "A Partial Differential Equation Approach to Inhalation Physiologically Based Pharmacokinetic Modeling," *CPT: Pharmacometrics & Systems Pharmacology* 7, no. 10 (2018): 638–646.

20. T. Liszka and J. Orkisz, "The Finite Difference Method at Arbitrary Irregular Grids and Its Application in Applied Mechanics," *Computers and Structures* 11, no. 1 (1980): 83–95.

21. S. A. Chin, "A Unified Derivation of Finite-Difference Schemes From Solution Matching," *Numerical Methods for Partial Differential Equations* 32, no. 1 (2016): 243–265.

22. O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu, *The Finite Element Method: Its Basis and Fundamentals* (Elsevier, 2005).

23. Q. Zhang and C. Cui, "Condensed Generalized Finite Element Method," *Numerical Methods for Partial Differential Equations* 37, no. 3 (2021): 1847–1868.

24. R. Eymard, T. Gallouët, and R. Herbin, "Finite Volume Methods," in *Solution of Equation in* $\mathbb{R}^n$ *(Part 3), Techniques of Scientific Computing (Part 3)*. Handbook of Numerical Analysis, vol. 7 (Elsevier, 2000), 713–1018.

25. Y. Zhang and X. Liu, "A Posteriori Error Analysis of a Quadratic Finite Volume Method for Nonlinear Elliptic Problems," *Numerical Methods for Partial Differential Equations* 38, no. 1 (2022): 48–64.

26. S. Širca and M. Horvat, *Spectral Methods for PDE* (Springer, 2012), 575–620.

27. N. Egidi and P. Maponi, "A Spectral Method for the Solution of Boundary Value Problems," *Applied Mathematics and Computation* 409 (2021): 125812.

28. J. Lu, Y. Jiang, C.-W. Shu, and M. Zhang, "Analysis of a Class of Spectral Volume Methods for Linear Scalar Hyperbolic Conservation Laws," *Numerical Methods for Partial Differential Equations* 40, no. 6 (2024): e23126.

29. M. Griebel and M. A. Sehweitzer, eds., "Meshfree Methods for Partial Differential Equations," in *Lecture Notes in Computational Science and Engineering*, vol. 26 (Springer-Verlag, 2003).

30. J. Yang, H. Hu, Y. Koutsawa, and M. Potier-Ferry, "Taylor Meshless Method for Solving Non-Linear Partial Differential Equations," *Journal of Computational Physics* 348 (2017): 385–400.

31. N. Egidi and P. Maponi, "The Singular Value Expansion of the Volterra Integral Equation Associated to a Numerical Differentiation Problem," *Journal of Mathematical Analysis and Applications* 460 (2018): 656–681.

32. N. Egidi, J. Giacomini, and P. Maponi, "A Fredholm Integral Operator for the Differentiation Problem," *Computational and Applied Mathematics* 41, no. 5 (2022): 220.

33. N. Egidi, J. Giacomini, P. Maponi, and M. Youssef, "An FFT Method for the Numerical Differentiation," *Applied Mathematics and Computation* 445 (2023): 127856.

34. N. Egidi, J. Giacomini, P. Maponi, and M. Youssef, "Nded-Numerical Derivatives From Equispaced Data," *Journal of Approximation Software* 1, no. 1 (2024): 39–49.

35. H. A. Eiselt, M. Gendeaeu, and G. G. Laporte, "Arc Routing Problems, Part 1: The Chinese Postman Problem," *Operations Research* 43, no. 2 (1995): 231–242.

36. H. Fleischner, *Eulerian Graphs and Related Topics*, vol. 2 (Elsevier, 1991).

37. V. Britanak, K. R. Rao, and P. Yip, *Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations* (Academic Press-Elsevier, 2007).

38. N. Egidi, J. Giacomini, and P. Maponi, "A Fast Algorithm for Numerical Differentiation From Scattered Data," in *International Conference on Numerical Computations: Theory and Algorithms* (Springer, 2023), 238–245.

39. N. Egidi, J. Giacomini, and P. Maponi, "Fast Numerical Derivatives of Univariate Functions on Non-Uniform Grids," *Journal of Computational and Applied Mathematics* 467 (2025): 116619.

40. Y. G. Reshetnyak, "On Taylor's Formula for Functions of Several Variables," *Siberian Mathematical Journal* 54 (2013): 566–573.

41. Y. Saad and M. H. Schultz, "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal on Scientific and Statistical Computing* 7, no. 3 (1986): 856–869.

42. Matlab, https://it.mathworks.com.

43. G. Xu and G. Liu, "Development of Irregular-Grid Finite Difference Method (IFDM) for Governing Equations in Strong Form," *WSEAS Transactions on Mathematics* 5 (2006): 1117.

44. A. C. Albuquerque-Ferreira, M. Ureña, and H. Ramos, "The Generalized Finite Difference Method With Third- and Fourth-Order Approximations and Treatment of Ill-Conditioned Stars," *Engineering Analysis With Boundary Elements* 127 (2021): 29–39.

45. G. Eriksson and V. Stiernström, "Acoustic Shape Optimization Using Energy Stable Curvilinear Finite Differences," *Journal of Computational Physics* 517 (2024): 113347.