Appl. Numer. Math. 210 (2025) 165–176

Contents lists available at ScienceDirect

Applied Numerical Mathematics

journal homepage: www.elsevier.com/locate/apnum

Research Paper

Numerical derivation of multivariate functions

Nadaniela Egidi *, Josephin Giacomini, Pierluigi Maponi

University of Camerino, School of Science and Technology, via Madonna delle Carceri 9, Camerino (MC), 62032, Italy

ARTICLE INFO

Keywords:
Partial derivatives discretization
Numerical differentiation
Errors bound
Singular value expansion

ABSTRACT

We consider the problem of the numerical derivation of a function of several real variables. The proposed numerical method is based on the singular value expansion of the integral formulation of the derivative problem generalised to the multivariate case. The resulting derivation method is able to compute the partial derivatives of a multivariate function sampled at points in general position. The accuracy of the proposed method is analysed and confirmed by numerical tests performed for different distributions of the sampling points.

1. Introduction

We consider the problem of the numerical derivation of a multivariate function known at a given sample data set $\mathcal{N}$. This is a fundamental problem in any research field where we need to know an approximation of the derivatives of differentiable functions known only at discrete sampled points [1,2]. Even if the exact expressions of the derivatives are known, numerical derivation is preferable due to the computational cost reason. For example, in nonlinear optimisation, gradient approximations are necessary to construct descent directions and to verify optimality conditions. Moreover, many application problems are described by Ordinary Differential Equations (ODEs) and Partial Differential Equations (PDEs), which need accurate approximations of both the solutions and their derivatives [3–8].

Due to its importance, many methods have been presented for the numerical differentiation of a function. There are methods based on finite differences [9], polynomial interpolation [10–12], regularization methods [13–15], and methods based on complex variables techniques. These last techniques were first proposed in [16] and used in [17] to approximate the first derivative avoiding the subtractive cancellation errors. Additionally, they have been analysed in noisy environments [18].

Among these numerical differentiation methods, we have different orders of the approximations. Unfortunately, the numerical differentiation is an ill-conditioned problem and the choice of the step h is crucial for the accuracy of the computed approximation; in particular, h has to be related to the noise level [19].

In this work, we consider the computation of the first derivatives of a multivariate function starting from its values at a set of nodes $\mathcal{N}$ not necessarily structured. The proposed method is based on an algorithm for univariate function that we prove to have a convergence order of $\mathcal{O}(h^4)$. This method can be extended to solve fractional differentiation problems, in fact, it is based on an integral formulation of the derivation problem and its Singular Value Decomposition (SVD) [20]. Hence, a similar approach can be considered for the integral formulation of the fractional derivation problem. Ultimately, the proposed derivation method and its generalisation to fractional derivatives can be profitably used to solve PDEs and fractional PDEs, like those studied in [21–23]. It is worth noting that the simple tools used, i.e., the curve fitting, one-dimensional derivation methods and linear system solution, give

* Corresponding author.
E-mail addresses: nadaniela.egidi@unicam.it (N. Egidi), josephin.giacomini@unicam.it (J. Giacomini), pierluigi.maponi@unicam.it (P. Maponi).

https://doi.org/10.1016/j.apnum.2024.12.011
Received 18 July 2024; Received in revised form 21 October 2024; Accepted 23 December 2024

Available online 31 December 2024

high flexibility to the proposed numerical derivation method. This flexibility makes the method profitably usable in solving PDEs on complicated domains, also in consideration of the satisfactory accuracy results. Moreover, the performed numerical tests confirm the accuracy and robustness of the proposed method.

Section 2 briefly introduces the numerical derivation method in the univariate case, since it is a fundamental component in the following method for the multivariate case. Section 3 presents the numerical method for the approximation of the partial derivatives of a multivariate function. Section 4 reports the numerical results obtained in a numerical experiment with the proposed method. Section 5 provides some conclusions and remarks for future investigations.

2. Numerical derivation in the univariate case

We begin by giving some used notations. We denote with $\mathbb{N}$ the set of natural numbers. Let $\mathbb{R}$ be the set of real numbers, and $\mathbb{R}^n$ be the n-dimensional real Euclidean space. An element in $\mathbb{R}^n$ is a column vector of the following form $\mathbf{x} = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$, where the superscript T means transposed. We denote with $\mathbb{R}^{m\times n}$ the spaces of real matrices having m rows and n columns. We denote with $||\cdot||_\infty$ the infinity norm of a matrix or a vector depending on the entry. We begin by giving some analytic results on derivatives approximation.

2.1. Analytic results

Given a differentiable function $G : [0,1] \to \mathbb{R}$, the first derivative G' of G, satisfies

$$\int_0^t G'(s)ds = G(t) - G(0), \tag{1}$$

that is, G' is the solution of the Fredholm integral equation having kernel $K : [0,1]\times[0,1] \to \mathbb{R}$,

$$K(t,s) = \begin{cases} 1, & s < t, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The singular value expansion (SVE) of K is given by the following,

$$K(t,s) = \sum_{j=1}^{\infty} \mu_j u_j(t) v_j(s), \qquad 0 \le t, s \le t, \tag{3}$$

where, for $j = 1, 2, \dots$, and $t \in [0,1]$, $u_j(t)$ and $v_j(t)$ are the singular functions, and μ_j is the corresponding singular value of the integral operator with kernel (2) associated to the first derivative operator; moreover, for $t \in [0,1]$, and $j = 1, 2, \dots$, we have:

$$u_j(t) = \sqrt{2}\sin(\gamma_j t), \qquad v_j(t) = \sqrt{2}\cos(\gamma_j t), \qquad \gamma_j = \pi\left(j - \frac{1}{2}\right),$$

and $\mu_j = 1/\gamma_j$, see [24] for details. By using this SVE, we obtain the following representation for G' [20],

$$G'(t) = \frac{1}{2}\left(\sum_{j=1}^{\infty} \langle G', v_j\rangle v_j(t) + \sum_{j=1}^{\infty} \langle G', u_j\rangle u_j(t)\right), \quad 0 < t < 1, \tag{4}$$

moreover

$$\langle G', v_j\rangle = -\sqrt{2}G(0) + \gamma_j \langle G, u_j\rangle, \tag{5}$$

$$\langle G', u_j\rangle = (-1)^{j-1}\sqrt{2}G(1) - \gamma_j \langle G, v_j\rangle, \tag{6}$$

where $\langle\cdot,\cdot\rangle$ denotes the usual inner product.

From the discretization of (4), (5) and (6), we obtain numerical schemes for computing the derivative of G, like the one proposed in [25]. The following section proposes another scheme, which is used in the subsequent section for the numerical approximation of partial derivatives.

2.2. The numerical approximation scheme

We suppose to know $G_l = G(t_l)$, $t_l = l/L$, $l = 0, 1, \dots, L$, $L \ge 4$, we denote with $\delta = 1/L$ the discretization step. In the following, we give a method for computing an approximation d_l of the derivative of G at t_l, that is, $d_l \approx G'(t_l)$, for $l = 0, 1, \dots, L$.

Let be $\mathbf{w} = (w_1, w_2, \dots, w_M)^T \in \mathbb{R}^M$, we denote with $FCT^{(2)}(\mathbf{w}) \in \mathbb{R}^M$ and $FCT^{(3)}(\mathbf{w}) \in \mathbb{R}^M$ its discrete cosine transform of type II and III, respectively, we denote with $FST^{(3)}(\mathbf{w}) \in \mathbb{R}^M$ its discrete sine transform of type III. More precisely, the generic mth components, $m = 1, 2, \dots, M$, of such transformed vectors are defined as follows:

$$\left(FCT^{(2)}(\mathbf{w})\right)_m = \frac{1}{\sqrt{M}}\sum_{r=1}^{M} w_r \cos\left(\frac{\pi}{M}\left(r - \frac{1}{2}\right)(m-1)\right), \tag{7}$$

$$\left(FCT^{(3)}(\mathbf{w})\right)_m = \frac{1}{\sqrt{M}}\left(w_1 + 2\sum_{r=2}^{M} w_r \cos\left(\frac{\pi}{M}(r-1)\left(m-\frac{1}{2}\right)\right)\right), \tag{8}$$

$$\left(FST^{(3)}(\mathbf{w})\right)_m = \frac{1}{\sqrt{M}}\left(2\sum_{r=1}^{M-1} w_r \sin\left(\frac{\pi}{M} r\left(m-\frac{1}{2}\right)\right) + (-1)^{m-1} w_M\right), \tag{9}$$

see [26] for a detailed description of discrete trigonometric transformations.

Let $M = L+1$, for $j = 1, 2, \dots, M$, we define

$$\tilde{c}_{j,k} = \cos\left(\gamma_j \frac{k}{M}\right), \quad \tilde{s}_{j,k} = \sin\left(\gamma_j \frac{k}{M}\right), \quad k \in \mathbb{Z},$$

and, for $l = 0, 1, \dots, L$,

$$\beta_l = \begin{cases} 1, & l = 0, \\ 2 & l \geq 1. \end{cases}$$

Theorem 1 *(Numerical derivative of univariate functions). Suppose that the function $G : [0,1] \to \mathbb{R}$ is sampled at $t_l = l\delta$, $l = 0, 1, \dots, L$, $\delta = 1/L$, and let $\{G_0, G_1, \dots, G_L\}$ be the corresponding data set. For $l = 0, 1, \dots, L$, we can compute the approximation d_l of $G'(t_l)$ by using the following formula*

$$d_l = \beta_l \left(FCT^{(2)}(\mathbf{w})\right)_{l+1}, \tag{10}$$

where $\mathbf{w} = (w_1, w_2, \dots, w_{L+1})^T \in \mathbb{R}^{L+1}$ has components

$$\begin{aligned} w_j = &\left(FCT^{(3)}\left(\mathbf{a}^{c,0}\right)\right)_j + \tilde{c}_{j,1}\left(FCT^{(3)}\left(\mathbf{a}^{c,1}\right)\right)_j + \tilde{c}_{j,2}\left(FCT^{(3)}\left(\mathbf{a}^{c,2}\right)\right)_j + \\ &+\tilde{s}_{j,1}\left(FST^{(3)}\left(\mathbf{a}^{s,1}\right)\right)_j + \tilde{s}_{j,2}\left(FST^{(3)}\left(\mathbf{a}^{s,2}\right)\right)_j, \ j = 1, 2, \dots, L+1, \end{aligned} \tag{11}$$

and

$$\mathbf{a}^{c,i} = \left(a_0^{c,i}, a_1^{c,i}, \dots, a_L^{c,i}\right)^T \in \mathbb{R}^M, \quad i = 0, 1, 2,$$
$$\mathbf{a}^{s,i} = \left(a_0^{s,i}, a_1^{s,i}, \dots, a_L^{s,i}\right)^T \in \mathbb{R}^M, \quad i = 1, 2,$$

are given by (A.1)-(A.5) and depends only on G_l, $l = 0, 1, \dots, L$.

For the approximation (10) we have:

$$d_l = G'(t_l) + \mathcal{O}(\delta^4), \qquad l = 0, 1, \dots, L. \tag{12}$$

Proof. See Appendix B. □

As a consequence of this theorem, formulas (10) and (11) allow the computation of the fourth order approximation d_l of $G'(t_l)$, $l = 0, 1, \dots, L$, by knowing the values of G at t_l, $l = 0, 1, \dots, L$; these data are contained in the vectors $\mathbf{a}^{\cdot,\cdot}$, whose appropriate refinements can produce higher order approximation schemes. In the following section, we use this numerical method for computing the partial derivatives of multivariate functions.

3. Numerical derivation in the multivariate case

We describe a procedure generalizing the numerical derivation method described in Section 2 to the multivariate case. The formula (10) has an analogue version in the multivariate case; in particular, let $f : \Omega \subseteq \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function up to order $k \geq 1$, let be $\mathbf{x} \in \Omega$ and $\mathbf{h} \in \mathbb{R}^n$ a non-null vector, such that $\mathbf{x} + t\mathbf{h} \in \Omega$ for every $t \in [0,1]$. Then for $t \in [0,1]$,

$$f(\mathbf{x}+t\mathbf{h}) = \sum_{|\boldsymbol{\alpha}|<k} \frac{\partial^{\boldsymbol{\alpha}} f(\mathbf{x})}{\boldsymbol{\alpha}!} t^{|\boldsymbol{\alpha}|}\mathbf{h}^{\boldsymbol{\alpha}} + k\sum_{|\boldsymbol{\alpha}|=k} \frac{\mathbf{h}^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\int_0^t (t-s)^{k-1}\partial^{\boldsymbol{\alpha}} f(\mathbf{x}+s\mathbf{h})ds, \tag{13}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_n) \in \mathbb{N}^n$ is a multi-index, $|\boldsymbol{\alpha}| = \alpha_1 + \alpha_2 + \cdots + \alpha_n$, $\boldsymbol{\alpha}! = \alpha_1!\alpha_2! \cdots \alpha_n!$, $\mathbf{h}^{\boldsymbol{\alpha}} = \prod_{i \in I_{\mathbf{h}}} h_i^{\alpha_i}$, $I_{\mathbf{h}}$ is the set of the indices of the non-null Cartesian components of $\mathbf{h}$, and $\partial^{\boldsymbol{\alpha}}$ we denote the derivative operator associated to $\boldsymbol{\alpha}$, that is

$$\partial^{\boldsymbol{\alpha}} f = \begin{cases} \dfrac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}\dots\partial x_n^{\alpha_n}} & |\boldsymbol{\alpha}| \neq 0, \\ f, & |\boldsymbol{\alpha}| = 0, \end{cases}$$

with the understanding that if $\alpha_i = 0$ then there is no partial derivation with respect to variable x_i, see [27] for a detailed discussion on formula (13). Given the function f and its derivatives up to the order $k-1$, formula (13) can be seen as an integral equation for

kth order derivatives of f. As in the univariate case, we restrict our attention to the first order derivative, i.e. $k = 1$, in this case (13) becomes

$$f(\mathbf{x}+t\mathbf{h}) = f(\mathbf{x}) + \sum_{i=1}^{n} h_i \int_0^t \frac{\partial f}{\partial x_i}(\mathbf{x}+s\mathbf{h})ds,\ t\in[0,1]. \tag{14}$$

We note that, given $\mathbf{x}\in\Omega$ and $\mathbf{h}\in\mathbb{R}^n$, formula (14) can be easily written in terms of function $F(t) = f(\mathbf{x}+t\mathbf{h})$, $t\in[0,1]$, which depends on $\mathbf{x}$ and $\mathbf{h}$. In fact, (14) is equivalent to

$$F(t) = F(0) + \int_0^t F'(s)ds,\ t\in[0,1], \tag{15}$$

which is analogous to equation (1); so, the numerical method described in Section 2 can be used to approximate $F'(t)$, $t\in[0,1]$ and in turn to approximate the partial derivatives of f in a neighbourhood of $\mathbf{x}\in\Omega$. However formulas (14) and (15) are not flexible enough to deal efficiently with the numerical derivation concerning a generic set of points. To this aim, we extend these formulas by considering a generic continuously differentiable function, $\mathbf{g} = (g_1, g_2, \dots, g_n)^T : [0,1]\to\Omega$, and by defining $G(t) = f(\mathbf{g}(t))$, $t\in[0,1]$. In this way, by rewriting equation (15) for G, we obtain

$$f(\mathbf{g}(t)) = f(\mathbf{g}(0)) + \sum_{i=1}^{n} \int_0^t g_i'(s)\frac{\partial f}{\partial x_i}(\mathbf{g}(s))ds,\ t\in[0,1]. \tag{16}$$

For example, given $L+1$ distinct points, $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_L \in \Omega$, we can compute a continuously differentiable function, $\mathbf{g} : [0,1]\to\Omega$, such that $\mathbf{g}(\tilde{t}_l) = \mathbf{p}_l$, $l = 0,1,\dots,L$, $0 = \tilde{t}_0 < \tilde{t}_1 < \dots < \tilde{t}_L = 1$, that is, $\mathbf{g}$ interpolates the $L+1$ points, $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_L \in \Omega$, at interpolation knots, $\tilde{t}_0, \tilde{t}_1, \dots, \tilde{t}_L$. Hence, given $G(t) = f(\mathbf{g}(t))$, from the chain rule, we have

$$G'(\tilde{t}_l) = \sum_{i=1}^{n} g_i'(\tilde{t}_l)\frac{\partial f}{\partial x_i}(\mathbf{p}_l),\ l = 0,1,\dots,L, \tag{17}$$

and from a numerical derivatives d_l,

$$d_l \approx G'(\tilde{t}_l), \qquad l = 0,1,\dots,L, \tag{18}$$

of the univariate function G, we have:

$$d_l \approx \sum_{i=1}^{n} g_i'(\tilde{t}_l)\frac{\partial f}{\partial x_i}(\mathbf{p}_l),\ l = 0,1,\dots,L. \tag{19}$$

In the following, we describe how to use equation (19) to approximate the partial derivatives $\frac{\partial f}{\partial x_i}(\mathbf{p}_l)$, $l = 0,1,\dots,L$, $i = 1,2,\dots,n$, but first we give an example that shows the quality of the approximation (19), when d_l, $l = 0,1,\dots,L$, are computed by the method described in Section 2. In particular, the function $\mathbf{g}$ is chosen as the natural cubic spline interpolating some points used to sample the function. We recall that the cubic spline is a piecewise cubic polynomial that fits the considered points and is twice continuously differentiable.

Example 1. We consider a two-dimensional case, where $\Omega = [0,1]\times[0,e]\subset\mathbb{R}^2$, and the function, $f(\mathbf{x}) = \sin(x_1x_2)\in\mathbb{R}$, $\mathbf{x} = (x_1,x_2)^T\in\Omega$, is sampled at points $\mathbf{p}_l = (t_l^2, \sin(\pi t_l)e^{t_l})^T$, $t_l = l\delta$, $l = 0,1,\dots,L$, and $\delta = 1/L$, see Fig. 1 for a pictorial description of such points. Let $\mathbf{g} : [0,1]\to\mathbb{R}^2$ be the cubic spline that interpolates these points, $G(t) = f(\mathbf{g}(t))$, $t\in[0,1]$. For $l = 0,1,\dots,L$, $G'(l/L)$ is computed by using (17) and the approximation d_l is computed by (10). In Table 1 we have reported the mean squared error E, for $L = 10, 100, 1000$, that is

$$E = \sqrt{\frac{1}{L+1}\sum_{l=0}^{L}\left(d_l - G'(l/L)\right)^2}, \tag{20}$$

and the values of E/δ^4, which confirm the theoretical convergence rate stated in Theorem 1.

This example shows that the method described in Section 2 can provide an accurate approximation (18) to use in (19).

However, (19) does not provide a straightforward approximation of the partial derivatives of the function f at $\mathbf{p}$, but only an approximation of the scalar product of the gradient $\nabla f(\mathbf{p})$ with the vector $\boldsymbol{\tau}(\bar{t})$ associated to the curve $\mathbf{g}$, where

$$\boldsymbol{\tau}(t) = \left(g_1'(t), g_2'(t), \dots, g_n'(t)\right)^T,\ t\in[0,1], \tag{21}$$

is the tangent vector to the curve $\mathbf{g}(t)$, $t\in[0,1]$, in $\mathbb{R}^n$, and $\bar{t}\in[0,1]$ is such that $\mathbf{g}(\bar{t}) = \mathbf{p}$.

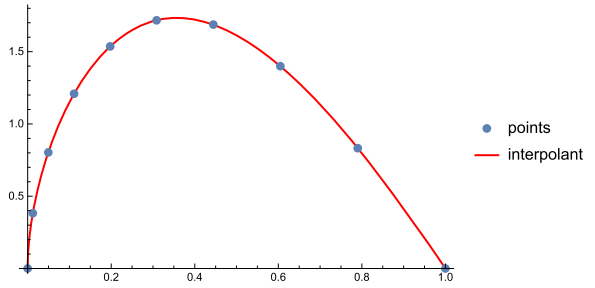

Fig. 1. Points $\mathbf{p}_l$, $l = 0, 1, \dots, L$, where $L = 10$, and the corresponding spline interpolating function $\mathbf{g}$.

Table 1
The mean squared error E and its convergence order E/δ^4, for the approximation (18) computed by the method described in Section 2 for $L = 10, 100, 1000$. The notation $x(y)$ stays for $x \cdot 10^y$.

L	E	E/δ^4
10	6.6(-2)	0.7(3)
100	7.2(-5)	7.2(3)
1000	2.7(-9)	2.7(3)

We remark that the partial derivatives of f at a point $\mathbf{p} \in \Omega$ can be computed by considering several curves $\mathbf{g}_j : [0,1] \to \Omega$, $\mathbf{g}_j = (g_{1,j}, g_{2,j}, \dots, g_{n,j})^T$, $j \in J_{\mathbf{p}}$ (set of indices), interpolating the point $\mathbf{p}$ and such that the following linear system arising from (17) has a unique solution

$$G'_j(\tilde{t}) = \sum_{i=1}^{n} g'_{i,j}(\tilde{t}) \frac{\partial f}{\partial x_i}(\mathbf{p}), \quad \tilde{t} \in T^{j,\mathbf{p}}, j \in J_{\mathbf{p}}, \tag{22}$$

where $G_j = f \circ \mathbf{g}_j$, $T^{j,\mathbf{p}} = \{t \in [0,1] \ : \ \mathbf{g}_j(t) = \mathbf{p}\}$. When $j \in J_{\mathbf{p}}$, we have that $T^{j,\mathbf{p}}$ has at least one element; on the other hand, when $J_{\mathbf{p}}$ has only one element, the unique curve g_j, $j \in J_{\mathbf{p}}$ must be self-intersecting in $\mathbf{p}$ at least n times and $T^{j,p}$ has at least n elements. We note that the solvability of the linear system (22) depends on the properties of the tangent vectors to the curves, $\boldsymbol{\tau}_j(t) = (g'_{1,j}(t), g'_{2,j}(t), \dots, g'_{n,j}(t))^T$, $t \in [0,1]$, $j \in J_{\mathbf{p}}$ at point $\mathbf{p}$. Moreover, the solution of (22) defines the partial derivatives of f at $\mathbf{p}$; the corresponding approximation of these derivatives can be computed by linear system (22), by considering approximation (18), that is by substituting $G'_j(\tilde{t})$ with $d_j(\tilde{t})$.

We precisely define this method by generalising the problem to the case of the numerical derivation at a set $\mathcal{N} \subset \Omega$ of N points. We suppose that f is sampled at $\mathcal{N}$ and the following problem can be solved. We denote with $C^1[0,1]$ the set of continuously differentiable curves $\mathbf{g} : [0,1] \to \Omega$.

Problem 1. Given f sampled at $\mathcal{N}$, find J curves $\mathbf{g}_j \in C^1[0,1]$, $j = 1, 2, \dots, J$, $\mathbf{g}_j = (g_{1,j}, g_{2,j}, \dots, g_{n,j})^T$, interpolating $N_j \geq 2$ points contained in $\mathcal{N}_j \subseteq \mathcal{N}$, such that

- for each point $\mathbf{p} \in \mathcal{N}$, curves $\mathbf{g}_j$, $j \in J_{\mathbf{p}} \subset \{1, 2, \dots, J\}$, interpolate $\mathbf{p}$, that is $\mathbf{p} \in \mathcal{N}_j$ and there exists at least an interpolation knot $\tilde{t} \in T^{j,\mathbf{p}}$ such that $\mathbf{g}_j(\tilde{t}) = \mathbf{p}$;
- for each point $\mathbf{p} \in \mathcal{N}$, the set of tangential vectors,

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

  contains n linearly independent vectors.

When Problem 1 is solved the evaluation of the derivatives of f can be easily done by taking into account approximation (19) and the following considerations. From the solution $\mathbf{g}_j(t)$, $t \in [0,1]$, $j = 1, 2, \dots, J$, we define $G_j(t) = f\left(\mathbf{g}_j(t)\right)$, $t \in [0,1]$, $j = 1, 2, \dots, J$. For each point $\mathbf{p} \in \mathcal{N}$ and $j \in J_{\mathbf{p}}$, let $d_j(\tilde{t})$ be the approximation of $G'_j(\tilde{t})$, $\tilde{t} \in T^{j,\mathbf{p}}$, then the numerical partial derivatives, $D_i(\mathbf{p}) \approx \frac{\partial f}{\partial x_i}(\mathbf{p})$, $i = 1, 2, \dots n$, of f at point $\mathbf{p} \in \mathcal{N}$ are computed by the solution of the following linear system:

$$\sum_{i=1}^{n} g'_{i,j}(\tilde{t}) D_i(\mathbf{p}) = d_j(\tilde{t}), \qquad \tilde{t} \in T^{j,\mathbf{p}}, \ j \in J_{\mathbf{p}}, \tag{23}$$

that is a linear system with n unknowns and $n_{\mathbf{p}}$ equations, where

$$n_{\mathbf{p}} = \sum_{j \in J_{\mathbf{p}}} n_{j,\mathbf{p}},$$

and $n_{j,\mathbf{p}}$ is the cardinality of $T^{j,\mathbf{p}}$. We note that $n_{\mathbf{p}}$ is equal to the cardinality of $J_{\mathbf{p}}$ if, for $j \in J_{\mathbf{p}}$, the set $T^{j,\mathbf{p}}$ has only one element. Moreover, the approximation $D_i(\mathbf{p})$, $i = 1, 2, \dots n$, of the partial derivatives $\frac{\partial f}{\partial x_i}(\mathbf{p})$ of f at $\mathbf{p} \in \mathcal{N}$ are well-defined by linear system (23), which has a unique solution as a consequence of the second point in Problem 1, which implies $n_{\mathbf{p}} \geq n$, and this solution coincides with the least square solution of (23) when $n_{\mathbf{p}} > n$. The following theorem gives an error estimation of the numerical derivatives computed by (23) in the special case $n_{\mathbf{p}} = n$ for the sake of simplicity.

Theorem 2. *If Problem 1 is solved with $n_{\mathbf{p}} = n$ for each $\mathbf{p} \in \mathcal{N}$, then linear systems (19) have order n and invertible coefficient matrices*

$$A^{\mathbf{p}} = \left(g'_{i,j}(\tilde{t}) \right)_{\tilde{t} \in T^{j,\mathbf{p}}, j \in J_{\mathbf{p}}; i=1:n}.$$

Moreover, if $C^{\mathbf{p}}$ is the condition number of $A^{\mathbf{p}}$ with respect to infinity norm, the proposed method gives an approximation $\mathcal{O}(C^{\mathbf{p}}\delta^4/||A^{\mathbf{p}}||_\infty)$, when the numerical derivatives $d_j(\tilde{t})$, $\tilde{t} \in T^{j,\mathbf{p}}$, $j \in J_{\mathbf{p}}$, in (23) are computed by (10) and

$$\delta = \max \left\{ \frac{1}{N_j - 1} \ : \ j \in J_{\mathbf{p}} \right\}.$$

Proof. Disregarding the interpolation error in the entries of matrix $A^{\mathbf{p}}$ the approximated gradient of f at $\mathbf{p}$,

$$\mathbf{D}(\mathbf{p}) = (D_1(\mathbf{p}), D_2(\mathbf{p}), \dots, D_n(\mathbf{p}))^T,$$
$$\mathbf{D}(\mathbf{p}) \approx \left(\frac{\partial f}{\partial x_1}(\mathbf{p}), \frac{\partial f}{\partial x_2}(\mathbf{p}), \dots, \frac{\partial f}{\partial x_n}(\mathbf{p}) \right)^T \in \mathbb{R}^n,$$

is the unique solution of

$$A^{\mathbf{p}}\mathbf{D}(\mathbf{p}) = \mathbf{d}(\mathbf{p}), \tag{24}$$

where

$$\mathbf{d}(\mathbf{p}) = \left(d_j(\tilde{t}) \right)_{\tilde{t} \in T^{j,\mathbf{p}}, j \in J_{\mathbf{p}}} \in \mathbb{R}^n, \quad d_j(\tilde{t}) = G'_j(\tilde{t}) + e_j(\mathbf{p}, \tilde{t}),$$

$G_j = f \circ \mathbf{g}_j$ and $e_j(\mathbf{p}, \tilde{t})$ is the error in the numerical derivation of functions $G_j(t)$ at $\tilde{t}$.

We note that, if $e_j(\mathbf{p}, \tilde{t}) = 0$ for each $\tilde{t} \in T^{j,\mathbf{p}}$ and $j \in J_{\mathbf{p}}$, the unique solution of (24) is $\mathbf{D}(\mathbf{p}) = \nabla f(\mathbf{p})$, the gradient of f at $\mathbf{p}$. Instead, if $d_j(\tilde{t}) \approx G'_j(\tilde{t})$, $\tilde{t} \in T^{j,\mathbf{p}}$, $j \in J_{\mathbf{p}}$, is computed by (10) with $G = G_j$ and $L = N_j - 1$, then, from (12) we have that

$$||\mathbf{e}(\mathbf{p})||_\infty = \max_{\tilde{t} \in T^{j,\mathbf{p}}, j \in J_{\mathbf{p}}} |e_j(\mathbf{p}, \tilde{t})| = \mathcal{O}(\delta^4).$$

From standard arguments on the perturbation theory in linear systems, we have

$$||\mathbf{D}(\mathbf{p}) - \nabla f(\mathbf{p})||_\infty \leq C^{\mathbf{p}} \frac{||\mathbf{e}(\mathbf{p})||_\infty}{||A^{\mathbf{p}}||_\infty},$$

which together with $||\mathbf{e}(\mathbf{p})||_\infty = \mathcal{O}(\delta^4)$ complete the proof. □

4. Numerical results

We present the results of a numerical experiment to test the performance of the proposed method in the two-dimensional case. We consider two functions given by

- $f_1(\mathbf{x}) = \sin(x_1 x_2)$,
- $f_2(\mathbf{x}) = \frac{x_1 + x_2}{x_2^2 + 1}$,

where $\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$. The set of points $\mathcal{N} \subset \mathbb{R}^2$, where these functions are sampled and where we want to compute their numerical derivatives, is chosen as the set of vertices of non-Cartesian grids; in particular, two different sets of points are considered:

- $\mathcal{N}^{(1)} = \{(u_h, \frac{1}{2}\sin(2\pi u_h) + v_k)^T,\ u_h = h/H,\ h = 0, 1, \dots, H,\ v_k = k/K,\ k = 0, 1, \dots, K\}$,
- $\mathcal{N}^{(2)} = \{((v_k + 1)\cos(\pi u_h), (v_k + 1 + \frac{1}{2}v_k \sin(\pi u_h))\sin(\pi u_h))^T,\ u_h = h/H,\ h = 0, 1, \dots, H,\ v_k = k/K,\ k = 0, 1, \dots, K\}$.

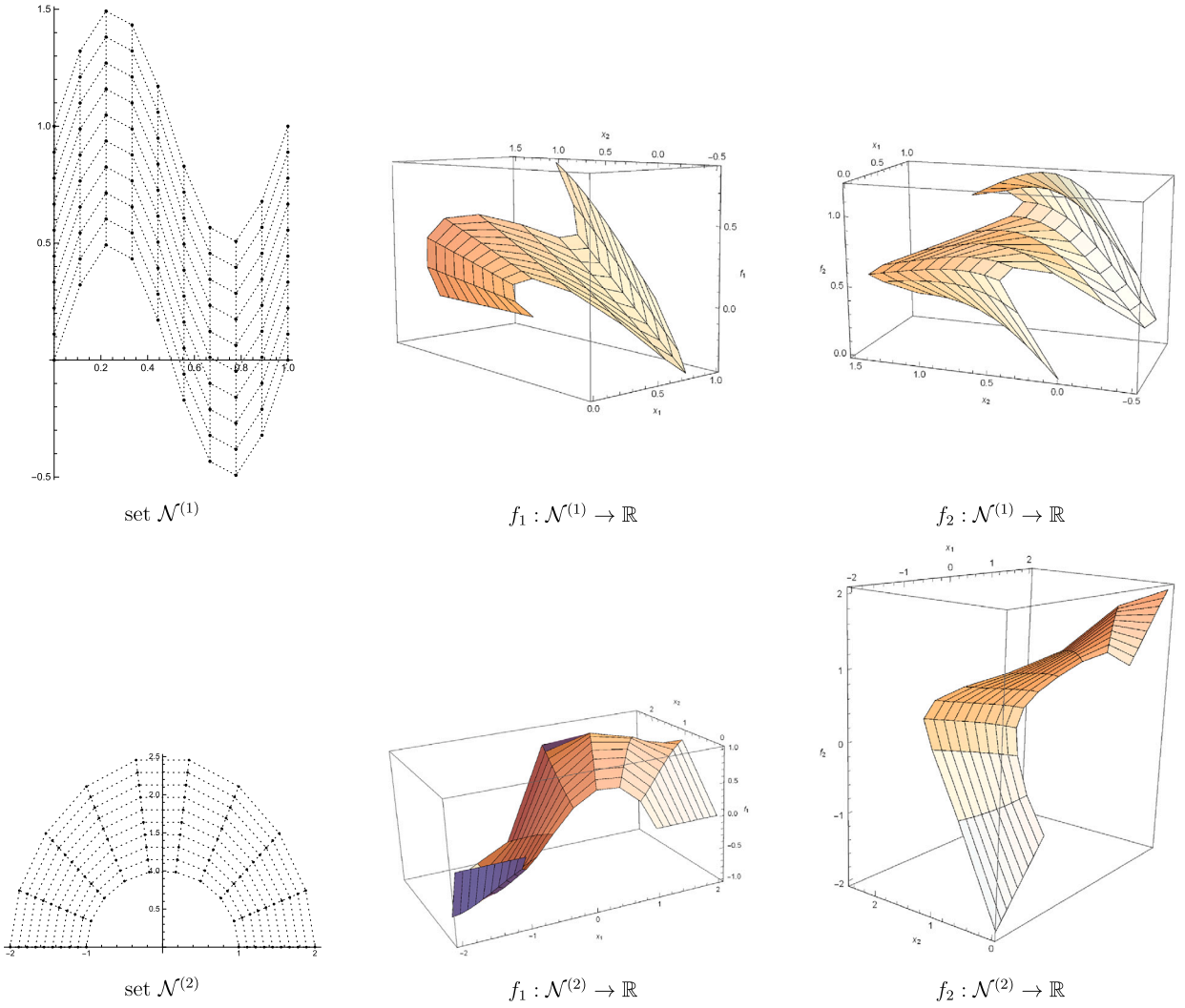

Fig. 2. The sets of points $\mathcal{N}^{(1)}$ and $\mathcal{N}^{(2)}$ for $H = K = 10$, and the corresponding values of functions f_1 and f_2. A surface is drawn through such sets of points for pictorial reasons.

These sets of points $\mathcal{N}^{(1)}$, $\mathcal{N}^{(2)}$, when $H = K = 10$, together the values of functions f_1, f_2 are shown in Fig. 2.

The numerical partial derivatives of functions f_1, f_2 are computed by the method discussed in the previous section and using the values of such functions sampled at points $\mathcal{N}^{(1)}$, $\mathcal{N}^{(2)}$. In particular, the use of grids $\mathcal{N}^{(1)}$, $\mathcal{N}^{(2)}$ makes easy the choice of functions $\mathbf{g}_j$, $j = 1, 2, \dots, J$, solving Problem 1. In particular, we denote with $\mathcal{N}$ a generic grid, where:

$$\mathcal{N} = \{(\xi_1(u_h, v_k), \xi_2(u_h, v_k))^T \in \mathbb{R}^2,\ u_h = h/H,\ h = 0, 1, \dots, H,$$
$$v_k = k/K,\ k = 0, 1, \dots, K\}. \tag{25}$$

For each $h = 0, 1, \dots, H$, we denote with $\boldsymbol{\chi}_h$, the cubic-spline function interpolating $(\xi_1(u_h,\ v_k),\ \xi_2(u_h,\ v_k))^T$, $k = 0, 1, \dots, K$, and for each $k = 0, 1, \dots, K$, we denote with $\boldsymbol{\kappa}_k$, the cubic-spline function interpolating $(\xi_1(u_h,\ v_k),\ \xi_2(u_h, v_k))^T$, $h = 0, 1, \dots, H$. Under standard arguments on geometry, when $(\xi_1(u, v),\ \xi_2(u, v))$ defines a regular surface, we have that the set of curves,

$$\{\mathbf{g}_j,\ j = 1, 2, \dots, J\} = \{\boldsymbol{\chi}_h,\ h = 0, 1, \dots, H\} \cup \{\boldsymbol{\kappa}_k,\ k = 0, 1, \dots, K\}, \tag{26}$$

solves Problem 1, in particular, for each $\mathbf{p} \in \mathcal{N}$ the set $J_{\mathbf{p}}$ has two elements and for each $j \in J_{\mathbf{p}}$ the set $T^{j,\mathbf{p}}$ has one element, hence $n_{\mathbf{p}} = 2$, $J = (H + 1) + (K + 1)$. In the proposed numerical experiment, the functions $\boldsymbol{\chi}_\cdot$ and $\boldsymbol{\kappa}_\cdot$ in (26) are computed by the cubic spline interpolating the corresponding set of points.

Table 2 shows the numerical results obtained with this method for N_j constant, in particular, $N_j - 1 = L = H = K = 10$, 50, 100. These results are in terms of the error in the numerical approximation of the derivatives. In particular, for $i = 1, 2$, $h = 0, 1, \dots, H$,

Table 2
The error parameters for the numerical derivatives of functions f_1 and f_2 on grids $\mathcal{N}^{(1)}$, $\mathcal{N}^{(2)}$ with different numbers of points $H = K = 10, 50, 100$. The notation $x(y)$ stays for $x \cdot 10^y$.

		f_1		f_2	
	$L = H = K$	E_2	E_{max}	E_2	E_{max}
$\mathcal{N}^{(1)}$	10	1.3(−1)	2.6(−2)	9.3(−1)	1.4(−1)
	50	1.1(−3)	4.2(−5)	5.9(−3)	2.4(−4)
	100	7.3(−5)	2.1(−6)	3.7(−4)	1.2(−5)
$\mathcal{N}^{(2)}$	10	1.8(0)	2.2(−1)	1.1(0)	1.0(−1)
	50	6.3(−3)	2.7(−4)	1.6(−2)	3.3(−4)
	100	3.9(−4)	1.4(−5)	6.9(−4)	1.3(−5)

$k = 0, 1, \dots, K$, we denote with $d_{i,h,k}$ the numerical approximation of the derivative with respect to x_i at the generic point $\mathbf{p} = (x_{1,h,k}, x_{2,h,k})^T = (\xi_1(u_h, v_k), \xi_2(u_h, v_k))^T \in \mathcal{N}$; given

$$e_{i,h,k} = d_{i,h,k} - \frac{\partial f}{\partial x_i}(\mathbf{p}),$$

the error for a generic function f is defined by the following two indices:

$$E_2 = \frac{1}{(H+1)(K+1)} \sum_{h=0}^{H} \sum_{k=0}^{K} \sqrt{\sum_{i=1,2} e_{i,h,k}^2}, \tag{27}$$

$$E_{max} = \max \left\{ \sqrt{\sum_{i=1,2} e_{i,h,k}^2},\ h = 0, 1, \dots, H,\ k = 0, 1, \dots, K \right\}. \tag{28}$$

From Table 2 we can see that the performance of the method is slightly different for the four considered cases; in particular, function f_2 seems to provide a more difficult case than f_1, likewise $\mathcal{N}^{(2)}$ seems more difficult than $\mathcal{N}^{(1)}$. A detailed error analysis has to be considered in future studies. However, the proposed method behaves satisfactorily showing a good convergence rate as the average distance of discretization points tends to zero.

In this test, we chose $L = H = K$ only for well reading the numerical results. It is clear that if f is smooth, larger is L more accurate is the solution, instead H and K are parameters related to the structure of the set $\mathcal{N}$ used in the test, it does not affect the results because we can always suppose that $\mathcal{N}$ is the union of (not necessarily disjoint) subsets having cardinality $(H+1)(K+1)$.

The good results, obtained in this numerical test, confirm the theoretical analysis of the accuracy of the proposed method and give a good starting point for applying it to the solution of PDEs, especially compared to other studies in numerical derivation of multivariate functions. The previously proposed methods, that had been used in [28], [29] and [30], usually have very complex methodological frameworks and provide accurate results comparable to ones of the proposed method.

In the numerical test, we have reported only the two-dimensional case to clearly show that the proposed method has a simple implementation. Similar results have been obtained in the analogous three-dimensional case, but the data were not reported for brevity, so we expect that for slightly higher dimensions the method also continues to work without major adjustments; however, the proposed method needs to be tested for much higher dimensions.

5. Conclusions

The paper proposes a new numerical derivation method for multivariate functions. This method is based on simple approximation techniques (like curve interpolation), numerical derivation of univariate functions and linear systems. A numerical experiment shows the performance of the proposed method and confirms its consistency and stability analysis given by the two stated theorems, in particular, the theoretical estimation of the convergence order had been numerically regained. The results that were obtained are promising and deserve further analysis in different directions. Problem 1 is a crucial step for the proposed method; its solvability should be studied from a theoretical point of view and practically by implementing algorithms to effectively compute $\mathbf{g}_j(t)$, $t \in [0, 1]$, $j = 1, 2, \dots, J$. More in detail, Problem 1 and its solution depend on the geometric properties of the set $\mathcal{N}$ and hence on the considered applied problem; for instance, when $\mathcal{N}$ is the set of vertices of a non-Cartesian grid in $\mathbb{R}^n$ we can use cubic-spline interpolating curves $\mathbf{g}_j$, as in the numerical test, or other standard uni-variate interpolation methods. Moreover, in this interpolation problem, the arrangement of the interpolation points is a key step, that we are trying to solve by graph routing problems, like the Chinese Postman problem and the Eulerian Path problem. Another important analysis is the error estimation with respect to the features of the function to be derived and of the sampling points as well as the property of the solution $\mathbf{g}_j(t)$, $t \in [0, 1]$, $j = 1, 2, \dots, J$, of Problem 1. This analysis strictly depends on the considered problem: $\mathcal{N}$, f and the set of curves $\mathbf{g}_j$, $j = 1, 2, \dots, J$, satisfying Problem 1, whose solution in not unique. The proposed method is potentially applicable to the numerical solution of partial differential equations, providing an alternative approach to finite difference methods, finite elements methods and finite volume methods; this is a further

interesting aspect that has to be investigated by future studies. As already discussed in the introduction, we expect this procedure could be generalised to solve fractional derivatives of multivariate functions.

The code of the proposed algorithm is available upon request to the corresponding author.

CRediT authorship contribution statement

Nadaniela Egidi: Writing – review & editing, Writing – original draft, Methodology, Conceptualization. **Josephin Giacomini:** Writing – review & editing, Validation, Methodology, Funding acquisition. **Pierluigi Maponi:** Writing – original draft, Software, Methodology, Formal analysis, Conceptualization.

Funding

This research received partial funding from the Unione Europea - FSE, Pon Ricerca e Innovazione 2014-2020 (Decreto Ministeriale 1062-10/08/2021).

Acknowledgements

The authors acknowledge support from INdAM-GNCS Project 2024 “Tecniche meshless ed equazioni integro-differenziali: analisi e loro applicazioni”. This research has been accomplished within the RITA “Research ITalian network on Approximation” and the UMI Group TAA “Approximation Theory and Applications”.

Appendix A. Reorganization of data

The data $\{G_0, G_1, \dots, G_L\}$ are reorganized in the following vectors

$$\mathbf{a}^{c,i} = \left(a_0^{c,i}, a_1^{c,i}, \dots, a_L^{c,i}\right)^T \in \mathbb{R}^M, \quad i = 0, 1, 2,$$

$$\mathbf{a}^{s,i} = \left(a_0^{s,i}, a_1^{s,i}, \dots, a_L^{s,i}\right)^T \in \mathbb{R}^M, \quad i = 1, 2,$$

whose components are:

$$a_l^{c,0} = \begin{cases} \frac{1}{12\delta}\left(-25G_0 + 16G_3 - 3G_4\right), & l = 0, \\ \frac{1}{24\delta}(-10G_1 + G_4), & l = 1, \\ 0, & 2 \le l \le L-2, \\ \frac{5}{12\delta}G_{L-1}, & l = L-1, \\ \frac{25}{24\delta}G_L, & l = L, \end{cases} \tag{A.1}$$

$$a_l^{c,1} = \begin{cases} -\frac{1}{4\delta}G_0, & l = 0, \\ \frac{10}{6\delta}G_1, & l = 1, \\ \frac{5}{12\delta}G_2, & l = 2, \\ 0, & 3 \le l \le L-3, \\ -\frac{5}{12\delta}G_{L-2}, & l = L-2, \\ -\frac{10}{6\delta}G_{L-1}, & l = L-1, \\ \frac{1}{8\delta}G_L, & l = L, \end{cases} \tag{A.2}$$

$$a_l^{c,2} = \begin{cases} \frac{1}{12\delta}G_0, & l = 0, \\ \frac{1}{24\delta}G_1, & l = 1, \\ -\frac{35}{24\delta}G_2, & l = 2, \\ -\frac{5}{24\delta}G_3, & l = 3, \\ 0, & 4 \le l \le L-4, \\ \frac{5}{24\delta}G_{L-3}, & l = L-3, \\ \frac{35}{24\delta}G_{L-2}, & l = L-2, \\ -\frac{1}{24\delta}G_l, & l = L-1, l = L, \end{cases} \tag{A.3}$$

$$a_l^{s,1} = \begin{cases} \frac{14}{6\delta} G_1, & l = 0, \\ \frac{13}{12\delta} G_2, & l = 1, \\ \frac{4}{6\delta} G_{l+1}, & 2 \le l \le L-4, \\ \frac{13}{12\delta} G_{L-2}, & l = L-3, \\ \frac{14}{6\delta} G_{L-1}, & l = L-2, \\ \frac{1}{8\delta} G_L, & l = L-1, \\ -\frac{4}{3\delta} G_{L-3} + \frac{1}{4\delta} G_{L-4}, & l = L, \end{cases} \tag{A.4}$$

$$a_l^{s,2} = \begin{cases} -\frac{1}{24\delta} G_1, & l = 0, \\ -\frac{37}{24\delta} G_2, & l = 1, \\ -\frac{7}{24\delta} G_3, & l = 2, \\ -\frac{1}{12\delta} G_{l+1}, & 3 \le l \le L-5, \\ -\frac{7}{24\delta} G_{L-3}, & l = L-4, \\ -\frac{37}{24\delta} G_{L-2}, & l = L-3, \\ -\frac{1}{24\delta} G_{l+1}, & l = L-2, L-1, \\ -\frac{1}{12\delta} G_{L-4}, & l = L, \end{cases} \tag{A.5}$$

where we recall that $\delta = 1/L$.

Appendix B. Convergence of the algorithm for univariate functions

We will prove Theorem 1, that is,

$$|G'(t_l) - d_l| = O(\delta^4), \qquad l = 0, 1, \dots, L,$$

where $\delta = 1/L$. Let be $M = L + 1$, $\xi_k = k/M$, $k \in \mathbb{Z}$, then, from the definition given in Section 2.2, we have

$$d_l = \beta_l \left(DCT^{(2)}(\mathbf{w}) \right)_{l+1}, \qquad l = 0, 1, \dots, L.$$

and for $j = 1, 2, \dots, M$,

$$\begin{aligned} & w_j = \left(FCT^{(3)}(\mathbf{a}^{c,0}) \right)_j + \tilde{c}_{j,1} \left(FCT^{(3)}(\mathbf{a}^{c,1}) \right)_j + \tilde{c}_{j,2} \left(FCT^{(3)}(\mathbf{a}^{c,2}) \right)_j + \\ & + \tilde{s}_{j,1} \left(FST^{(3)}(\mathbf{a}^{s,1}) \right)_j + \tilde{s}_{j,2} \left(FST^{(3)}(\mathbf{a}^{s,2}) \right)_j, \\ & \tilde{c}_{j,1}\tilde{c}_{j,k} = \frac{1}{2}\left(\tilde{c}_{j,k-1} + \tilde{c}_{j,k+1} \right), \quad \tilde{c}_{j,2}\tilde{c}_{j,k} = \frac{1}{2}\left(\tilde{c}_{j,k-2} + \tilde{c}_{j,k+2} \right), \\ & \tilde{s}_{j,1}\tilde{s}_{j,k} = \frac{1}{2}\left(\tilde{c}_{j,k-1} - \tilde{c}_{j,k+1} \right), \quad \tilde{s}_{j,2}\tilde{s}_{j,k} = \frac{1}{2}\left(\tilde{c}_{j,k-2} - \tilde{c}_{j,k+2} \right), \\ & (-1)^{j-1} = \tilde{s}_{j,L+1}, \quad \tilde{c}_{j,L+1+k} = -\tilde{c}_{j,L+1-k}. \end{aligned}$$

In particular for $j = 1, 2, \dots, L+1$, we have

$$\begin{aligned} w_j = & \frac{1}{\sqrt{M}} \Bigg[\left(a_0^{c,0} + a_0^{c,1}\tilde{c}_{j,1} + a_0^{c,2}\tilde{c}_{j,2} \right) + \\ & + 2 \sum_{m=1}^{L} \left(a_m^{c,0}\tilde{c}_{j,m} + \frac{1}{2}\left(a_m^{c,1}\tilde{c}_{j,m-1} + a_m^{c,1}\tilde{c}_{j,m+1} + a_m^{c,2}\tilde{c}_{j,m-2} + a_m^{c,2}\tilde{c}_{j,m+2} \right) \right) + \\ & + 2 \sum_{m=0}^{L-1} \frac{1}{2}\left(a_m^{s,1}\tilde{c}_{j,m} - a_m^{s,1}\tilde{c}_{j,m+2} + a_m^{s,2}\tilde{c}_{j,m-1} - a_m^{s,2}\tilde{c}_{j,m+3} \right) + \\ & + \frac{1}{2}\left(a_L^{s,1}\tilde{c}_{j,L} - a_L^{s,1}\tilde{c}_{j,L+2} + a_L^{s,2}\tilde{c}_{j,L-1} - a_L^{s,2}\tilde{c}_{j,L+3} \right) \Bigg] = \\ = & \frac{1}{\sqrt{M}} \Bigg[a_0^{c,0} + a_0^{c,1}\tilde{c}_{j,1} + a_0^{c,2}\tilde{c}_{j,2} + \\ & + \sum_{m=1}^{L} \left(2a_m^{c,0}\tilde{c}_{j,m} + a_m^{c,1}\tilde{c}_{j,m-1} + a_m^{c,1}\tilde{c}_{j,m+1} + a_m^{c,2}\tilde{c}_{j,m-2} + a_m^{c,2}\tilde{c}_{j,m+2} \right) + \\ & + \sum_{m=0}^{L-1} \left(a_m^{s,1}\tilde{c}_{j,m} - a_m^{s,1}\tilde{c}_{j,m+2} + a_m^{s,2}\tilde{c}_{j,m-1} - a_m^{s,2}\tilde{c}_{j,m+3} \right) + \end{aligned}$$

$$a_L^{s,1}\tilde{c}_{j,L} + a_L^{s,2}\tilde{c}_{j,L-1}\Bigg] = \left(DCT^{(3)}(\tilde{\mathbf{w}})\right)_j,$$

where

$$\tilde{w}_j = \begin{cases} a_0^{c,0} + a_1^{c,1} + a_2^{c,2} + a_0^{s,1} + a_1^{s,2}, & j = 1, \\ a_0^{c,1} + 2a_1^{c,0} + a_2^{c,1} + a_3^{c,2} + a_1^{c,2} + a_1^{s,1} + a_2^{s,2} + a_0^{s,2}, & j = 2, \\ a_0^{c,2} + 2a_2^{c,0} + a_3^{c,1} + a_1^{c,2} + a_4^{c,2} + a_2^{s,1} - a_0^{s,1} + a_3^{s,2}, & j = 3, \\ 2a_{j-1}^{c,0} + a_j^{c,1} + a_{j-2}^{c,1} + a_{j+1}^{c,2} + a_{j-3}^{c,2} + \\ \quad + a_{j-1}^{s,1} - a_{j-3}^{s,1} + a_j^{s,2} - a_{j-4}^{s,2}, & 4 \le j \le M-2, \\ 2a_{L-1}^{c,0} + a_L^{c,1} + a_{L-2}^{c,1} + a_{L-3}^{c,2} + \\ \quad + a_{L-1}^{s,1} - a_{L-3}^{s,1} - a_{L-4}^{s,2}, & j = M-1, \\ 2a_L^{c,0} + a_{L-1}^{c,1} + a_{L-2}^{c,2} - a_L^{c,2} + \\ \quad - a_{L-2}^{s,1} - a_{L-3}^{s,2} + a_{L-1}^{s,2}, & j = M. \end{cases}$$

We recall that from standard properties of discrete Fourier transforms

$$DCT^{(2)}(\mathbf{w}) = DCT^{(2)}\left(DCT^{(3)}(\tilde{\mathbf{w}})\right) = \tilde{\mathbf{w}},$$

so that from (10), for $l = 0, 1, \dots, L$,

$$d_l = \beta_l \left(FCT^{(2)}(\mathbf{w})\right)_{l+1} = \beta_l \tilde{\mathbf{w}}_{l+1}.$$

In particular by using the Taylor expansion of G_k, $k = 1, 2, 3, 4$, at t_0, we have

$$\begin{aligned} d_0 &= \beta_0 \tilde{w}_1 = a_0^{c,0} + a_1^{c,1} + a_2^{c,2} + a_0^{s,1} + a_1^{s,2} = \\ &= \frac{1}{24\delta}\left(-50G_0 + 32G_3 - 6G_4 + 40G_1 - 35G_2 + 56G_1 - 37G_2\right) = \\ &= \frac{1}{24\delta}\left(-50G_0 + 96G_1 - 72G_2 + 32G_3 - 6G_4\right) = \\ &= \tfrac{1}{24\delta}\Bigg(-50G_0 + 96\left(G_0 + G_0^{(1)}\delta + G_0^{(2)}\tfrac{\delta^2}{2} + G_0^{(3)}\tfrac{\delta^3}{6} + G_0^{(4)}\tfrac{\delta^4}{24} + g_{0,1}\delta^5\right) + \\ &\quad -72\left(G_0 + 2G_0^{(1)}\delta + 4G_0^{(2)}\tfrac{\delta^2}{2} + 8G_0^{(3)}\tfrac{\delta^3}{6} + 16G_0^{(4)}\tfrac{\delta^4}{24} + g_{0,2}\delta^5\right) + \\ &\quad +32\left(G_0 + 3G_0^{(1)}\delta + 9G_0^{(2)}\tfrac{\delta^2}{2} + 27G_0^{(3)}\tfrac{\delta^3}{6} + 81G_0^{(4)}\tfrac{\delta^4}{24} + g_{0,3}\delta^5\right) + \\ &\quad -6\left(G_0 + 4G_0^{(1)}\delta + 16G_0^{(2)}\tfrac{\delta^2}{2} + 64G_0^{(3)}\tfrac{\delta^3}{6} + 256G_0^{(4)}\tfrac{\delta^4}{24} + g_{0,4}\delta^5\right)\Bigg) = \\ &\quad = G_0^{(1)} + \tilde{g}_0\delta^4, \end{aligned}$$

where $G_0^{(k)}$ denotes the kth derivative of G at t_0, while $g_{0,k}$ and $\tilde{g}_0$ are values that depends of $G^{(5)}$ on the interval $[t_0, t_4]$.

With similar computations we obtain

$$d_l = G^{(1)}(t_l) + \tilde{g}_l h^4, \qquad l = 0, 1, \dots, L,$$

with $\tilde{g}_l$ a constant that depends on $G^{(5)}$.

References

[1] J. Cheng, X.Z. Jia, Y.B. Wang, Numerical differentiation and its applications, Inverse Probl. Sci. Eng. 15 (4) (2007) 339–357.
[2] Binbin Yin, Yuzhang Ye, Recovering the local volatility in Black-Scholes model by numerical differentiation, Appl. Anal. 85 (6–7) (2006) 681–692.
[3] Josephin Giacomini, Maria Chiara Invernizzi, Pierluigi Maponi, Massimo Verdoya, Testing a model of flow and heat transfer for U-shaped geothermal exchangers, Adv. Model. Anal. A 55 (2018) 151–157.
[4] Simone Angeloni, Josephin Giacomini, Pierluigi Maponi, Alessia Perticarini, Sauro Vittori, Luca Cognigni, Lauro Fioretti, Computer percolation models for espresso coffee: state of the art, results and future perspectives, Appl. Sci. 13 (2023) 2688.
[5] Josephin Giacomini, Pierluigi Maponi, Alessia Perticarini, CMMSE: a reduced percolation model for espresso coffee, J. Math. Chem. 61 (2023) 520–538.
[6] Jin Li, Zhilin Li, Kejia Pan, Accurate derivatives approximations and applications to some elliptic PDEs using HOC methods, Appl. Math. Comput. 459 (2023) 128265.
[7] Nadaniela Egidi, Josephin Giacomini, Pierluigi Maponi, Inverse heat conduction to model and optimise a geothermal field, J. Comput. Appl. Math. 423 (2023) 114957.
[8] Nadaniela Egidi, Josephin Giacomini, Pierluigi Maponi, Alessia Perticarini, Luca Cognigni, Lauro Fioretti, An advection-diffusion-reaction model for coffee percolation, Comput. Appl. Math. 41 (6) (2022) 229.
[9] Jianping Li, General explicit difference formulas for numerical differentiation, J. Comput. Appl. Math. 183 (2005) 29–52.
[10] A. Dutt, M. Gu, V. Rokhlin, Fast algorithms for polynomial interpolation, integration, and differentiation, SIAM J. Numer. Anal. 33 (1996) 0733082.
[11] J.A. Weideman, S.C. Reddy, A MATLAB differentiation matrix suite, ACM Trans. Math. Softw. 26 (2000) 465–519.

[12] Francesco Dell'Accio, Filomena Di Tommaso, Najoua Siar, Marco Vianello, Numerical differentiation on scattered data through multivariate polynomial interpolation, BIT Numer. Math. 62 (2018) 773–801.
[13] Abinash Nayak, A new regularization approach for numerical differentiation, Inverse Probl. Sci. Eng. 28 (12) (2020) 1747–1772.
[14] Baoqin Chen, Zhenyu Zhao, Zhi Li, Zehong Meng, Numerical differentiation by a Fourier extension method with super-order regularization, Appl. Math. Comput. 334 (2018) 1–10.
[15] Zhenyu Zhao, Lei You, A numerical differentiation method based on Legendre expansion with super order Tikhonov regularization, Appl. Math. Comput. 393 (2021) 125811.
[16] J.N. Lyness, C.B. Moler, Numerical differentiation of analytic functions, SIAM J. Numer. Anal. 4 (2) (1967) 202–210.
[17] William Squire, George Trapp, Using complex variables to estimate derivatives of real functions, SIAM Rev. 40 (1998) 110–112.
[18] Filip Nikolovski, Irena Stojkovska, Complex-step derivative approximation in noisy environment, J. Comput. Appl. Math. 327 (2018) 64–78.
[19] J.J. Moré, S.M. Wild, Estimating derivatives of noisy simulations, ACM Trans. Math. Softw. 38 (3) (2012) 1–21.
[20] Nadaniela Egidi, Josephin Giacomini, Pierluigi Maponi, A Fredholm integral operator for the differentiation problem, Comput. Appl. Math. 41 (5) (2022) 220.
[21] Rahul Kumar Maurya, Dongxia Li, Anant Pratap Singh, Vineet Kumar Singh, Numerical algorithm for a general fractional diffusion equation, Math. Comput. Simul. 223 (2024) 405–432.
[22] Vinita Devi, Rahul Kumar Maurya, Vineet Kumar Singh, A stable operational matrix based computational approach for multi-term fractional wave model arise in a dielectric medium, Chin. J. Phys. 87 (2024) 556–577.
[23] Rahul Kumar Maurya, Vineet Kumar Singh, A high-order adaptive numerical algorithm for fractional diffusion wave equation on non-uniform meshes, Numer. Algorithms 92 (2023) 1905–1950.
[24] Nadaniela Egidi, Pierluigi Maponi, The singular value expansion of the Volterra integral equation associated to a numerical differentiation problem, J. Math. Anal. Appl. 460 (2018) 656–681.
[25] Nadaniela Egidi, Josephin Giacomini, Pierluigi Maponi, Michael Youssef, An FFT method for the numerical differentiation, Appl. Math. Comput. 445 (2023) 127856.
[26] V. Britanak, K.R. Rao, P. Yip, Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations, Academic Press - Elsevier, Oxford, UK, 2007.
[27] Yu.G. Reshetnyak, On Taylor's formula for functions of several variables, Sib. Math. J. 54 (2013) 566–573.
[28] P.C.M. Lau, Curvilinear finite difference method for three-dimensional potential problems, J. Comput. Phys. 32 (3) (1979) 325–344.
[29] S.K. Kwok, An improved curvilinear finite difference (cfd) method for arbitrary mesh systems, Comput. Struct. 18 (4) (1984) 719–731.
[30] A.C. Albuquerque-Ferreira, Miguel Ureña, Higinio Ramos, The generalized finite difference method with third- and fourth-order approximations and treatment of ill-conditioned stars, Eng. Anal. Bound. Elem. 127 (2021) 29–39.