Доклади на Българската академия на науките
Comptes rendus de l'Académie bulgare des Sciences
Tome 79, No 8, 2026

MATHEMATICS

Computational mathematics

FAST ALGORITHMS FOR INTERPOLATION WITH CLAMPED *L*-SPLINES OF ORDER FOUR

Ognyan Kounchev[1,2]✉, Hermann Render[1,3], Georgi Simeonov[1], Tsvetomir Tsachev[1,2]

Received on May 20, 2026
Presented by S. Margenov, Corresponding Member of BAS, on June 30, 2026

Abstract

Interpolation and smoothing using cubic and generalized splines are fundamental tools in data analysis and statistical modelling. Recently, fast computational algorithms were developed for natural L-splines of order four, which arise as piecewise solutions to the differential operator $L_\xi^2 = (\frac{d^2}{dt^2} - \xi^2)^2$. In this paper, we extend this mathematical framework to the important case of clamped (or complete) boundary conditions, where the first derivatives at the interval endpoints are prescribed. We explicitly construct the governing linear system for the interpolation problem and mathematically prove that the resulting tridiagonal matrix is strictly row diagonally dominant, thereby guaranteeing its invertibility and the numerical stability of the fast algorithm. The proposed method is implemented in MATLAB. Furthermore, the developed clamped L-splines provide a foundation for constructing multivariate clamped polysplines, which serve as a promising alternative to Physics-informed neural networks (PINNs) for solving partial differential equations in mathematical physics.

The work of OK was partially supported by a project with Bulgarian NSF within the Multilateral competition for scientific and technological collaboration in the Danube region – 2024: Application of novel AI methods in analyzing Big data in Astrophysics and Physics: multidisciplinary and multilateral effort, entry number BG-175467353-2024-18-0018/FNI-238 of 17.01.2025. The work of GS and HR was partially supported by project with Bulgarian NSF, KP-06-RILA. The work of TT was partially supported by the Centre of Excellence in Informatics and ICT under Grant No BG16RFPR002-1.014-0018-C01, financed by the Research, Innovation and Digitalization for Smart Transformation Programme 2021–2027 and co-financed by the European Union.

https://doi.org/10.7546/CRABS.2026.08.02

Key words: L-splines, exponential splines, spline interpolation, clamped boundary conditions, fast algorithms, strictly diagonally dominant matrices

2020 Mathematics Subject Classification: 65D07, 41A15, 65D05, 65F05

1. Introduction. Interpolation with cubic polynomial splines is a popular method for the analysis and smoothing of big data – often used in statistical modelling, see [1–3] for mathematical aspects, and [4–8] for applications. Recently the authors proposed and studied in [9] fast algorithms for interpolation and smoothing for a special class of generalized splines arising as piece-wise solutions of differential operators, called L-splines. In [9] it is assumed that the differential operator L is of the form $L_\xi^2 := L_\xi \circ L_\xi$, where

$$L_\xi := \frac{d^2}{dt^2} - \xi^2 \quad \text{for} \quad \xi \in \mathbb{R}.$$

Let us recall the definition of an L_ξ^2-spline: assume that $\xi \geq 0$ is a real number and let $t_1 < t_2 < \cdots < t_{n-1} < t_n$ be real numbers. A function $g(\cdot) \in C^2[t_1, t_n]$ is called an L-spline if

$$L_\xi^2 g(t) = 0 \quad \text{for} \quad t \in (t_j, t_{j+1}).$$

A function $g(\cdot) \in C^2[t_1, t_n]$ is called an L_ξ^2-spline interpolating the data $\{g_1, g_2, \ldots, g_n\}$ at $\{t_1, t_2, \ldots, t_n\}$ if

$$g(t_j) = g_j \quad \text{for} \quad j = 1, 2, \ldots, n.$$

In order to obtain uniqueness for interpolation one has to add boundary conditions at the endpoints t_1 and t_n. In [9] we considered *natural L-splines* requiring the boundary condition

$$L_\xi g(t_1) = L_\xi g(t_n) = 0.$$

The main aim of the present paper is to show that one can derive fast algorithms as in [9] for the case of *clamped splines*, where the values $g'(t_1)$ and $g'(t_n)$ are prescribed. These splines are also called "complete splines", see [1] and/or in MATLAB. For related work on more general L-splines we refer to [10–12].

2. Fast interpolation for natural L_ξ^2-splines – a recap. Suppose that $g \in C^2[t_1, t_n]$ is an L_ξ^2-spline. In order to compute the L_ξ^2-spline g explicitly one has to solve for each interval $[t_j, t_{j+1}]$ with $j = 1, \ldots, n-1$, the boundary value problem $L_\xi^2 w = 0$ with

$$w(t_j) = g_j, \ w(t_{j+1}) = g_{j+1}, \text{ and } L_\xi w(t_j) = L_\xi g(t_j), \ L_\xi w(t_{j+1}) = L_\xi g(t_{j+1}).$$

Thus, in order to compute the natural L_ξ^2-spline g interpolating the data $z_j = g(t_j)$ for $j = 1, \ldots, n$, we need the knowledge of the two vectors

$$\mathbf{g}^T := (g(t_1), g(t_2), \ldots, g(t_n)), \tag{1}$$

$$\widetilde{\gamma}^T := (L_\xi g(t_1), \ldots, L_\xi g(t_n)). \tag{2}$$

In case of a natural interpolating L^2_ξ-spline we know $\mathbf{g}^T$ and $L_\xi g(t_1) = L_\xi g(t_n) = 0$. For this reason in [9] the vector

$$\gamma^T := (L_\xi g(t_2), L_\xi g(t_3), \ldots, L_\xi g(t_{n-1}))$$

is considered. When dealing with clamped L^2_ξ-splines we know $\mathbf{g}^T$ and the boundary conditions $g'(t_1) = d_1$ and $g'(t_n) = d_n$.

In [9] we proved the following:

Proposition 1. *Let $t_1 < \cdots < t_n$ and g be an L^2_ξ-spline. Then g restricted to the interval (t_j, t_{j+1}) for $j = 1, \ldots, n-1$ is equal to*

$$\psi_j(t) := L_\xi g(t_j) A_j^{[1]}(t) + L_\xi g(t_{j+1}) B_j^{[1]}(t) + g(t_j) A_j^{[2]}(t) + g(t_{j+1}) B_j^{[2]}(t), \tag{3}$$

where

$$A_j^{[2]}(t) = \frac{\sinh \xi(t - t_{j+1})}{\sinh \xi(t_j - t_{j+1})} \quad \text{and} \quad B_j^{[2]}(t) = \frac{\sinh \xi(t - t_j)}{\sinh \xi(t_{j+1} - t_j)}, \tag{4}$$

and

$$A_j^{[1]}(t) = \frac{\Phi_\xi(t - t_{j+1})}{\psi_\xi(t_j - t_{j+1})} - A_j^{[2]}(t) \frac{\Phi_\xi(t_j - t_{j+1})}{\psi_\xi(t_j - t_{j+1})}, \tag{5}$$

and

$$B_j^{[1]}(t) = \frac{\Phi_\xi(t - t_j)}{\psi_\xi(t_{j+1} - t_j)} - \frac{\Phi_\xi(t_{j+1} - t_j)}{\psi_\xi(t_{j+1} - t_j)} B_j^{[2]}(t). \tag{6}$$

Here we used the notations

$$\Phi_\xi(t) = \frac{1}{2\xi^3}(\xi t \cosh(\xi t) - \sinh(\xi t)),$$
$$\psi_\xi(t) = \frac{1}{\xi} \sinh \xi t = L_\xi \Phi_\xi(t).$$

Theorem 4.1 in [9] states:

Theorem 2. *Let $t_1 < t_2 < \cdots < t_n$ be real numbers. Assume that g is natural L^2_ξ-spline interpolating the data $g_1, \ldots, g_n$. Then there exists an $n \times (n-2)$ matrix Q and an $(n-2) \times (n-2)$ matrix R such that*

$$Q^T \mathbf{g} = R\gamma. \tag{7}$$

Here $\mathbf{g}$ and γ denote the vectors defined in (1) *and* (2).

The matrices R and Q can be explicitly described:

Theorem 3. *The matrix R is symmetric and tridiagonal and all entries on the diagonal and the first off-diagonal are positive. These entries are given by*

$$R_{j,j} = \rho(t_j - t_{j-1}) + \rho(t_{j+1} - t_j), \quad R_{j,j+1} = \sigma(t_{j+1} - t_j),$$

where $\rho(t)$ is defined by

$$\rho(t) = \frac{1}{4\xi} \frac{\sinh(2\xi t) - 2t\xi}{(\sinh \xi t)^2} \tag{8}$$

and

$$\sigma(t) = \frac{1}{2\xi} \frac{\xi t \cosh(\xi t) - \sinh(\xi t)}{\sinh^2(\xi t)}. \tag{9}$$

Theorem 4. *The elements of the $n \times (n-2)$ matrix Q are given, for $i = 1, \ldots, n$ and $j = 2, 3, \ldots, n-1$, by the formulas*

$$q_{j-1,j} = \frac{\xi}{\sinh \xi(t_j - t_{j-1})}, \quad q_{j+1,j} = \frac{\xi}{\sinh \xi(t_{j+1} - t_j)} \tag{10}$$

and

$$q_{jj} = -\xi \left(\frac{\cosh \xi(t_{j+1} - t_j)}{\sinh \xi(t_{j+1} - t_j)} + \frac{\cosh \xi(t_j - t_{j-1})}{\sinh \xi(t_j - t_{j-1})} \right).$$

The rest of the elements in Q are zero.

The arguments can be reversed to construct a natural L^2_ξ-spline interpolating given data $z = (z_1, \ldots, z_n)^T$ as we shall now outline:

Step 1. Set $g_i = z_i$ for $i = 1, \ldots, n$.

Step 2. Set $x = Q^T \mathbf{g}$ and solve $R\gamma = x$.

Then the natural cubic L^2_ξ-spline g interpolating the data z is then defined on each interval (t_j, t_{j+1}) by

$$\psi_j(t) := \gamma_j A_j^{[1]}(t) + \gamma_{j+1} B_j^{[1]}(t) + g_j A_j^{[2]}(t) + g_{j+1} B_j^{[2]}(t).$$

3. The main results. In this section we want to prove analogues of the results in Section 2 for clamped L^2_ξ-splines. As outlined in the last section, the L^2_ξ-spline $g(\cdot)$ coincides on the interval (t_j, t_{j+1}) with the function

$$\psi_j(t) := L_\xi g(t_j) A_j^{[1]}(t) + L_\xi g(t_{j+1}) B_j^{[1]}(t) + g(t_j) A_j^{[2]}(t) + g(t_{j+1}) B_j^{[2]}(t). \tag{11}$$

Since in (11) the numbers $g(t_j) = g_j$ for $j = 1, 2, \ldots, n$, are given, generating the spline $g(\cdot)$ reduces to finding the numbers $L_\xi g(t_j)$ for $j = 1, 2, \ldots, n$. For ease of notation we denote

$$\gamma_j := L_\xi g(t_j) \quad \text{for} \quad j = 1, 2, \ldots, n$$

and thus, formula (11) transforms to

$$\psi_j(t) = \gamma_j A_j^{[1]}(t) + \gamma_{j+1} B_j^{[1]}(t) + g_j A_j^{[2]}(t) + g_{j+1} B_j^{[2]}(t). \tag{12}$$

We next derive a system of n linear algebraic equations in n unknowns, in which $\{\gamma_j\}_{j=1}^n$ are the unknowns and $\{g_j := g(t_j)\}_{j=1}^n$ are given. From the continuity conditions at the knots $t_2 < t_3 < \cdots < t_{n-1}$ of g' we obtain the equalities

$$g'(t_j - 0) = g'(t_j + 0) \quad \big(\text{i.e. } \psi'_{j-1}(t_j) = \psi'_j(t_j)\big)$$

for $j = 2, 3, \ldots, n-1$. From the last and from (12) we obtain the following $n-2$ equations:

$$\begin{aligned}\gamma_{j-1}\frac{d}{dt}A_{j-1}^{[1]}(t_j) + \gamma_j \frac{d}{dt}B_{j-1}^{[1]}(t_j) + g_{j-1}\frac{d}{dt}A_{j-1}^{[2]}(t_j) + g_j\frac{d}{dt}B_{j-1}^{[2]}(t_j)\\ = \gamma_j\frac{d}{dt}A_j^{[1]}(t_j) + \gamma_{j+1}\frac{d}{dt}B_j^{[1]}(t_j) + g_j\frac{d}{dt}A_j^{[2]}(t_j) + g_{j+1}\frac{d}{dt}B_j^{[2]}(t_j)\end{aligned}$$

for $j = 2, 3, \ldots, n-1$. These equations may be rewritten in the form

$$\begin{aligned}&\gamma_{j-1}\frac{d}{dt}A_{j-1}^{[1]}(t_j) + \gamma_j\left(\frac{d}{dt}B_{j-1}^{[1]}(t_j) - \frac{d}{dt}A_j^{[1]}(t_j)\right) - \gamma_{j+1}\frac{d}{dt}B_j^{[1]}(t_j)\\ &\quad = -g_{j-1}\frac{d}{dt}A_{j-1}^{[2]}(t_j) + g_j\left(\frac{d}{dt}A_j^{[2]}(t_j) - \frac{d}{dt}B_{j-1}^{[2]}(t_j)\right) + g_{j+1}\frac{d}{dt}B_j^{[2]}(t_j)\end{aligned} \tag{13}$$

for $j = 2, \ldots, n-1$.

The *clamped boundary* conditions $\psi'_1(t_1) = g'(t_1) = d_1$ and $\psi'_{n-1}(t_n) = g'(t_n) = d_2$ give two additional equations

$$\gamma_1\frac{d}{dt}A_1^{[1]}(t_1) + \gamma_2\frac{d}{dt}B_1^{[1]}(t_1) = -g_1\frac{d}{dt}A_1^{[2]}(t_1) - g_2\frac{d}{dt}B_1^{[2]}(t_1) + d_1, \tag{14}$$

and

$$\gamma_{n-1}\frac{d}{dt}A_{n-1}^{[1]}(t_n) + \gamma_n\frac{d}{dt}B_{n-1}^{[1]}(t_n) = -g_{n-1}\frac{d}{dt}A_{n-1}^{[2]}(t_n) - g_n\frac{d}{dt}B_{n-1}^{[2]}(t_n) + d_2. \tag{15}$$

From (13), (14) and (15) we obtain a linear system of n equations in n unknowns $\gamma_1, \gamma_2, \ldots, \gamma_n$ which we want to write in matrix form of the type

$$\widetilde{R}\widetilde{\gamma}^T = \widetilde{Q}\mathbf{g}^T + \mathbf{d}^T \quad \text{for} \quad \mathbf{d} = (\underbrace{d_1, 0, \ldots, 0, d_2}_{n}),$$

where $\widetilde{\gamma} = \{\gamma_1, \gamma_2, \ldots, \gamma_n\}$ is the vector of the unknowns and $\mathbf{g} = \{g_1, g_2, \ldots, g_n\}$ are the given data. For convenience define as in [9]

$$\widetilde{R}_{j,j-1} = \frac{d}{dt}A_{j-1}^{[1]}(t_j), \quad \widetilde{R}_{j,j} = \frac{d}{dt}B_{j-1}^{[1]}(t_j) - \frac{d}{dt}A_j^{[1]}(t_j), \quad \widetilde{R}_{j,j+1} = -\frac{d}{dt}B_j^{[1]}(t_j)$$

for $j = 2, \dots, n-1$. The $n \times n$ matrix $\widetilde{R}$ is defined by

$$\widetilde{R} = \begin{pmatrix} \frac{d}{dt}A_1^{[1]}(t_1) & \frac{d}{dt}B_1^{[1]}(t_1) & 0 & \cdots & \cdots & 0 \\ \widetilde{R}_{2,1} & \widetilde{R}_{2,2} & \widetilde{R}_{2,3} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \widetilde{R}_{n-1,n-2} & \widetilde{R}_{n-1,n-1} & \widetilde{R}_{n-1,n} \\ 0 & \cdots & \cdots & 0 & \frac{d}{dt}A_{n-1}^{[1]}(t_n) & \frac{d}{dt}B_{n-1}^{[1]}(t_n) \end{pmatrix}.$$

Obviously, the structure of the matrix $\widetilde{R}$ is tridiagonal. Now we express the right hand side of the equations by an $n \times n$-matrix $\widetilde{Q}$ whose entries are defined by

$$\widetilde{Q}_{j-1,j} = -\frac{d}{dt}A_{j-1}^{[2]}(t_j) \quad \text{and} \quad \widetilde{Q}_{j+1,j} = \frac{d}{dt}B_j^{[2]}(t_j)$$

and

$$\widetilde{Q}_{jj} = \frac{d}{dt}A_j^{[2]}(t_j) - \frac{d}{dt}B_{j-1}^{[2]}(t_j).$$

Then it is easy to see that

$$\widetilde{Q} = \begin{pmatrix} -\frac{d}{dt}A_1^{[2]}(t_1) & -\frac{d}{dt}B_1^{[2]}(t_1) & 0 & 0 & \cdots & 0 \\ \widetilde{Q}_{2,1} & \widetilde{Q}_{2,2} & \widetilde{Q}_{2,3} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \ddots & \widetilde{Q}_{n-1,n-2} & \widetilde{Q}_{n-1,n-1} & \widetilde{Q}_{n-1,n} \\ 0 & 0 & \cdots & 0 & -\frac{d}{dt}A_{n-1}^{[2]}(t_n) & -\frac{d}{dt}B_{n-1}^{[2]}(t_n) \end{pmatrix}.$$

The following is the main result of this paper:

Theorem 5. *The matrix $\tilde{R}$ is strictly (row) diagonal dominant, in particular invertible.*

Proof. We show that $\widetilde{R}$ is strictly diagonal dominant; recall that a matrix $A = \{a_{i,j}\}_{i,j=1}^n$ is called strictly (row) diagonal dominant if $|a_{i,i}| > \sum_{i \neq j} |a_{i,j}|$ for all $i = 1, 2, \dots, n$, see, e.g., [4,13]. It is proved in [9] (see the proof of Theorem 4.2) that

$$\frac{d}{dt}B_1^{[1]}(t_1) = -\sigma(t_2 - t_1), \quad \text{and} \quad \frac{d}{dt}A_1^{[1]}(t_1) = -\rho(t_2 - t_1)$$

(just before Theorem 4.3). We also have $\sigma(x) > 0$ for $x > 0$ and $\rho(x) > 0$ for $x > 0$. Since in the proof of Theorem 4.4 of [9] it is established that $\sigma(x) < \frac{1}{2}\rho(x)$ for $x > 0$, we obtain

$$\left|\frac{d}{dt}B_1^{[1]}(t_1)\right| = \sigma(t_2 - t_1) \leq \frac{1}{2}\rho(t_2 - t_1) < \rho(t_2 - t_1) = \left|\frac{d}{dt}A_1^{[1]}(t_1)\right|.$$

Next we deal with the last row of the matrix R. Using formula (3.8) from [9] we obtain

$$\begin{aligned}\frac{d}{dt}A^{[1]}_{n-1}(t_n) &= \frac{\Phi'_\xi(t_n-t_n)}{\psi_\xi(t_{n-1}-t_n)} - \frac{d}{dt}A^{[2]}_{n-1}(t_n)\frac{\Phi_\xi(t_{n-1}-t_n)}{\psi_\xi(t_{n-1}-t_n)}\\ &= -\frac{\psi'_\xi(t_n-t_n)}{\psi_\xi(t_{n-1}-t_n)}\frac{\Phi_\xi(t_{n-1}-t_n)}{\psi_\xi(t_{n-1}-t_n)} = -\frac{\Phi_\xi(t_{n-1}-t_n)}{\psi^2_\xi(t_{n-1}-t_n)}\\ &= -\sigma(t_{n-1}-t_n) = \sigma(t_n-t_{n-1})\end{aligned}$$

(the second equality above follows from the proof of Theorem 4.2 of [9]). Also, we have the equality $\frac{d}{dt}B^{[1]}_{n-1}(t_n) = \rho(t_n - t_{n-1})$ (which follows from [9] – formula (3.3) and the proof of Theorem 4.2). Hence, again

$$\frac{d}{dt}A^{[1]}_{n-1}(t_n) = \sigma(t_n-t_{n-1}) \le \frac{1}{2}\rho(t_n-t_{n-1}) < \rho(t_n-t_{n-1}) = \frac{d}{dt}B^{[1]}_{n-1}(t_n).$$

So, we established the diagonal dominance for the first and for the last rows. The respective dominance for the rows of $\widetilde{R}$ with numbers $2, 3, \ldots, n-1$ is established in [9].

The strict (row) diagonal dominance implies that the matrix $\widetilde{R}$ is invertible, by [13, Theorem 6.1.10]. □

Thus we obtain

$$\widetilde{\gamma} = \left(\widetilde{R}\right)^{-1}\left(\widetilde{Q}\mathbf{g}^T + \mathbf{d}^T\right)$$

and we may proceed as at the end of Section 2 for computing the L-spline $g(t)$.

The present L-splines may be used to create multivariate clamped polysplines, cf. [14]. The last may be used as an alternative of the PINNs for solving equations of mathematical physics.

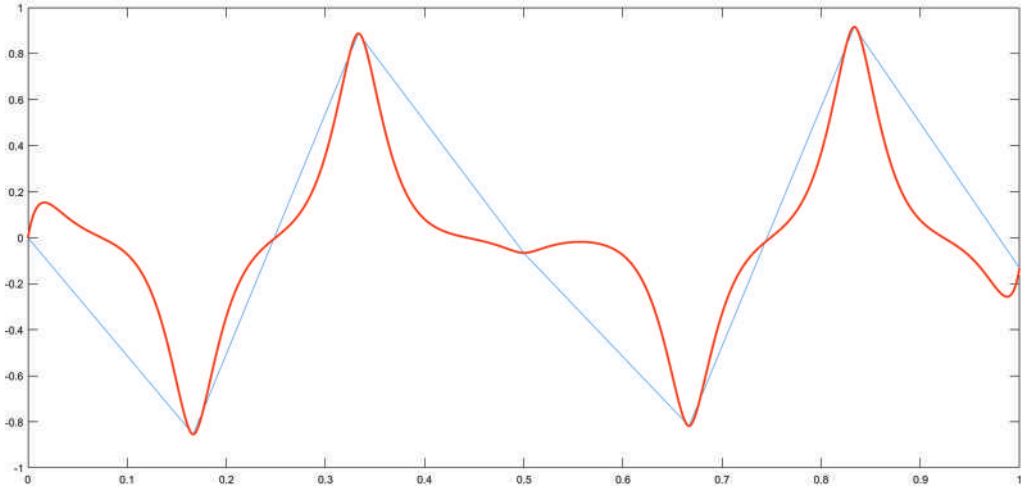


Fig. 1. Clamped L-spline

4. Figure. The fast algorithm above was implemented in MATLAB. As an example we provide the figure of a clamped L-spline for $L = L_\xi^2$ with $\xi = 5$, on the interval $[0, 1]$ with knots

$$t_1 = 0,\ t_2 = 0.1667,\ t_3 = 0.3333,\ t_4 = 0.5000,\ t_5 = 0.6667,\ t_6 = 0.8333,\ t_7 = 1$$

with derivatives $g'(t_1) = 25$ and $g'(t_7) = 25$ interpolating the function $f(t) = \sin(25t)$. In Fig. 1, the interpolating L_5^2-spline is compared with the linear spline interpolating $f(t)$.

REFERENCES

[1] de Boor C. (2001) A Practical Guide to Splines, Springer, Berlin.
[2] Stoer J., R. Bulirsch (2003) Introduction to Numerical Analysis, Springer, Berlin.
[3] Lyche T., L. L. Schumaker (1973) Computation of smoothing and interpolating natural splines via local bases, SIAM J. Numer. Anal., **10**(6), 1027–1038.
[4] Green P., B. Silverman (1994) Nonparametric Regression and Generalized Linear Models, Chapman and Hall.
[5] Gu Ch. (2013) Smoothing Spline ANOVA Models, Springer, New York.
[6] Hastie T., R. Tibshirani, J. Friedman (2013) The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, Berlin.
[7] Ramsay J., B. W. Silverman (2005) Functional Data Analysis, Springer, Berlin.
[8] Wahba G. (1990) Spline Models for Observational Data, SIAM, Philadelphia, Pennsylvania.
[9] Kounchev O., H. Render, T. Tsachev (2020) On a class of L-splines of order 4: fast algorithms for interpolation and smoothing, BIT Numer. Math., **60**, 879–899.
[10] McCartin B. J. (1990) Computation of exponential splines, SIAM J. Sci. Statist. Comput., **11**(2), 242–262.
[11] McCartin B. J. (1991) Theory of exponential splines, J. Approx. Theory **66**, 1–23.
[12] Kounchev O., H. Render, T. Tsachev (2023) Fast algorithms for interpolation with L-splines for differential operators L of order 4 with constant coefficients, J. Comp. Appl. Math., **422** 114898, 20 pp.
[13] Horn R. A., Ch. R. Johnson (1985) Matrix Analysis, Cambridge University Press, Cambridge, MA.
[14] Kounchev O. (2001) Multivariate Polysplines: Applications to Numerical and Wavelet Analysis, Academic Press, Inc., San Diego.

[1]*Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Akad. G. Bonchev St, Bl. 8, 1113 Sofia, Bulgaria*
e-mails: kounchev@math.bas.bg, render65@gmx.com, gsimeonov@math.bas.bg, tsachev@math.bas.bg

[2]*Centre of Excellence in Informatics and Information and Communication Technologies, Sofia, Bulgaria*

[3]*University College Dublin, Belfield 4, Dublin, Ireland*
e-mail: hermann.render@ucd.ie