Ann. Data. Sci. (2014) 1(1):5–14
DOI 10.1007/s40745-014-0002-z

Approximation of Irregular Geometric Data by Locally Calculated Univariate Cubic L^1 Spline Fits

Ziteng Wang · John Lavery · Shu-Cherng Fang

Received: 20 December 2012 / Revised: 19 March 2013 / Accepted: 20 April 2013 /
Published online: 25 April 2014

Abstract L^1 splines have been under development for interpolation and approximation of irregular geometric data. We investigate the advantages in terms of shape preservation and computational efficiency of calculating univariate cubic L^1 spline fits using a steepest-descent algorithm to minimize a global data-fitting functional under a constraint implemented by a local analysis-based interpolating-spline algorithm on 5-node windows. Comparison of these locally calculated L^1 spline fits with globally calculated L^1 spline fits previously reported in the literature indicates that the locally calculated spline fits preserve shape on the average slightly better than the globally calculated spline fits and are computationally more efficient because the locally-calculated-spline-fit algorithm can be parallelized.

Keywords Approximation · Cubic spline · L^1 spline · Shape preservation · Spline fit · Univariate

This work was generously supported by US Army Research Office Grant #W911NF-04-D-0003, by the North Carolina State University Edward P. Fitts Fellowship and by US NSF Grant # DMI-0553310. It is the policy of the Army Research Office that university personnel do not need to do joint work with ARO personnel in order to receive grants from the Army Research Office.

Z. Wang · J. Lavery · S.-C. Fang (✉)
Edward P. Fitts Department of Industrial and Systems Engineering, North Carolina State University, Raleigh, NC 27695-7906, USA
e-mail: fang@ncsu.edu

J. Lavery
Mathematical Sciences Division and Computing Sciences Division, Army Research Office, Army Research Laboratory, P.O. Box 12211, Research Triangle Park, Durham, NC 27709-2211, USA

1 Introduction

Over the past decade and more, L^1 splines have been under development for interpolation and approximation of irregular geometric data, that is, data with abrupt changes in magnitude, direction and/or spacing. L^1 splines, which are based on nonlinear programming that implements a geometric principle of shape preservation, have the advantage of being free of the extraneous oscillation common in other interpolants and approximants such as unconstrained and constrained polynomial and rational splines, including those with penalties, a posteriori filtering and fairing. Recent advances in univariate and bivariate L^1 interpolating splines (see [1,2,5,7,8,12,14,16] and [10,13], resp.) have been accompanied by some but fewer advances in univariate and bivariate L^1 approximating splines (see [9,11] and [3], resp.). This is unfortunate, since approximation is, from a practical point of view, more important than interpolation.

In 2007, Auquiert, Gibaru and Nyiri discovered that L^1 interpolating splines calculated on small sets of 5 nodes preserve linearity of 3-node subsets of the 5 nodes [2]. This discovery led to the development of univariate L^1 interpolating splines calculated locally on 5-point windows [7,8,14,16]. These locally calculated L^1 interpolating splines have the dual advantage of enhanced shape preservation and sharply reduced computing time, the latter of which is due an algorithm in which parallel analysis replaces sequential raw computing. The local-calculation approach that has been successful for interpolating splines has, however, not previously been investigated for approximating splines. This paper is a first step toward developing locally calculated L^1 approximating splines to replace the previously globally calculated L^1 approximating splines.

Approximation by L^1 (and other) splines can be carried out using spline fits or smoothing splines. Spline fits are calculated by minimizing a data fitting functional over a manifold of interpolating splines. Smoothing splines are calculated by minimizing a linear combination of a data fitting functional and an interpolating spline functional over a spline space. Smoothing splines depend on a "balance parameter" that represents the relative weight of fitting the data vs. that of smoothing out local variations in the data. Many researchers have investigated the balance parameter for classical smoothing splines (see [15] and references therein) but practically relevant, optimal balance parameters for large classes of data are not yet available. For L^1 smoothing splines, no theoretical information is available at all [9,11]. For this reason, spline fits, which do not require the user to provide a balance parameter, are of high interest and are the approximating splines that we choose here.

The precise goal of the present paper is to investigate the advantages in terms of shape preservation and computational efficiency of calculating univariate cubic L^1 spline fits using a steepest-descent algorithm to minimize a global data-fitting functional under a constraint implemented by the local analysis-based interpolating-spline algorithm of [16] on 5-node windows. We will compare these "locally calculated L^1 spline fits" with the globally calculated L^1 spline fits presented in [11].

2 Cubic L^1 Spline Fits: Definitions and Algorithms

Let the data to be approximated be $(\hat{x}_m, \hat{z}_m), m = 1, 2, \ldots, M$, where all of the $\hat{x}_m$ are in a finite real interval $[x_0, x_I]$. Let there be given strictly monotonically increasing but otherwise arbitrary spline nodes $x_i, i = 0, 1, ..., I$. Let the function values and the first derivatives of the spline fit at these nodes be denoted by z_i and b_i, respectively. Let $\mathbf{z}$ denote $(z_0, z_1, \ldots, z_I)$ and $\mathbf{b}$ denote $(b_0, b_1, \ldots, b_I)$. Let $h_i = x_{i+1} - x_i$ and $t = \frac{x-x_i}{h_i} - \frac{1}{2}$ for $-0.5 < t < 0.5$. We consider C^1-smooth piecewise cubic polynomials $z(x)$ with nodes $x_i, i = 0, 1, \ldots, I$, of the (Hermite) form

$$\begin{aligned} z(x) = &\left(\tfrac{1}{2} - \tfrac{3}{2}t + 2t^3\right) z_i + \left(\tfrac{1}{2} + \tfrac{3}{2}t - 2t^3\right) z_{i+1} \\ &+ \left(\tfrac{1}{8} - \tfrac{1}{4}t - \tfrac{1}{2}t^2 + t^3\right) h_i b_i + \left(-\tfrac{1}{8} - \tfrac{1}{4}t + \tfrac{1}{2}t^2 + t^3\right) h_i b_{i+1} \end{aligned} \tag{1}$$

in each interval $[x_i, x_{i+1}], i = 0, 1, \ldots, I-1$. A cubic L^1 spline fit is a function $z(x)$ of form (1) that minimizes the data fitting functional

$$F(\mathbf{z}, \mathbf{b}) := \sum_{m=1}^{M} \left| z(\hat{x}_m) - \hat{z}_m \right| \tag{2}$$

over a manifold of cubic L^1 interpolating splines. The free parameters in a cubic L^1 spline fit are the function values $z_i = z(x_i), i = 0, 1, \ldots, I$, at the nodes. The first derivatives $b_i = \mathrm{d}z/\mathrm{d}x(x_i), i = 0, 1, \ldots, I$ of the spline fit are dependent variables that are calculated locally or globally from the z_i by minimizing an L^1 interpolating spline functional as described in the next two paragraphs.

A globally calculated cubic L^1 interpolating spline is a function of form (1) that minimizes the global functional

$$\int_{x_0}^{x_I} \left| \frac{\mathrm{d}^2 z}{\mathrm{d}x^2} \right| \mathrm{d}x. \tag{3}$$

[11] over all functions of form (1). In this minimization process, the z_i are fixed and the free variables are the first derivatives $b_i = \mathrm{d}z/\mathrm{d}x(x_i), i = 0, 1, \ldots, I$ of the spline. Nonuniqueness is handled by choosing the feasible b_i that are closest to 0, a "regularization procedure" that was used for the computational examples in [11] with which we will compare the new results in this present paper. [Other regularizations could be used. In [16], nonuniqueness is handled by choosing the feasible b_i that is closest to $(z_{i+1} - z_{i-1})/(x_{i+1} - x_{i-1})$].

A locally calculated cubic L^1 interpolating spline [16] is a function of form (1) in which each of the first derivatives $b_i, i = 3, 4, \ldots, I-3$ for (1) is calculated by minimizing the local 5-node-window spline functional

$$\int_{x_{i-2}}^{x_{i+2}} \left| \frac{\mathrm{d}^2 z}{\mathrm{d}x^2} \right| \mathrm{d}x. \tag{4}$$

over the free variables $b_k, k = i-2, i-1, i, i+1, i+2$, with $z_k, k = i-2, i-1, i, i+1, i+2$, fixed. The first derivatives b_0, b_1 and b_2 are calculated by minimizing functional (4) with $i = 2$ and the first derivatives b_{I-2}, b_{I-1} and b_I are calculated by minimizing functional (4) with $i = I - 2$. Nonuniqueness is handled by choosing the feasible b_i that is closest to 0.

The coefficients of a globally calculated L^1 spline fit are the solution of a two-level or "bilevel" optimization problem, namely, that of minimizing the ℓ^1 data-fitting functional (2) for the z_i subject to a constraint that the b_i are functions of the z_i that are determined by minimizing the global L^1 interpolating spline functional (3). Many approaches for solving bilevel optimization problems exist, including branch-and-bound, complementary-pivoting, descent, penalty-function and trust-region approaches [6]. However, there are still many challenges, especially for nonlinear bilevel problems, a category that includes calculation of the coefficients of globally calculated L^1 spline fits. In [11], a "Lagrange-multiplier primal-affine algorithm" was used to solve this bilevel problem. On each iterative step, this algorithm applies a primal-affine approach to reduce the minimization of the data-fitting functional (2) and the minimization of the global L^1 interpolating spline functional (3) to weighted least-squares problems. One formulates the linear systems that correspond to these two problems, links these two linear systems by Lagrange multipliers and computes the next iterate by solving the resulting larger linear system. A theoretical proof of of convergence is not available but computational results indicate good performance. At the same time, the method is relatively computationally expensive due to the global structure of the calculations.

The recent development of locally calculated L^1 interpolating splines, which are better at shape preservation and much more computationally efficient than globally calculated L^1 interpolating splines, opens up opportunities for approximation by locally calculated L^1 spline fits that we wish to begin to explore in this paper. The opportunities are threefold: (1) increased computational efficiency by shifting from minimization of a global L^1 interpolating spline functional to parallel minimization of many local L^1 interpolating spline functionals, (2) increased shape-preservation capability and (3) potential availability of theoretical proof of convergence due to knowledge of the analytical structure of the b_i when they are calculated locally from the z_i. There exists no known analogue of this analytical structure for globally calculated L^1 splines. The present paper focuses on the first two of these opportunities.

The algorithm that we propose for locally calculated L^1 spline fits is a steepest-descent algorithm for minimizing $F(\mathbf{z}, \mathbf{b}(\mathbf{z}))$ with $\mathbf{z}$ as a free, unconstrained vector of variables. The initial iterate from which the steepest-descent process starts is the linear spline fit of the data. The linear spline fit is the function $\zeta(x)$ that minimizes the data fitting functional

$$\sum_{m=1}^{M} \left|\zeta(\hat{x}_m) - \hat{z}_m\right| \tag{5}$$

over the manifold of C^0-smooth piecewise linear polynomials $\zeta(x)$ with nodes x_i, $i = 0, 1, \ldots, I$, of the form

$$\zeta(x) = \tfrac{(x_{i+1}-x)}{h_i} z_i + \tfrac{(x-x_i)}{h_i} z_{i+1} \tag{6}$$

in each interval $[x_i, x_{i+1}]$. The linear spline fit is geometrically close to the cubic L^1 spline fit (an observation for which theoretical proof is not yet available), which is a factor in promoting good performance of the steepest-descent algorithm that we introduce here.

To proceed from an approximant $\mathbf{z}^{(k)}$ on the kth step to the next approximant $\mathbf{z}^{(k+1)}$, we first compute a numerical gradient $\bigtriangledown^{(k)}$ of $F(\mathbf{z}, \mathbf{b}(\mathbf{z}))$ with respect to $\mathbf{z}$ at $\mathbf{z} = \mathbf{z}^{(k)}$. The derivative $\mathrm{d}\mathbf{b}/\mathrm{d}\mathbf{z}$ that occurs in this gradient is calculated by analytical differentiation of the $\mathbf{b}(\mathbf{z})$ of [7,16] but with nonuniqueness resolved by choosing the feasible b_i that is closest to 0 instead of by choosing the b_i closest to $(z_{i+1} - z_{i-1})/(x_{i+1} - x_{i-1})$. This formula is not fully applicable at boundaries where the analytical derivative of $\mathbf{b}$ with respect to $\mathbf{z}$ is discontinuous. This aspect of the algorithm will be adjusted in the future. However, since the initial iterate, the linear spline fit, is close to the final solution, that is, to the cubic L^1 spline fit, there are not a large number of boundaries with discontinuous derivatives between the initial iterate and the final solution and the current version of the algorithm performs well. After calculating $\bigtriangledown^{(k)}$, we use golden section line search to numerically calculate a step length $\gamma^{(k)}$ that minimizes $F(\mathbf{z}^{(k)} - \gamma\bigtriangledown^{(k)}, \mathbf{b}(\mathbf{z}^{(k)} - \gamma\bigtriangledown^{(k)}))$ over $0 \le \gamma \le 1$. Finally, we set $\mathbf{z}^{(k+1)} = \mathbf{z}^{(k)} - \gamma^{(k)}\bigtriangledown^{(k)}$. If $F(\mathbf{z}^{(k+1)}, \mathbf{b}(\mathbf{z}^{(k+1)})) > F(\mathbf{z}^{(k)}, \mathbf{b}(\mathbf{z}^{(k)}))$, we stop and output $\mathbf{z}^{(k)}$. Otherwise, we increase k by 1 and repeat the process. A maximum number of iterations (100 in our computational experiments) is set for potential cases of slow convergence or non-convergence.

3 Computational Results

Computations were carried out on a LENOVO Thinkpad T410 laptop with an Intel Core i5 2.67GHz CPU and 4.00GB RAM. The software environment was Microsoft Windows 7 Professional and MATLAB R2012a. Locally calculated L^1 spline fits were computed using a MATLAB code and globally calculated L^1 spline fits were computed using a legacy C++ code, both codes implemented sequentially. On the T410 laptop, MATLAB was 112.3 times slower than a C++ code for a task consisting of 10^7 additions and 10^7 multiplications. In reporting CPU times below, we have normalized the CPU times required by the MATLAB code to corresponding C++ code CPU times by division by 112.3.

We present in Table 1 computational results for locally and globally calculated cubic L^1 spline fits for "data-and-node sets" 1, 2 and 4, resp., of [11]. These cubic L^1 spline fits are shown in Figs. 1, 2, 3, 4, 5, 6. In these figures, the data points are represented by the symbol $+$ and the spline nodes are represented by the symbol $\circ$.

Comparison of the locally calculated L^1 spline fits of Figs. 1, 3 and 5 with the corresponding globally calculated L^1 spline fits of [11] in Figs. 2, 4 and 6 indicates that the locally calculated spline fits preserve shape in these computational experiments on the average slightly better than the globally calculated spline fits, a situation that is reflected in the ℓ^1 errors for data-and-node sets 1 and 4. This observation is analogous to previous results in the literature that indicate that locally calculated L^1 interpolating splines preserve shape slightly better than globally calculated L^1 interpolating splines.

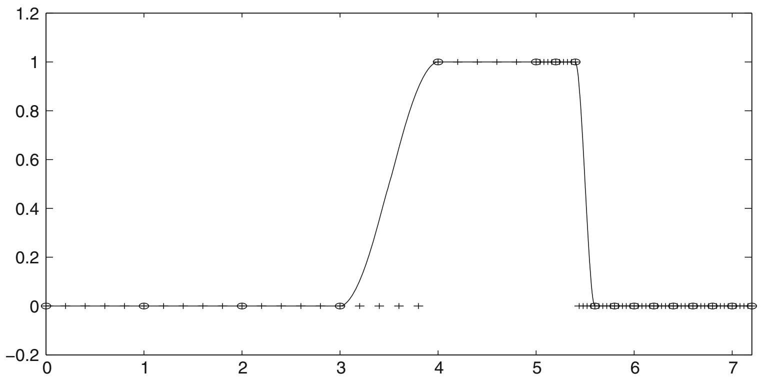


Fig. 1 Locally calculated cubic L^1 spline fit for data-and-node set 1

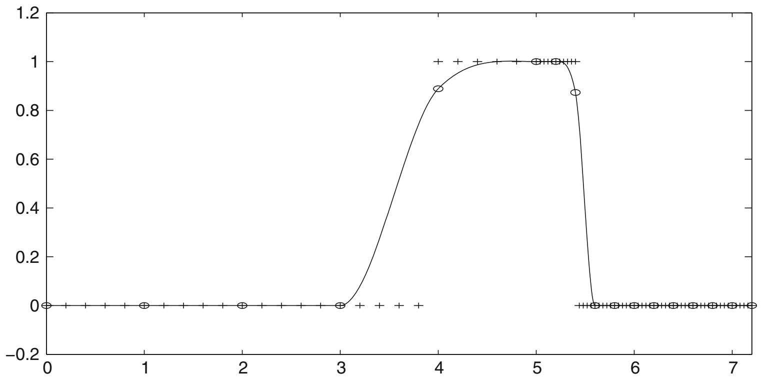


Fig. 2 Globally calculated cubic L^1 spline fit for data-and-node set 1

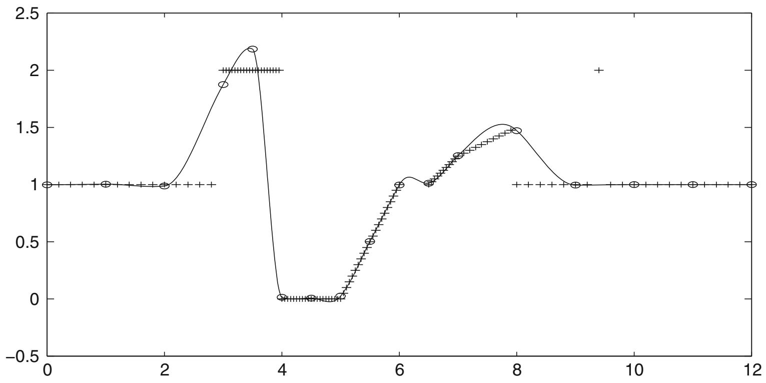


Fig. 3 Locally calculated cubic L^1 spline fit for data-and-node set 2

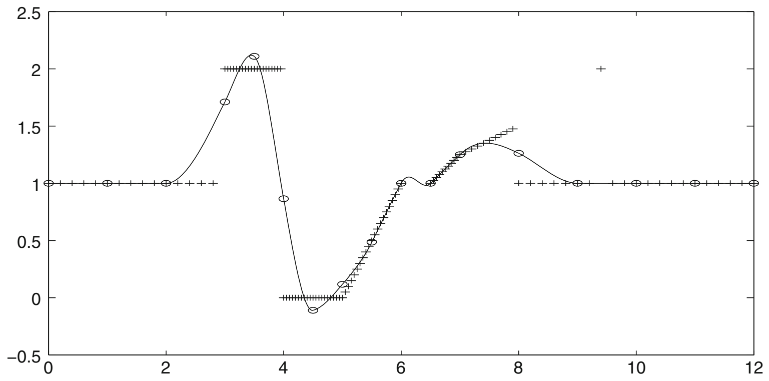


Fig. 4 Globally calculated cubic L^1 spline fit for data-and-node set 2

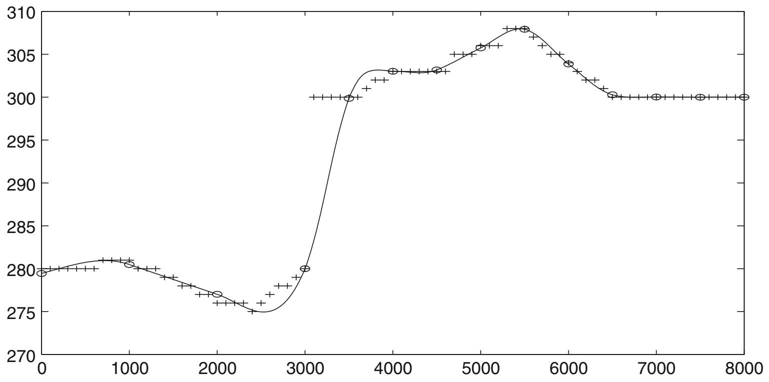


Fig. 5 Locally calculated cubic L^1 spline fit for data-and-node set 4

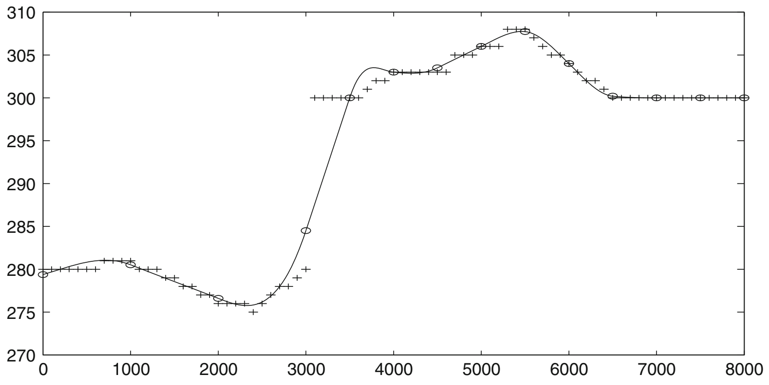


Fig. 6 Globally calculated cubic L^1 spline fit for data-and-node set 4

Table 1 Computational results

	Locally calculated L^1 spline fit			Globally calculated L^1 spline fit	
Data-and-node set	# of iterations	CPU time	ℓ^1 error	CPU time	ℓ^1 error
1	1	3.5 ms	0.0494	3.8 ms	0.0501
2	21	10.4 ms	0.123	1.8 ms	0.102
4	5	1.9 ms	0.902	1.9 ms	1.36

The locally calculated L^1 spline fit for data-and-node set 2 preserves shape better than the globally calculated L^1 spline fit in the interval [4, 5] but not as well in the intervals [1, 3.5] and [7, 9.5]. The sequential computing times for locally and globally calculated L^1 spline fits reported in Table 1 are of the same order of magnitude. (No further conclusion about relative computing time is appropriate at present due to the small amount of data and the fact that the MATLAB computing time for the locally calculated L^1 spline fits was artificially normalized to a corresponding C++ computing time.) The sequential structure of the current locally-calculated-L^1-spline-fit code will be parallelized in the future, which will lead to a large increase in computational efficiency. The globally-calculated-L^1-spline-fit code cannot be parallelized.

4 Conclusion

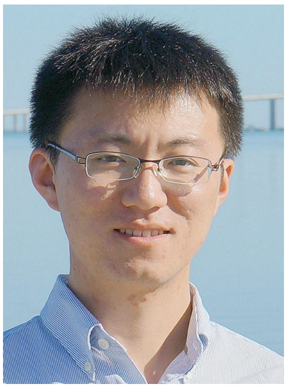
The evidence that we have presented here is preliminary but is already sufficient to suggest that further investigation of the local-calculation approach for L^1 spline fits may well lead to significant advances in shape-preserving approximation. While it is commonly expected that one must trade off shape-preservation capability vs. computational efficiency, the results presented here indicate that the local-calculation approach with its parallel structure may allow both shape-preservation capability and computational efficiency to be enhanced simultaneously. One option that has not yet been investigated but that will be investigated in the future is replacement of the global data-fitting functional (2) by local ℓ^1 data-fitting functionals, perhaps on the same 5-node windows that are used for the local L^1 interpolating spline functional (4).

Theoretical proof of convergence of the algorithm is hampered by the complexity of the dependence of $\mathbf{b}$ on $\mathbf{z}$. The authors expect that the interplay of the regularization strategy for the L^1 interpolating spline functional (that is, how a specific solution is chosen when the minimum occurs over a whole interval) with the resulting smoothness of $\mathbf{b}(\mathbf{z})$ will play a large role in this proof when the authors develop it in the future.

The results about univariate L^1 spline fits that we have presented here are not merely a step toward improved univariate approximation but also a basis for bivariate and multivariate approximation. In contrast to the situation with univariate conventional splines, univariate L^1 splines generalize in natural ways to higher dimensions, a statement that is valid not merely for interpolating splines but also for spline fits. Further univariate results will pave the way for future investigation of the shape-preservation capability and computational efficiency of bivariate and multivariate L^1 spline fits.

References

1. Auquiert P, Gibaru O, Nyiri E (2007a) C^1 and C^2-continuous polynomial parametric L_p splines ($p \geq 1$). Comput Aided Geom Design 24:373–394
2. Auquiert P, Gibaru O, Nyiri E (2007b) On the cubic L_1 spline interpolant to the Heaviside function. Numer Algorithms 46:321–332
3. Bulatov D, Lavery JE (2010) Reconstruction and texturing of 3D urban terrain from uncalibrated monocular images using L_1 splines. Photogramm Eng Remote Sens 76:439–449
4. Cheng H, Fang S-C, Lavery JE (2004) An efficient algorithm for generating univariate cubic L_1 splines. Comput Optim Appl 29:219–253
5. Cheng H, Fang S-C, Lavery JE (2005) Shape-preserving properties of univariate cubic L_1 splines. J Comput Appl Math 174:361–382
6. Colson B, Marcotte P, Savard G (2005) Bilevel programming: a survey. 4OR 3, 87–107
7. Jin Q, Lavery JE, Fang S-C (2010) Univariate cubic L_1 interpolating splines: analytical results for linearity, convexity and oscillation on 5-point windows. Algorithms 3:276–293
8. Jin Q, Yu L, Lavery JE, Fang S-C (2012) Univariate cubic L_1 interpolating splines based on the first derivative and on 5-point windows: analysis, algorithm and shape-preserving properties. Comput Optim Appl 51:575–600
9. Lavery JE (2000) Shape preserving, multiscale fitting of univariate data by cubic L_1 smoothing splines. Comput Aided Geom Design 17:715–727
10. Lavery JE (2001) Shape-preserving, multiscale interpolation by bi- and multivariate cubic L_1 splines. Computer Aided Geom Design 18:321–343
11. Lavery JE (2004) Shape preserving approximation of multiscale univariate data by cubic L_1 spline fits. Comput Aided Geom Design 21:43–64
12. Lavery JE (2009) Shape-preserving univariate cubic and higher-degree L_1 splines with function-value-based and multistep minimization principles. Comput Aided Geom Des 26:1–16
13. Lin Y-M, Zhang W, Wang Y, Fang S-C, Lavery J.E (2006) Computationally efficient models of urban and natural terrain by non-iterative domain decomposition with L_1 smoothing splines. In: Proceedings of the 25th army science conference, Department of the Army, Washington, DC, Nov 2006
14. Nyiri E, Gibaru O, Auquiert P (2011) Fast L_1^k-C^k polynomial spline interpolation algorithm with shape-preserving properties. Comput Aided Geom Des 28:65–74
15. Wahba G (1990) Spline models for observational data. SIAM, Philadelphia
16. Yu L, Jin Q, Lavery JE, Fang S-C (2010) Univariate cubic L_1 interpolating splines: spline functional, window size and analysis-based algorithm. Algorithms 3:311–328



Ziteng Wang received his BS degree in Mathematics from Tsinghua University, Beijing, China in 2010. He finished his MS degree in Industrial Engineering and is now working at his PhD degree in Industrial and Systems Engineering at the North Carolina State University, Raleigh, NC, USA. His research interests include supply chain management, geometric modeling and mathematical optimization.

John Lavery received his Ph. D. degree in Mathematics from the University of Maryland, College Park, MD, USA. He has been working at the US Army Research Office since 1995. His research interests include R&D in mathematical modeling for geometric design and medical imaging, and L_1 splines for modeling irregular objects (terrain, biological objects, etc.).

Shu-Cherng Fang received his Ph. D. degree in Industrial Engineering and Management Science from Northwestern University, Evanston, IL, USA. He holds the Walter Clark Chair Professorship and Alumni Distinguished Graduate Professorship at the North Carolina State University, USA. His research interests include linear and nonlinear programming, fuzzy optimization and decision making, soft computing, logistics and supply chain management.