Computer-Aided Design 58 (2015) 179–188

Contents lists available at ScienceDirect

Computer-Aided Design

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

Knot calculation for spline fitting via sparse optimization☆

Hongmei Kang, Falai Chen*, Yusheng Li, Jiansong Deng, Zhouwang Yang

School of Mathematical Sciences, University of Science and Technology of China, Hefei 230026, PR China

HIGHLIGHTS

- We reduce the computation time dramatically by solving convex optimization problem.
- We can simultaneously find a good combination of the knot number and knot locations.
- The algorithm has less knots with good fitting performance compared to other methods.
- We can recover the ground truth knots when data is sampled enough from a B-spline.

ARTICLE INFO

Keywords:
Spline fitting
Knot calculation
Sparse optimization

ABSTRACT

Curve fitting with splines is a fundamental problem in computer-aided design and engineering. However, how to choose the number of knots and how to place the knots in spline fitting remain a difficult issue. This paper presents a framework for computing knots (including the number and positions) in curve fitting based on a sparse optimization model. The framework consists of two steps: first, from a dense initial knot vector, a set of active knots is selected at which certain order derivative of the spline is discontinuous by solving a sparse optimization problem; second, we further remove redundant knots and adjust the positions of active knots to obtain the final knot vector. Our experiments show that the approximation spline curve obtained by our approach has less number of knots compared to existing methods. Particularly, when the data points are sampled dense enough from a spline, our algorithm can recover the ground truth knot vector and reproduce the spline.

1. Introduction

Curve fitting with splines is a traditional and fundamental problem in many engineering practices. In Computer Aided Design (CAD) and Geometric Modeling, curves are fitted with splines to reconstruct geometric models from measurement data [1–4]. In signal processing and image processing, splines are often adopted to process noisy signals or to approximate complicated functions [5,6].

The intuitive idea of curve fitting with splines is to formulate it as a least-square problem when knots are fixed. However, the fitting result is not always satisfactory. Actually, it has long been known that freeing knots in fitting improves the result dramatically [7–10]. But spline fitting with free knots is still a challenging problem. The reasons are as follows. First, analytic expressions for optimal knot locations, or even for general characteristics of optimal knot distributions, are not easy to derive [11]. Second, the unknown number and position of knots result in a large and nonlinear optimization problem, which is computationally very difficult.

In the literature many methods have been proposed to optimize knots with a given number of knots. The problem of knot placement is formulated as a nonlinear optimization problem with the constraint that knots should form a nondecreasing sequence. The first type of techniques transforms the constrained optimization problem into an unconstrained problem, then local gradient-based method or Gauss–Newton method are employed for minimization [11–13]. However, local optimization methods require a good initial guess and cannot guarantee global optimality. The second type of techniques applies global optimization to avoid the drawbacks of local methods, but it is computationally more expensive [14–16]. There are also some works which utilize the underlying feature information of the data to select knots, instead of solving a nonlinear optimization problem [1,17,18]. However, in such methods the number of knots is determined beforehand and the results are sensitive to measurement noises.

Another approach for knot calculation is based on knot-removal strategy which is to reduce the number of knots of a given spline

☆ This paper has been recommended for acceptance by Vadim Shapiro.

* Corresponding author.
E-mail addresses: khmkangkang@gmail.com, chenfl@ustc.edu.cn (F. Chen).

http://dx.doi.org/10.1016/j.cad.2014.08.022
0010-4485/

by perturbing the spline within a given tolerance [19,20]. The main idea of the technique is to remove interior knots according to assigned weights. For data approximation, a piecewise linear approximation of the data is computed, then knot removal strategy is performed on the linear approximation, and finally the data is approximated by a smooth spline with computed knots.

A new development in recent years for knot calculation is based on sparse optimization [21,22]. Sparsity is the core of compressed sensing which is widely used in computer vision and signal processing [23–27]. Sparsity means that a signal can be represented in a linear combination of some bases or dictionaries such that most of the combination coefficients are zero. In [21], the authors formulated the spline fitting problem as a convex optimization problem, where the l_1 norm of jump of third order derivatives of C^2 cubic splines is minimized; while in [22], the authors first selected a subset of basis functions from the pre-specified multi-resolution basis set using the statistical variable selection method-Lasso, then identified a concise knot vector that is sufficient to characterize the vector space spanned by the selected basis functions to fit the data. These two methods can compute the number and positions of the knots simultaneously, yet they still produce a lot of redundant knots.

Targeting on the limitations of existing methods, we propose a computationally efficient framework to calculate knots for splines fitting via sparse optimization. The framework is composed of two stages: firstly we solve a convex sparse optimization model starting from a dense initial knot vector. The output is those knots (which we call active knots) at which a certain order derivatives of the fitting spline is discontinuous. The idea to formulate the optimization model in this step is the same as that in [21] but with a distinct formulation. Secondly, we adjust the active knots in the first stage by certain rules to remove redundant knots. Furthermore, several theoretical results about the algorithm are established in this paper. In particular, when the data points are sampled dense enough from a spline, the knots of this spline can be recovered by the proposed framework in any given precision.

The remainder of the current paper is organized as follows. In Section 2, we review some preliminary knowledge about B-splines and least-square fitting with B-splines. In Section 3, a two-stage framework of curve fitting with B-splines is described. Some related theoretical results are also presented. In Section 4, we illustrate the effectiveness of the proposed method through numerical experiments and comparisons with existing methods. Finally, in Section 5, we conclude the paper with discussions on future research problems.

2. Preliminaries

We refer to the fundamental book [1] for a complete treatment of splines. Here we simply introduce the adopted notations which are needed for presenting our results.

2.1. B-splines

Let $\{c_i\}_{i=0}^n \in \mathbb{R}^d$ be $n+1$ control points, and $N_i^p(t)$ be the B-spline basis functions of degree p defined on a knot vector $U = \{t_0, t_1, \ldots, t_{n+p+1}\}$ with $t_i \le t_{i+1}, i = 0, 1, \ldots, n+p$, then a B-spline curve of degree p is defined by

$$c(t) = \sum_{i=0}^{n} c_i N_i^p(t), \tag{1}$$

where $N_i^p(t)$ is defined recursively as follows:

$$N_i^0(t) = \begin{cases} 1, & t \in [t_i, t_{i+1}) \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$N_i^p(t) = \frac{t - t_i}{t_{i+p} - t_i} N_i^{p-1}(t) + \frac{t_{i+p+1} - t}{t_{i+p+1} - t_{i+1}} N_{i+1}^{p-1}(t), \quad p \ge 1. \tag{3}$$

When $d = 1$, $c(t)$ is called a B-spline function and the control points are called spline coefficients. In this paper, we only consider data fitting with B-spline functions.

U is often chosen as an open knot vector, namely boundary knots are set to $a = t_0 = t_1 = \cdots = t_p$, $t_{n+1} = \cdots = t_{n+p+1} = b$. The knots t_i, $i = p+1, \ldots, n$ are called interior knots of U. The multiplicity of an interior knot t_i is denoted by m_i ($m_i \le p+1$). An interior knot t_i is called an *active knot* of $c(t)$ if the $(p+1-m_i)$th order derivative of $c(t)$ is discontinuous at t_i, otherwise it is called an *inactive knot* of $c(t)$. Fig. 1 shows a C^2 continuous cubic B-spline function and its first three derivatives. The spline has 9 interior knots (marked by crosses) and 3 active knots (marked by black crosses) where the third order derivatives are discontinuous.

The kth order derivative of $c(t)$ is a B-spline of degree $p-k$:

$$c^{(k)}(t) = \Pi_{i=1}^{k}(p+1-i) \sum_{i=k}^{n} c_i^{(k)} N_i^{p-k}(t), \tag{4}$$

with

$$c_i^{(k)} = \begin{cases} c_i, & \text{if } k = 0, \\ \dfrac{c_i^{(k-1)} - c_{i-1}^{(k-1)}}{t_{i+p+1-k} - t_i}, & \text{if } k > 0. \end{cases} \tag{5}$$

The Fourier transform of the jth basis function $N_j^p(t)$ is defined as

$$\widehat{N}_j^p(t) = \int_{-\infty}^{-\infty} N_j^p(t) e^{iwt} dt = \frac{(p+1)!}{(iw)^{p+1}} \sum_{k=j}^{p+1+j} \frac{e^{iwt_k}}{\theta'(t_k)}, \tag{6}$$

where $\theta(t) = \Pi_{k=j}^{p+1+j}(t - t_k)$, $w \in \mathbb{R}$ represents frequency. For uniformly distributed knots, the Fourier transform can be simplified as:

$$\widehat{N}_j^p(t) = \left(\frac{e^{i|\tau|w} - 1}{i|\tau|w} \right)^{p+1}, \tag{7}$$

where $|\tau|$ is defined as $|\tau| = \max_i (t_{i+1} - t_i)$.

2.2. Least squares approximation by splines

Given a set of data $\{P_i\}_{i=1}^N$, and corresponding parameter values $\{s_i\}_{i=1}^N$, the least square approximation with splines is defined

$$\min_{c(t)} \sum_{i=1}^{N} (c(s_i) - P_i)^2, \tag{8}$$

where $c(t)$ is a spline function defined by (1). When the knot vector U of the spline function $c(t)$ is fixed, problem (8) is reduced to

$$\min_{C \in \mathbb{R}^{n+1}} \|P - AC\|^2, \tag{9}$$

where $P = (P_1, \ldots, P_N)^T$, $A = (a_{ij})_{N \times (n+1)}$ with $a_{ij} = N_j^p(s_i)$, and $C = (c_0, c_1, \ldots, c_n)^T$ is the coefficient vector.

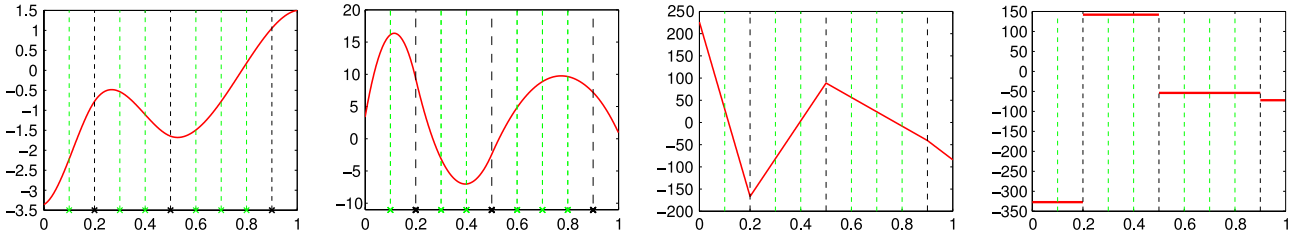
If A has rank $n+1$, then A^TA is nonsingular, thus the solution C of problem (9) is obtained uniquely by

$$A^TAC = A^TP. \tag{10}$$

The sufficient and necessary conditions for A to have rank $n+1$ are stated by Schoenberg and Whitney [28]. If A does not have full rank, the solution C is defined as the solution which minimizes $\|C\|_2$ among all the solutions of (10).

3. Knot calculation for spline fitting

In this section, we will present a two-stage framework of knot calculation for spline fitting in detail. We start with an outline of the algorithm. Then the sparse optimization model and knot adjustment strategy are described respectively.



(a) $c(t)$. (b) $c^{(1)}(t)$. (c) $c^{(2)}(t)$. (d) $c^{(3)}(t)$.

Fig. 1. A cubic B-spline function and its first three derivatives. The active knots are indicated by black crosses and the inactive knots are indicated by green crosses.

3.1. Outline of knot calculation

For data fitting with spline, it is often expected that the number of knots should be as few as possible under the condition that the fitting performance is good enough. Suppose $c(t)$ is defined as in (1) with an initial knot vector $U = \{t_0, t_1, \ldots, t_{n+p+1}\}$, then the number of valid knots can be described by the number of nonzero elements in the vector $v = (|c^{(p)}(t_j^+) - c^{(p)}(t_j^-)|)_{j=p+1}^{j=n}$ which describes the jumps of pth order derivatives of $c(t)$ at the interior knots, because $c^{(p)}(t)$ now is a piecewise constant function and the breakpoints are interior knots. Thus to minimize the number of knots, one can apply l_0-norm minimization technique to vector v, which has been widely used in computer vision and signal processing.

Let $\{P_i\}_{i=1}^N$ be given data points with corresponding parameters $\{s_i\}_{i=1}^N$, and $c(t)$ be a B-spline function defined by Eq. (1). Then the fitting problem is formulated as

$$\min_{\substack{c_i \in \mathbb{R}, i=0\ldots n \\ t_j, j=p+1,\ldots,n}} \quad \|Jc^{(p)}\|_0 \tag{11a}$$

$$s.t. \sum_{i=1}^{N} (c(s_i) - P_i)^2 \le N\varepsilon \tag{11b}$$

where the l_0 norm $\|\cdot\|_0$ indicates the number of nonzero elements in the vector, $\varepsilon > 0$ is a fixed tolerance to control the quality of fit and $Jc^{(p)}$ is the jump vector of pth order derivatives of $c(t)$ at the interior knots, defined as

$$Jc^{(p)} = (Jc^{(p)}(t_{p+1}), \ldots, Jc^{(p)}(t_n))$$

with $Jc^{(p)}(t_j) = c^{(p)}(t_j^+) - c^{(p)}(t_j^-), j = p+1, \ldots, n$. For small p, an explicit formula for $Jc^{(p)}$ can be written down based on (5). In this paper, we will restrict our experiments on fitting with cubic spline functions.

Problem (11) has three parts of unknowns: spline coefficients $\{c_i\}_{i=0}^n$, knot number n and interior knot vector $\{t_i\}_{i=p+1}^n$. Since a spline is a non-convex and nonlinear function of knot values, problem (11) is a non-convex and nonlinear optimization problem which is very difficult to compute. To find an optimal solution, we start from a dense knot vector to search for the best possible number of knots and their locations simultaneously by solving a convex optimization problem, followed by a further knot adjustment strategy, see Fig. 2 for reference.

For the first step, an initial knot vector containing enough knots for representing the details of data is provided. We then find a spline function with provided initial knots to fit the data with least square approximation by solving problem (11) except that the knots are fixed. That means we find a spline approximation with fewest number of pth order derivative jumps. The nonzero jumps correspond to some *active knots* which are used as the input for the second step.

For the second step, our goal is to further decrease the knot number while keeping the fitting quality. Only pruning knots from the input knot vector cannot fix redundant knot problem. So here we adopt the following adjustment strategy. For each interval $[t_{i_k}, t_{i_{k+1}}]$ of the input active knot vector $\tilde{U} = \{t_{i_k}\}_{k=1}^m$, we insert the middle point t^* in the interval to obtain a new vector U^* and compute the approximated degree p spline function $c^*(t)$ by solving problem (11) with U^*. If t^* is an active knot of $c^*(t)$, then we replace the interval $[t_{i_k}, t_{i_{k+1}}]$ with one of the following two intervals $[t_{i_k}, t^*]$ and $[t^*, t_{i_{k+1}}]$. This process is repeated until the length of the interval is small enough or the interval does not contain any parameters s_i. We then merge the interval to a single knot.

As an illustration example shown in Fig. 2, data points are uniformly sampled in $[0, 1]$ with $N = 101$ from the function $f(t) = 1.0/((x - 0.5)^2 + 0.02)$. The tolerance $\varepsilon = 0.005$ and the initial knots are chosen as 11 equidistant points in $[0, 1]$. Fig. 2(a) shows the approximated B-spline (red curve) and the corresponding active knots (blue triangles) obtained in the first step. Fig. 2(b) depicts the approximated B-spline together with the final five interior knots by the second step. The five interior knots are (0.2000, 0.3941, 0.5000, 0.6066, 0.8000), where the knots 0.3941 and 0.6066 are the results of performing adjustment strategy on the two intervals (0.3, 0.4) and (0.6, 0.7) respectively.

3.2. Sparse fitting

In this section, we firstly present details of formulating and solving the sparse optimization model, then we provide some theoretic results about the properties of such a problem, which are helpful for understanding and designing strategy in second stage.

3.2.1. Sparse optimization model

Suppose we have fixed an initial knot vector $U = \{t_0, t_1, \ldots, t_{n+p+1}\}$, then problem (11) can be reformulated as

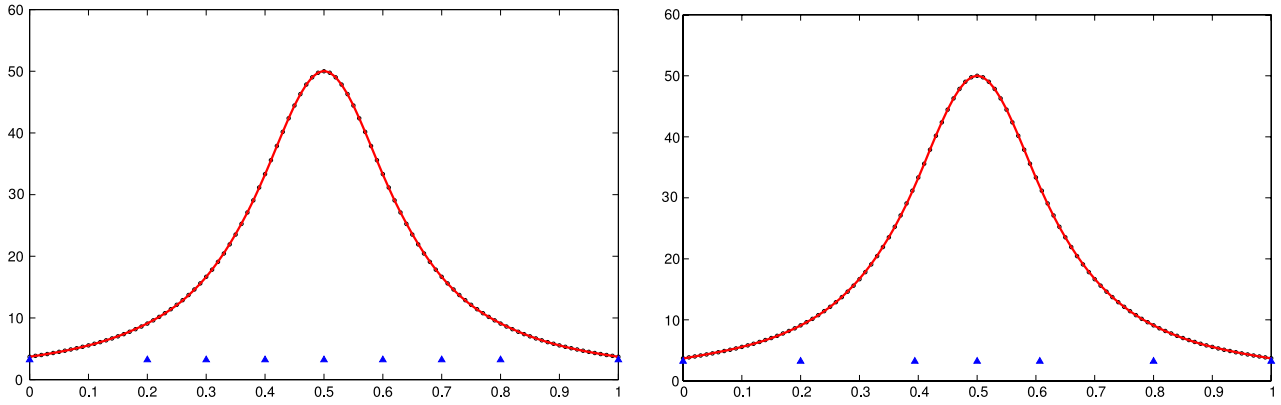
$$\min_{c_i \in \mathbb{R}, i=0\ldots n} \|Jc^{(p)}\|_1 \tag{12a}$$

$$s.t. \sum_{i=1}^{N} (c(s_i) - P_i)^2 \le N\varepsilon. \tag{12b}$$

Here we change l_0 norm of objective function to l_1 norm for the ease of solve:

$$\|Jc^{(p)}\|_1 \equiv \sum_{j=p+1}^{n} |c^{(p)}(t_j^+) - c^{(p)}(t_j^-)|.$$

We call (12) as sparse optimization model. Since it is a convex optimization problem, it can be solved efficiently. For initial knots in U, they are usually chosen uniformly in domain $[a, b]$. The authors [22] proposed a way of estimating the number of knots for given sample points according to the Fourier transformation of B-splines and Nyquist–Shannon sampling theorem [29]. The idea is as follows. A basis function can be treated approximately as a band limited signal with band width $[-2\pi/|\tau|, 2\pi/|\tau|]$ from Eq. (7). Additionally, there are N data points in the interval $[a, b]$, so the sampling rate ω equals $\frac{N}{(b-a)}$. If we want to capture all the information



(a) The result solved by first step. (b) The result solved by second step.

Fig. 2. Workflow of knot calculation. (a) The blue triangles are marked for the active knots of the approximated B-spline (red curve) solved by first step. (b) The knots (blue triangles) and the corresponding approximated B-spline (red curve) obtained by second step.

contained in the given data, then the basis functions should contain frequency components with frequency at least $\omega/2$ according to Nyquist–Shannon sampling theorem, that is $2\pi/|\tau| \geq \omega/2$, hence $|\tau| \leq 4\pi/\omega$. Then the number of interior knots of the knot vector should be at least $1/|\tau| \geq \frac{N}{4\pi(b-a)}$. For example, suppose $a = 0$, $b = 1$, $N = 1001$, then there should be at least 80 interior knots.

It should be pointed out that the above estimate only provides a lower bound for initial knot number. In practice, initial knots are often selected more than the lower bound to capture the fine features of data points.

When the jump $|c^{(p)}(t_j^+) - c^{(p)}(t_j^-)|$ is close to zero, the corresponding knot t_j is pruned. The remaining knots are called *active knots*. When problem (12) is solved, the active knots of $c(t)$ can be consequently determined. The objective function in problem (12) favors $c(t)$ has few number of active knots. Consequently most of the initial knots are pruned. Notice that $c(t)$ can be represented as a spline defined over the active knots, that is, inactive knots can be deleted without changing the spline function.

3.2.2. Characteristic of sparse model

In this section, we study the solution of problem (12) when data points are sampled dense enough from a spline function. We illustrate it with $p = 0$. Suppose data points $\{P_i\}_{i=1}^N$ with corresponding parameter values $\{s_i\}_{i=1}^N$ are sampled from a piecewise constant function with breakpoints $k_1 < \cdots < k_l$, and a B-spline $c(t)$ of degree $p = 0$ defined over an initial knot vector $U = \{t_0, t_1, \ldots, t_n, t_{n+1}\}$ is used to fit the data points via problem (12). We are interested in the characteristic of spline $c(t)$ at the interval $I_{i_0} = [t_{i_0}, t_{i_0+1}]$ which contains a breakpoint.

Without loss of generality, suppose $k_i \in [t_{i_0}, t_{i_0+1}]$ and the value sampled at k_i is shown in Fig. 3(a). For convenience, $c_{i-1} < c_i$ is assumed. The value of $c(t)$ at the three intervals $I_{i_0-1}, I_{i_0}, I_{i_0+1}$ are denoted by d_1, d_2, d_3 respectively. We also suppose the data points in the two adjacent intervals of I_{i_0} have been fitted well, that is $d_1 = c_{i-1}$, $d_3 = c_i$, see Fig. 3(b) for reference. Notice that the optimal value of problem (12) does not depend on d_2 as long as $c_{i-1} \leq d_2 \leq c_i$ since the sum of the jump in $c(t)$ at t_{i_0} and t_{i_0+1} equals $|d_2 - d_1| + |d_3 - d_2| = d_3 - d_1 = c_i - c_{i-1}$. But the fitting error $E = \sum_{i=1}^N (c(s_i) - P_i)^2$ depends on d_2. Particularly, the fitting error at an interval I_{i_0} is defined by $E_{i_0} = \sum_{s_i \in I_{i_0}} (c(s_i) - P_i)^2$. We are going to study how the fitting error E_{i_0}, thus E, changes when a midpoint is inserted in I_{i_0}. There are two cases.

(1) First case: none of the parameters $\{s_i\}_{i=1}^N$ lies in I_{i_0}. Fig. 3(b) shows this case, where the horizontal dashed (solid) line segments are marked for the value of true solution (approximated solution) on I_{i_0}. For this case, we merge the endpoints t_{i_0} and t_{i_0+1} as the midpoint $t^* = 0.5(t_{i_0} + t_{i_0+1})$, and the resulting knot vector is denoted by

$$U^* = \{t_0, \ldots, t_{i_0-1}, t^*, t_{i_0+2}, \ldots, t_{n+1}\}.$$

A spline $c^*(t)$ defined on U^* is constructed as shown in Fig. 3(c). Substituting $c^*(t)$ in problem (12), it can be seen that both the objection value and fitting error do not change comparing to that of $c(t)$. But the number of knots of $c^*(t)$ is one less than that of $c(t)$.

(2) Second case: among s_i, $i = 1, 2, \ldots, N$, there are n_1 parameters in $[t_{i_0}, k_i]$ and n_2 parameters in $[k_i, t_{i_0+1}]$. Then the value d_2 should be $\frac{c_{i-1}n_1 + c_i n_2}{n_1 + n_2}$ to minimize the fitting error E_{i_0} and consequently we have $E_{i_0} = \frac{n_1 n_2 (c_i - c_{i-1})^2}{n_1 + n_2}$. For this case, we insert the midpoint $t^* = 0.5(t_{i_0} + t_{i_0+1})$ in I_{i_0} and the resulting knot vector is denoted by $U^* = \{t_0, \ldots, t_{i_0}, t^*, t_{i_0+1}, \ldots, t_{n+1}\}$. A spline $c^*(t)$ defined on U^* is constructed as shown in Fig. 3(d), where $d_2^* = \frac{c_{i-1}n_0 + c_i n_2}{n_0 + n_2}$ and the corresponding fitting error at I_{i_0} is denoted by $E_{i_0}^* = \frac{n_0 n_2 (c_i - c_{i-1})^2}{n_0 + n_2}$, where n_0 is the number of parameters of s_i lying in $[t^*, k_i]$ (or $[k_i, t^*]$). Comparing to $c(t)$, the objection value of problem (12) do not change, but the fitting error $E_{i_0}^* < E_{i_0}$ if $n_0 < n_1$.

The above analysis inspires us to subdivide I_{i_0} into two subintervals $[t_{i_0}, t^*]$ and $[t^*, t_{i_0+1}]$, then replace I_{i_0} by one of the subintervals to make the fitting error decrease. This process can be repeated until the length of the interval is small enough or the interval does not contain any parameter s_i. The two end points of the interval is then merged into one single knot.

For general case $p \neq 0$, when the given data points are sampled from a B-spline of degree p, we have a similar conclusion. We summarize the observation in the following theorem.

Theorem 3.1. *Suppose the data points are sampled dense enough from a B-spline of degree p with true knots $k_1 < \cdots < k_l$, $\varepsilon > 0$ is small enough, and the initial knot vector is $U = \{t_0, t_1, \ldots, t_{n+p+1}\}$. The active knots selected by problem (12) are collected increasingly in a vector $\tilde{U} = \{t_{i_1}, t_{i_2}, \ldots, t_{i_m}\}$. Then for each k_j, $j = 1, \ldots, l$, there exists j_1, $1 \leq j_1 \leq m-1$ such that $k_j \in [t_{i_{j_1}}, t_{i_{j_1+1}}]$ and $i_{j_1} + 1 = i_{j_1+1}$.*

Proof. We prove the conclusion by contradiction. Without loss of generality, suppose $k_1 \in [t_{i_1}, t_{i_2}]$, but $i_1 + 1 \neq i_2$. $c(t)$ is the approximated B-spline solved by problem (12) with knot vector U and parameter ε. The optimal value of problem (12) with respect to $c(t)$ is denoted as TV. As assumed $i_1 + 1 \neq i_2$, then there exists a knot t^* in U such that $t_{i_1} < t^* < t_{i_2}$. And t^* must be an inactive knot of $c(t)$.

Now we consider a new B-spline $c^*(t)$ of degree p defined on a knot vector U^* which is defined as follows: If $k_1 \in [t^*, t_{i_2}]$,

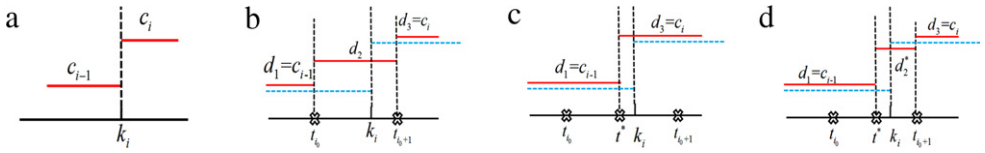
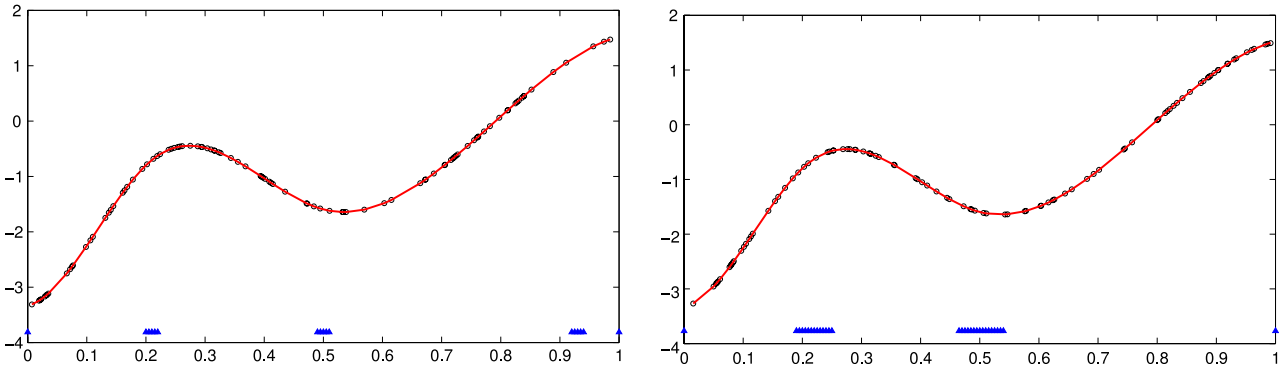


Fig. 3. Optimal solutions of constant piecewise function. The horizontal dashed (solid) line segments are marked for the value of true solution (approximated solution) on I_{i_0}.



(a) $\varepsilon = 10^{-6}$. (b) $\varepsilon = 10^{-3}$.

Fig. 4. The results solved by the first step with different tolerances. The blue triangles are marked for the active knots of the approximated B-spline (red curve) and the black circles are data points.

$U^* = \{t^*, t_{i_2}, \ldots, t_{i_m}\}$; otherwise $U^* = \{t_{i_1}, t^*, t_{i_3}, \ldots, t_{i_m}\}$ (here for simplicity, we ignore the boundary knots of U^*). Let $TV^* = \|Jc^{*(p)}(t)\|_1$ as defined in problem (12), then $c^*(t)$ is determined by solving a least square approximation of the given data points with the constraint $TV^* = TV$.

Notice that the sampled B-spline function is a piecewise polynomial in $[t_{i_1}, t_{i_2}]$ since there is an active knot $k_1 \in [t_{i_1}, t_{i_2}]$. The constructed new spline $c^*(t)$ happens to be a piecewise polynomial in the interval $[t_{i_1}, t_{i_2}]$, while $c(t)$ is a continuous polynomial. This means $c^*(t)$ has one more degree of freedom than $c(t)$ when they approximate the sampled data points. So the fitting error of $c^*(t)$ is less than or equal to the one of $c(t)$, that is

$$\sum_{i=1}^{N}(c^*(s_i) - P_i)^2 \le \sum_{i=1}^{N}(c(s_i) - P_i)^2 \le N\varepsilon.$$

Firstly, it can be asserted that the fitting error of $c^*(t)$ is less than that of $c(t)$, otherwise there will be two different solutions with the same optimal value for the problem (12) which contradicts with the uniqueness of solution of problem (12).

Then, because problem (12) is a convex optimization, so the fitting error of the optimal solution must be equal to $N\varepsilon$. Thus TV^* cannot be an optimal value, then TV is not an optimal value because $TV = TV^*$, then $c(t)$ is not the optimal solution of problem (12) which contradicts with the assumption.

Above all, if $k_1 \in [t_{i_1}, t_{i_2}]$ with $i_1+1 \neq i_2$, then $c(t)$ is not the optimal solution to the problem (12). In other words, if $k_1 \in [t_{i_1}, t_{i_2}]$, then it must have $i_1 + 1 = i_2$. Hence the theorem is proved. □

Remark 3.1. 1. For $p = 0$, we have $d_2 = \frac{c_{i-1}n_1 + c_i n_2}{n_1+n_2}$. Suppose $|c_i - c_{i-1}| < \eta$, then $|d_2 - c_i| = |\frac{n_1(c_i - c_{i-1})}{n_1+n_2}| < \eta\frac{n_1}{n_1+n_2} < \eta$. Analogously, we have $|d_2 - c_{i-1}| < \eta$. If η is small enough, then $d_2 \approx c_i$ or $d_2 \approx c_{i-1}$. Additionally, when $n_1 \ll n_2$ or $n_2 \ll n_1$, we also have $d_2 \approx c_i$ or $d_2 \approx c_{i-1}$. For such two cases, we may not have the conclusion in Theorem 3.1, that is the two knots t_{i_0}, t_{i_0+1} which are adjacent to k_i may not be contained in $\widetilde{U}$. For general degree p, when the jump at k_j is too small or the parameters of data points around k_j are distributed unevenly, it may have $i_{j_1} + 1 \neq i_{j_1+1}$, that is $t_{i_{j_1}}$ or $t_{i_{j_1+1}}$ may not be in $\widetilde{U}$. Thus in the second step the true knots cannot be recovered precisely or even omitted in some cases. Fortunately such kinds of knots always have little effect on fitting performance and there is no redundancy phenomenon around them from the experiments.

2. If some true knots are contained in several adjacent intervals, such as $k_i \in [t_{i_0}, t_{i_0+1}]$, $k_{i+1} \in [t_{i_0+1}, t_{i_0+2}]$, . . ., then the conclusion in Theorem 3.1 may not be true. So the initial knots are always chosen dense enough to avoid this case.

We take Fig. 4 as an illustration for Theorem 3.1. Data points are randomly sampled in [0, 1] with $N = 101$ from a B-spline associated with three interior knots 0.21, 0.5, and 0.93. The tolerance $\varepsilon = 10^{-6}$ and the 201 initial knots are chosen uniformly from [0, 1]. Fig. 4(a) shows the approximated B-spline (red curve) and the associated active knots (blue triangles) selected in the first step. For each true knot, the two initial knots adjacent to it are selected as active knots. It should be pointed out that the requirement (ε should be small enough) in Theorem 3.1 is necessary. Fig. 4(b) shows the active knots and the corresponding approximated B-spline for the case $\varepsilon = 0.001$, where the two knots adjacent to knot 0.93 are not included in the active knots set.

Fig. 4 also illustrates that the knots of the approximated splines only by the first step are still much more than expected. Fortunately these active knots are clustered into three groups cleanly and are distributed intensively around a true knot. Such characteristic is not special when the data are sampled from a spline function. The following subsection will discuss how to design the knot adjustment strategy on account of this characteristic.

3.3. Locally adjust knot position

Let U be the initial knot vector, $\widetilde{U}$ be the set of active knots selected by problem (12) and $k_1 < k_2 < \cdots < k_l$ be the real knots of a given B-spline function $f(t)$ as defined in Theorem 3.1. Obviously $\widetilde{U}$ is a subset of U. And if $[t_j, t_{j+1}]$ contains a knot of the given B-spline, then t_j and t_{j+1} must be in $\widetilde{U}$ based on Theorem 3.1. And such interval can be narrowed down to a single knot of the B-spline function. In this subsection, we will discuss how to find such interval and how to merge such interval to a single knot.

For each interval $[t_{i_k}, t_{i_{k+1}}]$, we insert the middle point t^* of the interval in $\widetilde{U}$ to get a new knot vector

$$U^* = \{t_{i_1}, \ldots, t_{i_k}, t^*, t_{i_{k+1}}, \ldots, t_{i_m}\}.$$

And the sparse optimization problem (12) is solved again with U^*. According to Theorem 3.1, if there is a real knot of $f(t)$ in $[t_{i_k}, t_{i_{k+1}}]$, then t^* will be selected as an active knot, and, t_{i_k} or $t_{i_{k+1}}$ will become an inactive knot. For convenience and numerical stability, we usually find the minimal one of three jumps of the pth derivative of the approximated B-spline at three knots t_{i_k}, t^*, $t_{i_{k+1}}$ and if the jump at t_{i_k} or $t_{i_{k+1}}$ is the minimal one, then the interval $[t_{i_k}, t_{i_{k+1}}]$ is considered as the candidate interval which will be narrowed down to a single knot. We conclude this idea in the function $flag$ = **IsCandidateInterval** $(i, \mathbf{t}, Err)$.

For each candidate interval $[t_{i_k}, t_{i_{k+1}}]$, we replace interval $[t_{i_k}, t_{i_{k+1}}]$ with either $[t_{i_k}, t^*]$ or $[t^*, t_{i_{k+1}}]$ depending on the local fitting error, where t^* is the middle point of the interval $[t_{i_k}, t_{i_{k+1}}]$. Or in other words, we replace either knot t_{i_k} or $t_{i_{k+1}}$ with t^*. This process is repeated until the interval does not contain any parameters s_i or the length of the interval is small enough. We then merge the two end points of the interval into a single knot.

The detailed procedure for adjusting knots is summarized in Algorithm 1.

Algorithm 1 Locally adjust the knots position (I)

Input: threshold tol, knot vector $\widetilde{U}$.
Output: knot vector U^*.

Initialization $U^* = \widetilde{U}$; $i = 1$;
  Err = **LeastSquareFit**(U^*);

while$(i < size(U^*))$ {
  $\mathbf{t} = [U^*(1:i)\ \ 0.5(U^*(i) + U^*(i+1))\ \ U^*(i+1:end)]$;
  $flag$ = **IsCandidateInterval**$(i, \mathbf{t}, Err)$;
  if $1 == flag$
    $pre = U^*(i)$; $next = U^*(i+1)$;
    while($next - pre > tol$) {
      $mid = 0.5(next + pre)$;
      $\mathbf{Lt} = [U^*(1:i)\ \ mid\ \ U^*(i+2:end)]$;
      $Lerr$ = **LeastSquareFit**(**Lt**);
      $\mathbf{Rt} = [U^*(1:i-1)\ \ mid\ \ U^*(i+1:end)]$;
      $Rerr$ = **LeastSquareFit**(**Rt**);
      if $Rerr < Lerr$
        $U^* = \mathbf{Rt}$;
      else
        $U^* = \mathbf{Lt}$;
      endif
      $pre = U^*(i)$; $next = U^*(i+1)$;
    } **endwhile**
    $U^*(i) = 0.5(pre + next)$; $U^*(i+1) = []$;
  endif
    $i = i + 1$;
} **endwhile**

Algorithm 2 Err = **LeastSquareFit**$(\mathbf{t})$

Input: knot vector **t**.
Output: fitting error Err.

  solve problem (9) with **t**;
  $Err = \sum_{i=1}^{N}(c(s_i) - P_i)^2/N$;
return Err.

In general, for data points sampled from a spline function, the strategy of finding a candidate interval can be simplified. The simplification is based on the observation that the selected active knots by the first step are separated into several groups naturally. So firstly we separate the input knots $\widetilde{U}$ by the function G = **ClusterClassify**$(\delta, \widetilde{U})$ according to knot spacing δ. In detail, suppose $\widetilde{U}(i)$ belongs to the jth group G_j, if $\widetilde{U}(i+1) - \widetilde{U}(i) > \delta$, then $\widetilde{U}(i+1)$ belongs to the $(j+1)$th group G_{j+1}. The knot spacing δ equals the length of two adjacent interior knots in the initial knot vector U. With this function, the candidate intervals are found without solving problem (12) again.

Algorithm 3 $flag$ = **IsCandidateInterval** $(i, \mathbf{t}, Err)$

Input: interval index i, knot vector **t** and error Err.
Output: interval flag $flag$.

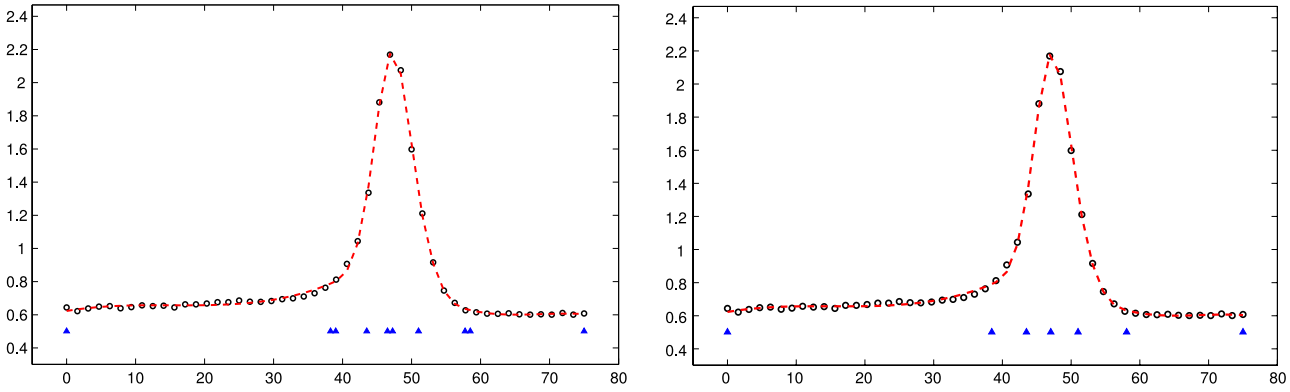
  $flag = 1$;
  solve problem (12) with $U := \mathbf{t}$ and $\varepsilon := Err$;
  $J_l = |c^{(p)}(t^+_{i+l}) - c^{(p)}(t^-_{i+l})|$, $l = 0, 1, 2$;
  if $J_1 < J_0$ && $J_1 < J_2$
    $flag = 0$;
  endif
return $flag$.

In addition, in order to allow inserting multiple knots in an interval, a comparison between E_d and E_p is added, where E_d and E_p are the fitting errors of the least square approximated B-spline defined on **Mdt** and **Mpt** respectively. Here **Mdt** and **Mpt** are the knot vectors obtained by inserting a single knot and a double knot in $\widetilde{U}$, which are defined in Algorithm 4. We summarize this adjustment strategy in Algorithm 4.

Algorithm 4 Locally adjust knots position(II)

Input: knots spacing δ, threshold tol, knot vector $\widetilde{U}$.
Output: knot vector U^*.

Initialization $i = 1$; G = **ClusterClassify**$(\delta, \widetilde{U})$; $U^* = \widetilde{U}(G)$
while$(i < size(U^*))$ {
  $mid = 0.5(U^*(i) + U^*(i+1))$
  $\mathbf{Mdt} = [U^*(1:i)\ \ mid\ \ U^*(i+1:end)]$;
  E_d = **LeastSquareFit**(**Mdt**);
  $\mathbf{Mpt} = [U^*(1:i)\ \ mid\ \ mid\ \ U^*(i+1:end)]$;
  E_p = **LeastSquareFit**(**Mpt**);
  if $2E_p < E_d$
    $flag = 1$;
  else
    $flag = 2$;
  endif
  $pre = U^*(i)$; $next = U^*(i+1)$;
  while($next - pre > tol$) {
    $\mathbf{Lt} = [U^*(1:i)\ \ 0.5(next + pre)\ \ U^*(i+2:end)]$;
    $Lerr$ = **LeastSquareFit**(**Lt**);
    $\mathbf{Rt} = [U^*(1:i-1)\ \ 0.5(next + pre)\ \ U^*(i+1:end)]$;
    $Rerr$ = **LeastSquareFit**(**Rt**);
    if $Rerr < Lerr$
      $U^* = \mathbf{Rt}$;
    else
      $U^* = \mathbf{Lt}$;
    endif
    $pre = U^*(i)$; $next = U^*(i+1)$;
  } **endwhile**
    if $2 == flag$
      $U^*(i) = 0.5(pre + next)$; $U^*(i+1) = []$;
    else
      $U^*(i) = U^*(i+1) = 0.5(pre + next)$;
    endif
    $i = i + 1$;
} **endwhile**



(a) First step. (b) Second step.

Fig. 5. Approximation of Titanium heat data by our algorithm with $MSE = 0.014128$. The circles are sample points and the dashed line is the approximated spline by our method. The triangles in (a) and (b) are marked for the knots optimized by the first step and the second step respectively.

Algorithm 5 G = **ClusterClassify** $(\delta, \widetilde{U})$

Input: threshold δ, knot vector $\widetilde{U}$.
Output: group start indexes and end indexes G.

```
k = 1; G(1, 1) = 2;
for i = 2 : size(Ũ) − 1
  if Ũ(i + 1) − Ũ(i) > δ
    G(k, 2) = i;
    G(k + 1, 1) = i + 1;
    k = k + 1;
  endif
endfor
G(k, :) = []; G = reshape(G, 2(k − 1), 1);
```

return G.

Now we are ready to describe the algorithm to fit data points by a B-spline.

Input	data points $\{(s_i, P_i)\}_{i=1}^N$, degree p, initial knot vector U, tolerance ε.
Output	approximated B-spline $c^*(t)$.
Step 1	Solve optimization problem (12) and the resulting active knots are denoted by $\widetilde{U}$.
Step 2	Locally adjust the knots in $\widetilde{U}$ by **Algorithm 1** or **Algorithm 4**. The resulting knot vector is regarded as U^*.
Step 3	Solve least squares problem (9) and the approximated spline is the output $c^*(t)$.

The proposed Algorithm of fitting with B-splines have the following convergence property.

Theorem 3.2. *Suppose the data points are sampled dense enough from a B-spline of degree p with true knots $K = \{k_1 \le \cdots \le k_l\}$, then the final knot vector U^* obtained by the above algorithm is an approximation of the true knots K with the given tolerance tol. More precisely, for every true knot k_i, there is a knot $t_{i_k} \in U^*$ satisfying $|t_{i_k} - k_i| < tol$, where tol is the input parameter in Algorithm* 1.

Proof. This is obviously true based on Theorem 3.1 and Algorithm 1. □

Remark 3.2. 1. When the data points are sampled from spline function, Algorithm 4 is much faster and more effective than Algorithm 1. For general given data points, the selected knots by the first step do not have obvious phenomenon of cluster, so it is better to use Algorithm 1. Above all, if the knots selected by the first step are separated into several groups obviously, then Algorithm 4 is preferred.

4. Experimental examples

For all numerical examples in this section, we use cubic B-splines ($p = 3$) for least square fitting. The mean squared error (MSE for short) and the maximum error (ME for short), defined as $MSE = \sum_{i=1}^N (c(s_i) - P_i)^2/N$ and $ME = \max_{i=1,\ldots,N} \|c(s_i) - P_i\|$ respectively, are used to measure the fitting equality.

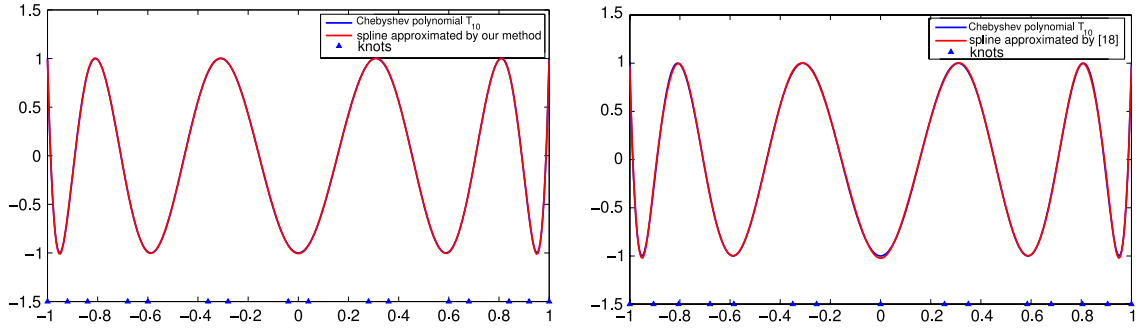
In Section 4.1, the Titanium heat data are fitted by our method to show the advantage of our method which favors fewer knots. In Section 4.2, the data points sampled from a Chebyshev polynomial of degree 10 are approximated by our method to demonstrate the effectiveness of approximation of functions. In Section 4.3, the proposed method is applied to the data points which are sampled from a B-spline. This example shows the exclusive property of recovering knots of a sampled spline with enough sampling by our method. We have implemented our algorithm on a PC with four-core Intel i5 CPU and 4GB RAM. All computations are done in Matlab and the CVX [30] package is called for solving sparse optimization problem (12).

4.1. Titanium heat data

Firstly we consider the Titanium heat data with 49 observations [31]. The data are hard to fit well by traditional techniques due to the sharp peak in the data, and have therefore often been used to test spline fitting algorithms [11,12,31,32]. The data originally ranged from [595, 1075] and are normalized to [0, 75] in this paper. The normalization using linear change of scale has no effect on the optimal knot distribution.

The initial knots are chosen as 101 equidistant knots in [0, 75] and $\varepsilon = 0.0017$. Fig. 5(a) shows the eight active knots (triangles) selected in the first step and the corresponding fitted spline (dashed line). Fig. 5(b) shows the approximated B-spline together with five interior knots by the second step. The fitting error MSE of our fit is 0.014128.

Table 1 lists the interior knots and residual error of the approximated spline using our method and other two works [21,22]. The residual error is defined by $(\frac{\sum_{i=1}^N w_i (c(s_i)-P_i)^2}{N-1})^{1/2}$ with $w_1 = w_N = 1/2$, $w_i = 1, i = 2, \ldots, N-1$. From the comparisons in Table 1, we can see that the second step of our method indeed reduces redundant knots and favors one less knot comparing to the other two methods [21,22], which indicates the significance of the second step. Additionally, the five knots calculated by our method are very close to the five knots selected by Jupp's method [11] in which it has been identified that the five interior knots are optimal when the knots number is 5. However, the knots number is fixed in advance by Jupp's method while is found automatically by our method. The last line in Table 1 shows the five optimal knots selected by Jupp's method.



(a) $ME = 1.7258\text{e}{-2}$. (b) $ME = 3.7530\text{e}{-2}$.

Fig. 6. From (a) to (b): approximation of the Chebyshev polynomial T_{10} using our method and the method in [20] respectively.

Table 1
Comparisons of approximation performance for Titanium heat data.

	Interior knots	Residual error
Iteration Algorithm [21]	(38.25, 39.00, 43.50, 46.50, 47.25, 51.00)	1.4066e−2
Adaptive Algorithm [22]	(18.00, 39.34, 44.82, 46.61, 50.13, 58.33)	1.7695e−2
Proposed algorithm	(38.41, 43.50, 47.04, 51.00, 58.09)	1.4128e−2
Jupp's algorithm [11]	(37.65, 43.97, 47.37, 50.12, 59.20)	1.2270e−2

4.2. Sampled from a function

In this subsection, we approximate the Chebyshev polynomial T_{10} of degree 10 on the interval $[-1, 1]$ which has been considered by T. Lyche and K. Mørken [20]. The data points are sampled from T_{10} at 401 uniformly spaced points. In the paper [20], firstly the knot removal strategy was used for the linear interpolate of the data points, then the knot removal strategy was used again for the cubic spline which was transformed from the reduced linear interpolate. We recompute the approximated cubic spline by the method in [20] which has built in SISL package (https://github.com/SINTEF-Geometry/SISL) and the approximated spline together with 13 interior knots is shown in Fig. 6(b), where the knots are marked by triangles.

Fig. 6(a) shows the approximated spline together with 14 interior knots calculated by our method. The 14 interior knots are marked by triangles and the fitting error is $MSE = 3.4745\text{e}{-5}$. The initial knots are chosen uniformly 25 equidistant points in $[-1, 1]$ and the parameter is chosen as $\varepsilon = 0.003$. By the way, for this function, the second step is not performed. The final fourteen interior knots of the approximated spline are selected automatically only by the first step.

The approximated error *ME* computed by our algorithm with 14 interior knots is 0.017258 compared with 0.037530, which is computed by the knot removal algorithm [20] with 13 interior knots. As seen from Fig. 6(a), the one more knot by our algorithm is around zero and contributes to the smaller approximated error *ME* comparing to the algorithm in [20]. And the interior knots calculated by our method distribute regularly in the domain $[-1, 1]$ and are selected adaptively according to the underlying curvature structures of the unknown function. By the way, as the SISL algorithm was developed with a great deal of optimization, the SISL implementation is faster than our algorithm.

4.3. Sampled from a B-spline function

In this subsection, we consider a B-spline function with the following knots:

(0.0439, 0.0653, 0.2293, 0.2367, 0.4821,
0.4907, 0.5408, 0.5408, 0.6209, 0.7051, 0.9407),

where 0.5408 is a double knot. Fig. 7(a) shows this spline function. The data points are sampled uniformly 1001 points from this B-spline function. The parameter ε is set to be $2.0\text{e} - 5$ and the initial knots are chosen uniformly 501 equidistant points in [0, 1].

Fig. 7(b) shows the approximated B-spline function (red curve) together with the active knots (blue triangles) selected by the first step. Notice that the active knots are clustered into several groups and especially around the double knot 0.5408. Fig. 7(c) shows the knots of the approximated B-spline function after the second step. The *MSE* of our fit is $3.7596\text{e} - 6$. The obtained interior knots are

(0.043969, 0.065281, 0.229406, 0.236594,
0.482094, 0.490906, 0.496031, 0.540859,
0.540859, 0.620969, 0.705031, 0.940844).

If we remove the knot 0.496031 which is not in the true knot vector, the fitting error *MSE* increases to $1.469837\text{e}{-4}$.

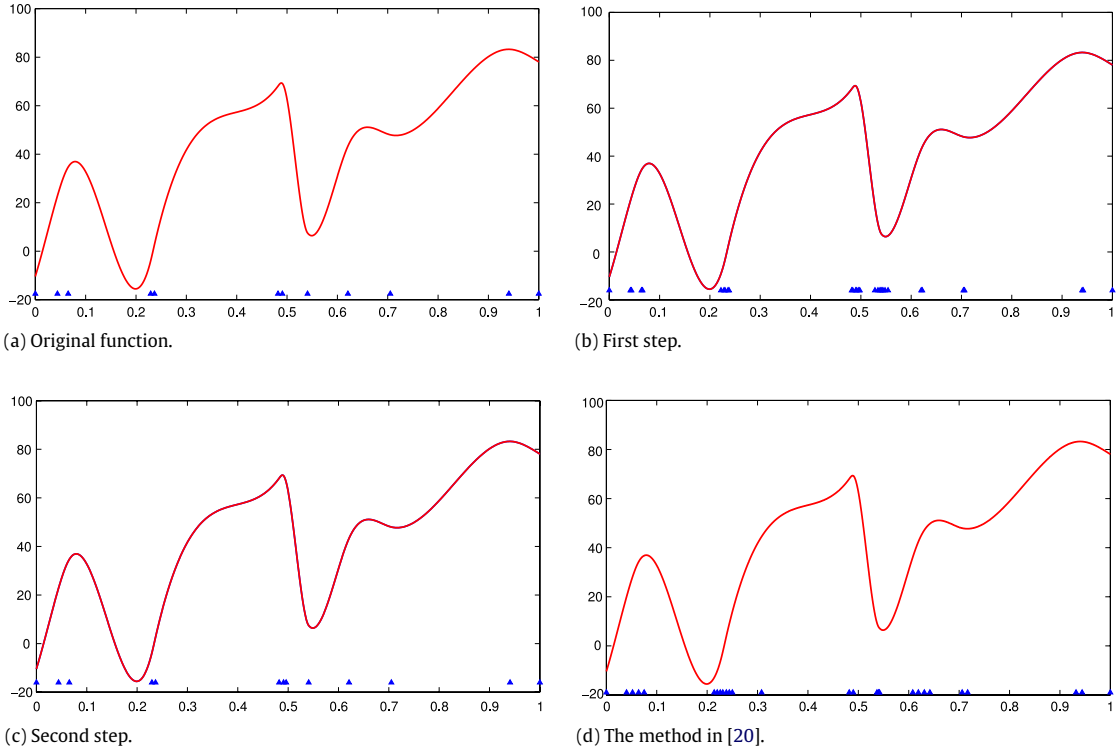
We compared our method with the three methods in the work [20–22]. Fig. 7(d) shows the approximated B-spline function together with the knots using the method in [20]. The resulting spline has 28 interior knots (0.49 is a double knot and 0.54 is a triple knot) and the maximum fitting error *ME* is $1.49802\text{e}{-2}$. The fitting error of our fit is $ME = 1.2769\text{e}{-2}$ which is smaller than that of the method in [20] and also the knots number determined by our method is fewer than that of the method in [20]. Fig. 8 illustrates the fitting result using the method in [21,22]. It can be seen that there are still much redundant knots in the approximated spline function, especially the redundant knots around the double knot 0.5408.

From the above comparison, for the data points sampled from a spline function, our method shows a significant superiority of removing redundant knots and recovering the true knots of the given spline.

Finally we discuss the effect of non-uniform sampling on recovering the true knots of a given spline function. We take the above ground truth as an illustration. Fig. 9 shows the fitting result when the data points are sampled randomly. The final interior knots are

(0.043969, 0.065282, 0.229406, 0.236598, 0.482031, 0.490906,
0.531969, 0.537969, 0.537969, 0.620099,
0.705406, 0.940531).

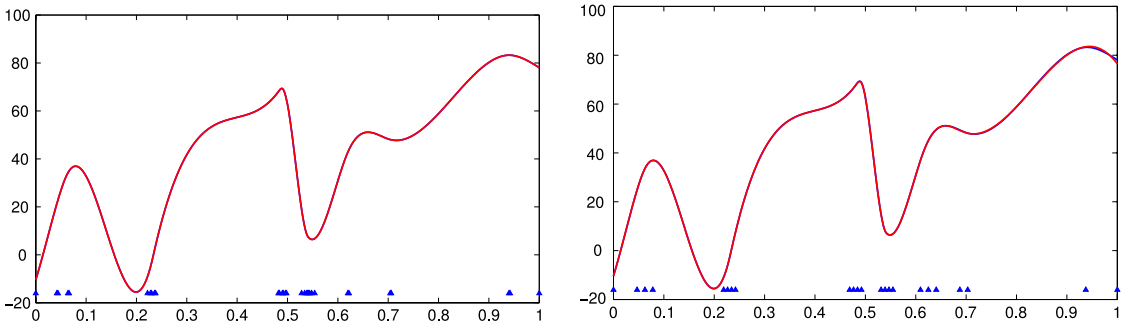
It can be seen that now the true double knot 0.5408 is recovered with a larger error than that of the uniform sampling. The worst case of non-uniform sampling is some of the true knots especially the multiple knots cannot be recovered. The reason can be



(a) Original function. (b) First step.

(c) Second step. (d) The method in [20].

Fig. 7. Approximation of a B-spline function with a double knot 0.5408 using our method and the method in [20].



(a) $MSE = 4.0041\text{e}{-7}$. (b) $MSE = 4.6633\text{e}{-4}$.

Fig. 8. From (a) to (b): approximation of a B-spline function with a double knot 0.5408 by the method in [21,22] respectively.

explained as in Remark 3.1. The random sampling results in nonuniform distribution of data points in an interval, consequently some of the candidate intervals cannot be found exactly in the second step, leading to some true knots omitted.

Remark 4.1. 1. The code of our algorithm is not optimized thoroughly. Therefore the time cost is expensive. The time cost of the first step is about 0.22 s while the time cost of the second step is about 40 s for the above three examples. It can be seen that the time cost of first step is much smaller than that of the second step. Generally speaking, the time cost of the first step depends on the given data points and the time cost of the second step depends on the number of input active knots.

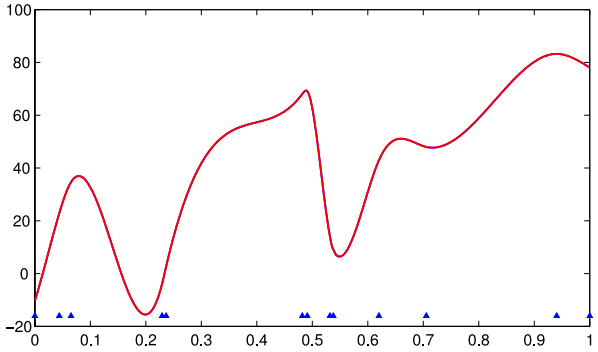


Fig. 9. Approximation of a given B-spline with random sampling.

5. Conclusions

In this paper, we propose a framework for computing knots number and position in curve fitting with splines based on a sparse optimization model. The knots number and position can be found automatically by two steps. In the first step, several knots are selected from the initial knots by solving a sparse optimization problem. There are still much more redundant knots after the first step. We further remove redundant knots and adjust the positions of active knots to obtain the final knot vector. So the resulting knot vector in the second step is not a subset of the initial knot vector given in the first step any more. This is the main difference

between our method and the two methods in [21,22]. Our experiments show that the approximation spline curve obtained by our approach has less number of knots compared to existing methods. Particularly, when the data points are sampled dense enough from a spline, our algorithm can recover the ground truth knot vector and reproduce the spline.

There are a few problems worthy of further investigation. First, the time cost now is expensive and the method can only handle data with small noise. We will investigate how to optimize the code to decrease the time cost and how to add penalty term in the sparse optimization model to deal with noisy data. Second, how to design a much more effective strategy to deal with multiple knots? A possible way is to estimate the derivatives information from given data points to locate the multiple knots. Third, how to extend the idea to parametric curves and surfaces will be an interesting but a challenging problem.

Acknowledgments

The authors thank the reviewers for providing useful comments and suggestion. The work is supported by 973 Program 2011CB302400, the National Natural Science Foundation of China (Nos. 11031007, 11371341, 11171322) and the Program NCET-11-08815.

References

[1] De Boor C. A practical guide to splines. 2001.
[2] Ma W, Kruth JP. Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces. Computer-Aided Des 1995; 27(9):663–75.
[3] Ma W, Kruth JP. Nurbs curve and surface fitting for reverse engineering. Int J Adv Manuf Technol 1998;14(12):918–27.
[4] Piegl LA, Tiller W. The nurbs book. Heidelberg, Germany: Springer Verlag; 1997.
[5] Precioso F, Barlaud M, Blu T, Unser M. Smoothing B-spline active contour for fast and robust image and video segmentation. In: International conference on image processing. Barcelona, Spain: IEEE; 2003. p. 137–40.
[6] Unser M, Aldroubi A, Eden M. B-spline signal processing ii. Efficiency design and applications. IEEE Trans Signal Process 1993;41(2):834–48.
[7] Burchard HG. Splines (with optimal knots) are better. Appl Anal 1974;3:309.
[8] De Boor C, Rice JR. Least squares cubic spline approximation, I-Fixed knots. In: Computer science technical reports. Purdue University; 1968.
[9] Rice JR. The approximation of functions, vol. 2. Reading, MA: Addison-Wesley; 1969.
[10] Rice JR. On the degree of convergence of nonlinear spline approximation. In: Schoenberg IJ, editor. Proc. symposium on approximation theory (Madison, WI), vol. 31. New York: Academic Press; 1969. p. 349–65.
[11] Jupp DLB. Approximation to data by splines with free knots. SIAM J Numer Anal 1978;15(2):328–43.
[12] Dierckx Paul. Curve and surface fitting with splines. New York, NY, USA: Oxford University Press Inc.; 1993.
[13] Loach PD, Wathen AJ. On the best least squares approximation of continuous functions using linear splines with free knots. IMA J Numer Anal 1991;11: 393–409.
[14] Beliakov Gleb. Least squares splines with free knots: global optimization approach. Appl Math Comput 2004;149(3):783–98.
[15] Miyata S, Shen X. Adaptive free-knot splines. J Comput Graph Statist 2003; 12(1):197–231.
[16] Randrianarivony M, Brunnett G. Approximation by nurbs curves with free knots, In: Proceedings of the vision, modeling, and visualization conference, Erlangen, Germany; 2002, p. 195–201.
[17] Li W, Xu S, Zhao G, Goh LP. Adaptive knot placement in B-spline curve approximation. Computer-Aided Des 2005;37(8):791–7.
[18] Razdan A. Knot placement for B-spline curve approximation. In: Technical report. Arizona State University; 1999.
[19] Lyche T, Mørken K. Knot removal for parametric b-spline curves and surfaces. Comput Aided Geom Des 1987;4:217–30.
[20] Lyche T, Mørken K. A data reduction strategy for splines with applications to the approximation of functions and data. IMA J Numer Anal 1988;8: 185–208.
[21] Van Loock W, Pipeleers G, De Schutter J, Swevers J. A convex optimization approach to curve fitting with B-splines, in: Preprints of the 18th international federation of automatic control (IFAC). Milano (Italy); 2011, p. 2290–5.
[22] Yuan Yuan, Chen Nan, Zhou Shiyu. Adaptive B-spline knots selection using multi-resolution basis set. IIE Trans 2013;45(12):1263–77.
[23] Candes E, Eldar Y, Needell D, Randall P. Compressed sensing with coherent and redundant dictionaries. Appl Comput Harmon Anal 2010;31(1):59–73.
[24] Candes E, Tao T. Decoding by linear programming. IEEE Trans Inf Theory 2005; 51(12):4203–15.
[25] Candes E, Wakin M. An introduction to compressive sampling. IEEE Signal Process Mag 2008;25(2):21–30.
[26] Donoho D. Compressed sensing. IEEE Trans Inf Theorey 2006;52:1289–306.
[27] Eldar Y, Kutyniok G. Compressed sensing with coherent and redundant dictionaries. 2012.
[28] Schoenberg IJ, Whitney Anne. On pólya frequency functions. iii: The positivity of translation determinants with an application to the interpolation problem by spline curves. Trans Amer Math Soc 1953;74(1):246–59.
[29] Shannon CE. Communication in the presence of noise. Proc IRE 1949;37(1): 10–21.
[30] Grant Michael, Boyd Stephen P. Cvx:Matlab software for disciplined convex programming, version 1.22. http://cvxr.com/cvx, May 2013.
[31] De Boor C, Rice JR. Least squares cubic spline approximation, II-variable knots. In: Computer science technical reports. Purdue University; 1968.
[32] Molinari Nicolas, Durand Jean-Francois, Sabatier Robert. Bounded optimal knots for regression splines. Comput Statist Data Anal 2004;45(2):159–78.