Computer-Aided Design 47 (2014) 32–44

Contents lists available at ScienceDirect

Computer-Aided Design

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

Progressive and iterative approximation for least squares B-spline curve and surface fitting[☆]

Chongyang Deng[a], Hongwei Lin[b,*]

[a] School of Science, Hangzhou Dianzi University, Hangzhou 310018, China

[b] Department of Mathematics, State Key Lab. of CAD & CG, Zhejiang University, Hangzhou 310027, China

HIGHLIGHTS

- A new progressive and iterative approximation method for least square fitting (LSPIA) is presented.
- LSPIA can handle a point set of large size.
- LSPIA is so flexible that it allows the adjustment of the number of control points, and a knot vector in the iterations.
- LSPIA is easy to make the fitting curve hold the shape preserving property.
- LSPIA can be performed in parallel efficiently.

ARTICLE INFO

Article history:
Received 28 June 2012
Accepted 24 August 2013

Keywords:
Progressive and iterative approximation
Least square fitting
Iteration
Geometric design

ABSTRACT

The progressive and iterative approximation (PIA) method is an efficient and intuitive method for data fitting. However, in the classical PIA method, the number of the control points is equal to that of the data points. It is not feasible when the number of data points is very large. In this paper, we develop a new progressive and iterative approximation for least square fitting (LSPIA). LSPIA constructs a series of fitting curves (surfaces) by adjusting the control points iteratively, and the limit curve (surface) is the least square fitting result to the given data points. In each iteration, the difference vector for each control point is a weighted sum of some difference vectors between the data points and their corresponding points on the fitting curve (surface). Moreover, we present a simple method to compute the practical weight whose corresponding convergence rate is comparable to that of the theoretical best weight. The advantages of LSPIA are two-fold. First, with LSPIA, a very large data set can be fitted efficiently and robustly. Second, in the incremental data fitting procedure with LSPIA, a new round of iterations can be started from the fitting result of the last round of iterations, thus saving great amount of computation. Lots of empirical examples illustrated in this paper show the efficiency and effectiveness of LSPIA.

1. Introduction

Progressive and iterative approximation (PIA) method [1,2] is an efficient and intuitive method for data fitting. It cannot only avoid the computational cost of solving a large system of linear equations, but generate a series of approximation curves or surfaces as well. However, in the classical PIA method, the number of the control points is equal to that of the data points. It is not feasible when the number of data points is very large. Although the classical PIA method is extended in Ref. [3] to approximate a given data set, the limit of the generated curve (surface) sequence is not the least square fitting (LSF) result to the data set. Certainly, the least square fitting is one of the most commonly used mathematical tools in practice. Therefore, in this paper, we devise a progressive and iterative approximation method, namely, progressive and iterative approximation for least square fitting (abbr. LSPIA), whose limit is the least square fitting result to a given data set.

Similar as the classical PIA method, LSPIA starts with an initial blending curve (surface), and constructs a series of fitting curves (surfaces) by adjusting the control points iteratively. In each iteration, the adjusting vector of each control point is a weighted sum of some difference vectors between the data points and their corresponding points on the fitting curve (surface). Compared with the traditional least square method, LSPIA has the following advantages:

- LSPIA can handle point set of large size;
- LSPIA is so flexible that it allows the adjustment of the number of control points, and knot vector in the iterations;

[☆] This paper has been recommended for acceptance by Xiaoping Qian.

[*] Corresponding author.
E-mail addresses: dcy@hdu.edu.cn (C. Deng), hwlin@zju.edu.cn (H. Lin).

0010-4485/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cad.2013.08.012

- LSPIA is easy to make the fitting curve hold the shape preserving property;
- LSPIA can be performed in parallel efficiently.

This paper is organized as follows. In Section 1.1, we briefly review the related work. Then we introduce the iterative method of LSPIA and show its convergence in Section 2. In Section 3, we study how to assign appropriate values for the weight of LSPIA. Afterwards, Section 4 presents some numerical examples. Section 5 discusses its advantages and shortcomings, and Section 6 concludes this paper.

1.1. Related work

The progressive and iterative approximation (PIA) is a new and effective technique to seek the curve or surface fitting the data points. The PIA property of the uniform cubic B-spline curve, is first discovered by Qi et al. [4] and de Boor [5], respectively. Lin et al. [1] show that the non-uniform cubic B-spline curve and surface also hold the property. Furthermore, the PIA method is extended to the blending curve and surface with NTP basis [2]. Moreover, it is proved that the rational B-spline curve and surface (NURBS) have the property, too [6]. Lu [7] devises a weighted PIA method to speed up the convergence of the PIA method. More importantly, Lin [8] discovers the local property of the PIA, by which PIA can control the fitting precision of each data point individually. Recently, the EPIA(extended PIA) method is proposed [3] to fit data using normalized totally positive (NTP) bases. And Chen et al. [9] show the convergence of the PIA method for triangular B–B patches at uniform nodes.

While the PIA method depends on the parametric distance between the data points and the corresponding foot points on the curve with the same parameters, Maekawa et al. invent an iterative fitting method, called *interpolation by geometric algorithm* [10,11], which is similar to PIA method, but relies on geometric distance between the data points and their closest points on the curve(surface). The geometric interpolation algorithm [10] is extended to approximate the vertices of a triangular mesh using Loop subdivision surface [12], and point cloud with B-spline surface [13].

The PIA method has been extended to subdivision surface fitting, named *progressive interpolation* (PI), too. Cheng et al. design the PI method of subdivision fitting for a Loop subdivision surface [14], and prove its convergence. Fan et al. develop the PI method of Doo–Sabin subdivision surface fitting [15]. The PI method for Catmull–Clark subdivision surface fitting is proposed in [16]. Very recently, Deng and Ma [17] develop the weighted PIA method for interpolating mesh by a Loop subdivision surface and give a numerical method for finding an appropriate value of the weight.

2. The progressive and iterative approximation for least square fitting (LSPIA) and its convergence

In this section, we introduce the LSPIA iterative methods for blending curves and surfaces in Sections 2.1 and 2.2, respectively.

2.1. The LSPIA iterative method for blending curves

In this section, we present the LSPIA iterative method for blending curves and show its convergence.

Assume that $\{\boldsymbol{Q}_j\}_{j=0}^m$ be an ordered point set to be fitted, and $\{0 = t_0 < t_1 < \cdots < t_m = 1\}$ be the parameters of $\{\boldsymbol{Q}_j\}_{j=0}^m$. At the beginning of the iteration, we select $\{\boldsymbol{P}_i^0\}_{i=0}^n$ from $\{\boldsymbol{Q}_j\}$ as the control point set and construct a piece of blending curve $\boldsymbol{P}^0(t)$, i.e.,

$$\boldsymbol{P}^0(t) = \sum_{i=0}^{n} B_i(t)\boldsymbol{P}_i^0, \quad t \in [t_0, t_m] \tag{1}$$

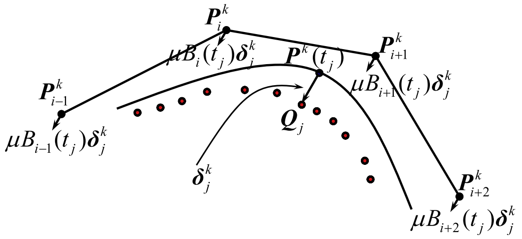

$\boldsymbol{P}_i^k$
$\mu B_i(t_j)\boldsymbol{\delta}_j^k$
$\boldsymbol{P}^k(t_j)$
$\boldsymbol{P}_{i+1}^k$
$\mu B_{i+1}(t_j)\boldsymbol{\delta}_j^k$
$\boldsymbol{P}_{i-1}^k$
$\boldsymbol{Q}_j$
$\mu B_{i-1}(t_j)\boldsymbol{\delta}_j^k$
$\boldsymbol{\delta}_j^k$
$\boldsymbol{P}_{i+2}^k$
$\mu B_{i+2}(t_j)\boldsymbol{\delta}_j^k$

Fig. 1. The distribution of δ_j^k to control points.

where $\{B_i(t); i = 0, 1, \ldots, n\}$ is a NTP blending basis. The collocation matrix of the NTP blending basis on $\{0 = t_0 < t_1 < \cdots < t_m = 1\}$ is,

$$\boldsymbol{A} = \begin{bmatrix} B_0(t_0) & B_1(t_0) & \cdots & B_n(t_0) \\ B_0(t_1) & B_1(t_1) & \cdots & B_n(t_1) \\ \cdots & \cdots & \cdots & \cdots \\ B_0(t_m) & B_1(t_m) & \cdots & B_n(t_m) \end{bmatrix}. \tag{2}$$

Remark 2.1. In theory the initial points $\{\boldsymbol{P}_i^0\}_{i=0}^n$ can be set arbitrary.

Letting

$$\boldsymbol{\delta}_j^0 = \boldsymbol{Q}_j - \boldsymbol{P}^0(t_j), \quad j = 0, 1, \ldots, m,$$

and taking the first adjusting vector for the *i*-th control point as (see Fig. 1),

$$\boldsymbol{\Delta}_i^0 = \mu \sum_{j=0}^{m} B_i(t_j)\boldsymbol{\delta}_j^0, \tag{3}$$

where μ is a constant satisfying the condition

$$0 < \mu < \frac{2}{\lambda_0}, \tag{4}$$

where λ_0 is the largest eigenvalue of $\boldsymbol{A}^T\boldsymbol{A}$, we can get the new control points by,

$$\boldsymbol{P}_i^1 = \boldsymbol{P}_i^0 + \boldsymbol{\Delta}_i^0, \quad i = 0, 1, \ldots, n,$$

and the new curve,

$$\boldsymbol{P}^1(t) = \sum_{i=0}^{n} B_i(t)\boldsymbol{P}_i^1.$$

Similarly, supposing we have gotten the *k*th curve $\boldsymbol{P}^k(t)$ after the *k*th iteration, and letting,

$$\boldsymbol{\delta}_j^k = \boldsymbol{Q}_j - \boldsymbol{P}^k(t_j), \quad j = 0, 1, \ldots, m; \tag{5}$$

$$\boldsymbol{\Delta}_i^k = \mu \sum_{j=0}^{m} B_i(t_j)\boldsymbol{\delta}_j^k, \quad i = 0, 1, \ldots, n; \tag{6}$$

$$\boldsymbol{P}_i^{k+1} = \boldsymbol{P}_i^k + \boldsymbol{\Delta}_i^k, \quad i = 0, 1, \ldots, n, \tag{7}$$

the $(k + 1)$st curve can be generated as,

$$\boldsymbol{P}^{k+1}(t) = \sum_{i=0}^{n} B_i(t)\boldsymbol{P}_i^{k+1}.$$

In this way, we get a curve sequence $\{\boldsymbol{P}^k(t)\}_{k=0}^{\infty}$. In the following Theorem 2.4, we will prove that the limit of this curve sequence is just the least square fitting result to the data points $\{\boldsymbol{Q}_j\}_{j=0}^m$.

Remark 2.2. From (6) we can see that the adjusting vector $\boldsymbol{\Delta}_i^k$ for control point $\boldsymbol{P}_i^k$ is a weighted sum of the difference vectors (5) related to $\boldsymbol{P}_i^k$, i.e., for $B_i(t_j) > 0$, the difference vector $\boldsymbol{\delta}_j^k$ is added to the adjusting vector $\boldsymbol{\Delta}_i^k$ with weight $\mu B_i(t_j)$. Note that $\sum_{i=0}^{n} B_i(t_j) = 1$, we can also see that difference vector $\boldsymbol{\delta}_j^k$ is distributed to the control points which impact $\boldsymbol{Q}_j$. So the intuitional geometric meaning of LSPIA lies in the following two aspects: as for the difference vector $\boldsymbol{\delta}_j^k$, it is distributed to control points according to the impact of that control point to $\boldsymbol{Q}_j$; as for the adjusting vector $\boldsymbol{\Delta}_i^k$, it is accumulated by the weighted difference vector of those data points which are related to $\boldsymbol{P}_i^k$.

Remark 2.3. For the case of interpolating end points $\boldsymbol{Q}_0, \boldsymbol{Q}_m$, we just let $\boldsymbol{P}_0^0 = \boldsymbol{Q}_0, \boldsymbol{P}_n^0 = \boldsymbol{Q}_m$ and $\boldsymbol{P}_0^{k+1} = \boldsymbol{P}_0^k, \boldsymbol{P}_n^{k+1} = \boldsymbol{P}_n^k$ in the iterative process.

Theorem 2.4. *The aforementioned LSPIA iterative method is convergent and the limit curve is the LSF curve of the initial data* $\{\boldsymbol{Q}_j\}_{j=0}^{m}$.

Proof. As the result of the above iterative procedure, a curve sequence $\{\boldsymbol{P}^k(t), k = 0, 1, \ldots\}$ is generated. To show its convergence, let

$$\boldsymbol{P}^k = \{\boldsymbol{P}_0^k, \boldsymbol{P}_1^k, \ldots, \boldsymbol{P}_n^k\}^T;$$

$$\boldsymbol{Q} = \{\boldsymbol{Q}_0, \boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_m\}^T.$$

According to Eq. (7), we have

$$\begin{aligned}\boldsymbol{P}_i^{k+1} &= \boldsymbol{P}_i^k + \mu \sum_{j=0}^{m} B_i(t_j)(\boldsymbol{Q}_j - \boldsymbol{P}^k(t_j))\\ &= \boldsymbol{P}_i^k + \mu \sum_{j=0}^{m} B_i(t_j)\left[\boldsymbol{Q}_j - \sum_{l=0}^{n} B_l(t_j)\boldsymbol{P}_l^k\right].\end{aligned}$$

Then, we get,

$$\boldsymbol{P}^{k+1} = \boldsymbol{P}^k + \mu \boldsymbol{A}^T(\boldsymbol{Q} - \boldsymbol{A}\boldsymbol{P}^k), \tag{8}$$

where $\boldsymbol{A}$ is the collocation matrix (2).

Letting $\boldsymbol{I}$ be the $n+1$ rank identity matrix and $\boldsymbol{D} = \boldsymbol{I} - \mu\boldsymbol{A}^T\boldsymbol{A}$, by (8) we have

$$\begin{aligned}\boldsymbol{P}^{k+1} - (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q} &= (\boldsymbol{I} - \mu\boldsymbol{A}^T\boldsymbol{A})[\boldsymbol{P}^k - (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q}]\\ &= (\boldsymbol{I} - \mu\boldsymbol{A}^T\boldsymbol{A})^2[\boldsymbol{P}^{k-1} - (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q}]\\ &= \cdots\\ &= \boldsymbol{D}^{k+1}[\boldsymbol{P}^0 - (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q}].\end{aligned} \tag{9}$$

Supposing $\{\lambda_i(\boldsymbol{D})\}(i = 0, 1, \ldots, n)$ are the eigenvalues of $\boldsymbol{D}$ sorted in non-decreasing order, we get $\lambda_i(\boldsymbol{D}) = 1 - \mu\lambda_i$, where $\lambda_i, (i = 0, 1, \ldots, n)$ are the eigenvalues of $\boldsymbol{A}^T\boldsymbol{A}$ sorted in non-decreasing order. Because $\boldsymbol{A}$ is a non-singular matrix, $\boldsymbol{A}^T\boldsymbol{A}$ is positive definite. Noting $0 < \mu < \frac{2}{\lambda_0}$, we have $0 < \mu\lambda_i < 2$, and $-1 < \{\lambda_i(\boldsymbol{D})\} < 1, i = 0, 1, \ldots, n$. It leads to $0 < \rho(\boldsymbol{D}) < 1$, where $\rho(\boldsymbol{D})$ is the spectral radius of $\boldsymbol{D}$. Therefore,

$$\lim_{k\to\infty} \boldsymbol{D}^k = (\boldsymbol{0})_{n+1}$$

where $(\boldsymbol{0})_{n+1}$ is the $n+1$ rank zero matrix.

By (9), it follows,

$$\begin{aligned}\boldsymbol{P}^{\infty} &= (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q} + \boldsymbol{D}^{\infty}[\boldsymbol{P}^0 - (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q}]\\ &= (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q}.\end{aligned}$$

It is equivalent to $(\boldsymbol{A}^T\boldsymbol{A})\boldsymbol{P}^{\infty} = \boldsymbol{A}^T\boldsymbol{Q}$, meaning that the LSPIA is convergent and the limit curve is the LSF result to the initial data. □

Remark 2.5. In many practical applications, $\boldsymbol{P}(t_0) = \boldsymbol{Q}_0, \boldsymbol{P}(t_m) = \boldsymbol{Q}_m$ are required. In a such case, $\boldsymbol{A}$ and $\boldsymbol{Q}$ are replaced by the following $\overline{\boldsymbol{A}}, \overline{\boldsymbol{Q}}$ [18].

$$\overline{\boldsymbol{A}} = \begin{bmatrix} B_1(t_1) & B_2(t_1) & \cdots & B_{n-1}(t_1)\\ B_1(t_2) & B_2(t_2) & \cdots & B_{n-1}(t_2)\\ \cdots & \cdots & \cdots & \cdots\\ B_1(t_{m-1}) & B_2(t_{m-1}) & \cdots & B_{n-1}(t_{m-1})\end{bmatrix} \tag{10}$$

and $\overline{\boldsymbol{Q}} = [\boldsymbol{Q}_1 - B_0(t_1)\boldsymbol{Q}_0 - B_n(t_1)\boldsymbol{Q}_m, \boldsymbol{Q}_2 - B_0(t_2)\boldsymbol{Q}_0 - B_n(t_2)\boldsymbol{Q}_m, \ldots, \boldsymbol{Q}_{m-1} - B_0(t_{m-1})\boldsymbol{Q}_0 - B_n(t_{m-1})\boldsymbol{Q}_m]^T$. Similar as the Proof of Theorem 2.4, using $\overline{\boldsymbol{A}}, \overline{\boldsymbol{Q}}$, we can derive the same conclusion that the limit curve of LSPIA is the LSF curve.

2.2. The LSPIA iterative method for blending surfaces

The LSPIA iterative method for blending curves can be extended to tensor product surfaces easily. We present the details for the LSPIA iterative method for blending surfaces in the following.

Assume that $\{\boldsymbol{Q}_{ij}\}_{i=0,j=0}^{m_1,m_2}$ is an ordered point set to be fitted, and $\{u_i, v_j\}_{i=0,j=0}^{m_1,m_2}$ are the parameters of $\{\boldsymbol{Q}_{ij}\}_{i=0,j=0}^{m_1,m_2}$.

At the beginning of the iteration, we select $\{\boldsymbol{P}_{hl}^0\}_{h=0,l=0}^{n_1,n_2}$ from $\{\boldsymbol{Q}_{ij}\}$ as the control point set and construct a blending surface $\boldsymbol{P}^0$, i.e.,

$$\boldsymbol{P}^0(u, v) = \sum_{h=0}^{n_1}\sum_{l=0}^{n_2} B_h(u)B_l(v)\boldsymbol{P}_{hl}^0, \quad 0 \le u,\ v \le 1 \tag{11}$$

where $\{B_h(u), B_l(v); h = 0, 1, \ldots, n_1, l = 0, 1, \ldots, n_2\}$ are NTP blending bases.

Letting

$$\boldsymbol{\delta}_{ij}^0 = \boldsymbol{Q}_{ij} - \boldsymbol{P}^0(u_i, v_j),$$

and taking the first adjusting vector for the (h, l)-th control point as

$$\boldsymbol{\Delta}_{h,l}^0 = \mu \sum_{i=0}^{m_1}\sum_{j=0}^{m_2} B_h(u_i)B_l(v_j)\boldsymbol{\delta}_{ij}^0, \tag{12}$$

where μ is a constant satisfying the condition

$$0 < \mu < \frac{4}{\hat{\lambda}_0\overline{\lambda}_0}. \tag{13}$$

Here $\hat{\lambda}_0, \overline{\lambda}_0$ are the largest eigenvalues of $\hat{\boldsymbol{A}}^T\hat{\boldsymbol{A}}, \overline{\boldsymbol{A}}^T\overline{\boldsymbol{A}}$, and $\hat{\boldsymbol{A}}, \overline{\boldsymbol{A}}$ are the collocation matrix of $\{B_h(u)\}, \{B_l(v)\}$, respectively. We can get the new control points by

$$\boldsymbol{P}_{hl}^1 = \boldsymbol{P}_{hl}^0 + \boldsymbol{\Delta}_{hl}^0, \quad h = 0, 1, \ldots, n_1,\ l = 0, 1, \ldots, n_2,$$

and the new surface,

$$\boldsymbol{P}^1(u, v) = \sum_{h}^{n_1}\sum_{l}^{n_2} B_h(u)B_l(v)\boldsymbol{P}_{hl}^1, \quad 0 \le u,\ v \le 1.$$

Similarly, supposing we have gotten the kth surface $\boldsymbol{P}^k(u, v)$ after the kth iteration, and letting

$$\boldsymbol{\delta}_{ij}^k = \boldsymbol{Q}_{ij} - \boldsymbol{P}^k(u_i, v_j), \tag{14}$$

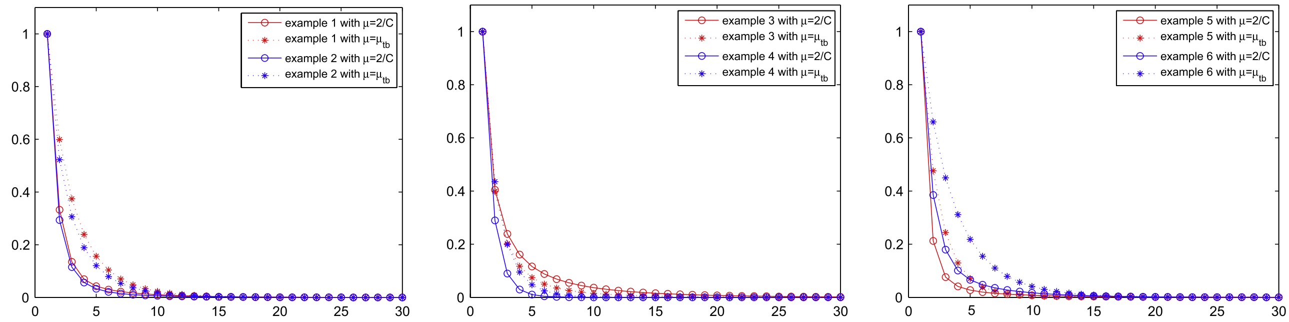
$$\boldsymbol{\Delta}_{h,l}^k = \mu \sum_{i=0}^{m_1}\sum_{j=0}^{m_2} B_h(u_i)B_l(v_j)\boldsymbol{\delta}_{ij}^k, \tag{15}$$

$$\boldsymbol{P}_{hl}^{k+1} = \boldsymbol{P}_{hl}^k + \boldsymbol{\Delta}_{hl}^k, \quad h = 0, 1, \ldots, n_1, l = 0, 1, \ldots, n_2, \tag{16}$$

the $(k+1)$st surface can be generated as,

$$\boldsymbol{P}^{k+1}(u, v) = \sum_{h}^{n_1}\sum_{l}^{n_2} B_h(u)B_l(v)\boldsymbol{P}_{hl}^{k+1}, \quad 0 \le u, v \le 1. \tag{17}$$

In this way, we get a surface sequence $\{\boldsymbol{P}^k(u, v)\}_{k=0}^{\infty}$. Similar as the proof of the curve case, the limit of this surface sequence is just the least square fitting result to the data points $\{\boldsymbol{Q}_{ij}\}_{i=0,j=0}^{m_1,m_2}$.

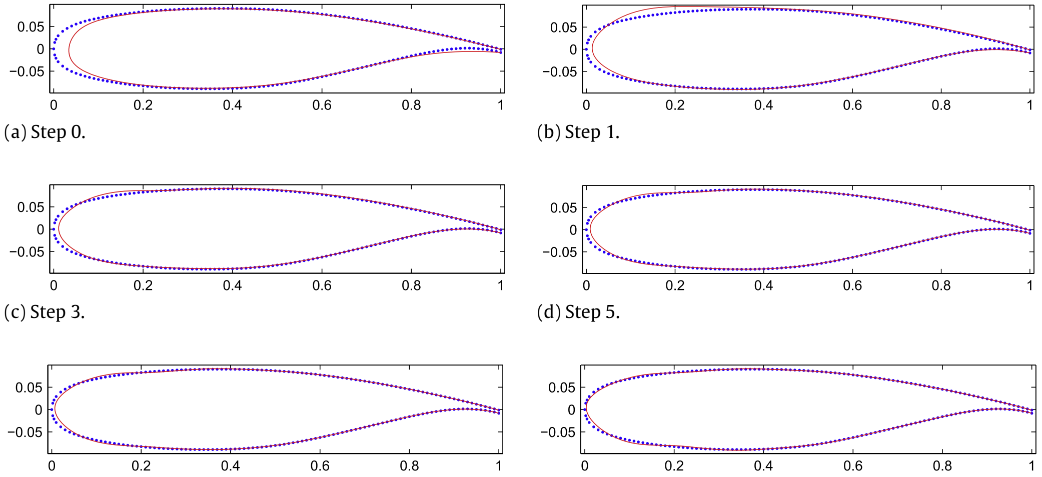


(a) Examples 1 and 2.

(b) Examples 3 and 4.

(c) Examples 5 and 6.

Fig. 2. D_k for weights $\mu = \frac{2}{C}$ and $\mu = \mu_{tb}$ (scaled such that $D_0 = 1$).



(a) Step 0.

(b) Step 1.

(c) Step 3.

(d) Step 5.

(e) Step 7.

(f) Fitting curve.

Fig. 3. An airfoil-shape data set of 205 data points is fitted by a cubic B-spline curve with 20 control points.

3. Weight selection

As shown in Eq. (6), a weight is required for generating the adjusting vector. In this section, we will discuss how to select an appropriate weight to improve the convergence rate of the LSPIA iterative methods presented in Section 2. For brevity, only the LSPIA iterative method for blending curves is handled, and the results for LSPIA iterative method for blending surfaces can be deduced similarly.

3.1. The theoretical best weight

In Theorem 3.1, we present the best weight which can lead to the fastest convergence rate.

Theorem 3.1. *The LSPIA iterative method (9) described in Section 2.1 has the fastest convergence rate when*

$$\mu = \frac{2}{\lambda_0 + \lambda_n} := \mu_{tb}, \tag{18}$$

and in such a case,

$$\rho(\boldsymbol{D}) = \frac{\lambda_0 - \lambda_n}{\lambda_0 + \lambda_n}, \tag{19}$$

where $\rho(\boldsymbol{D})$ is the spectral radius of $\boldsymbol{D}$.

Proof. Clearly, smaller the spectral radius of $\boldsymbol{D}$ is, faster the convergence rate of the iterative method (9) is. For $\mu \in (0, \frac{1}{\lambda_0}]$, we have

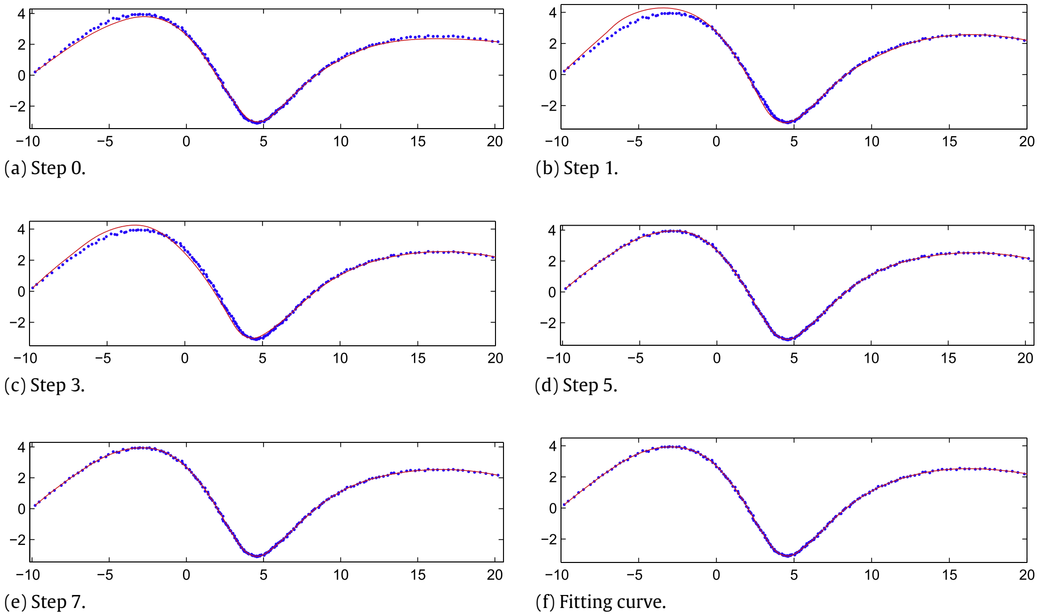
$$\begin{aligned}\rho(\boldsymbol{D}) &= \rho(\boldsymbol{I} - \mu\boldsymbol{A}^T\boldsymbol{A}) = \max\{|1-\mu\lambda_0|, |1-\mu\lambda_n|\} \\ &= 1 - \mu\lambda_n \geq 1 - \frac{\lambda_n}{\lambda_0} > \frac{\lambda_0 - \lambda_n}{\lambda_0 + \lambda_n}.\end{aligned}$$

For $\mu \in \left(\frac{1}{\lambda_0}, \frac{2}{\lambda_0}\right)$, two cases are classified as:
(1) If $\mu\lambda_n \geq 1$, then

$$\begin{aligned}\rho(\boldsymbol{D}) &= \rho(\boldsymbol{I} - \mu\boldsymbol{A}^T\boldsymbol{A}) = \max\{|1-\mu\lambda_0|, |1-\mu\lambda_n|\} \\ &= \max\{\mu\lambda_0 - 1, \mu\lambda_n - 1\} = \mu\lambda_0 - 1 \\ &= \frac{\lambda_0 - \lambda_n}{\lambda_0 + \lambda_n} + \lambda_0\left(\mu - \frac{2}{\lambda_0 + \lambda_n}\right) \\ &\geq \frac{\lambda_0 - \lambda_n}{\lambda_0 + \lambda_n} + \lambda_0\left(\frac{1}{\lambda_n} - \frac{2}{\lambda_0 + \lambda_n}\right) > \frac{\lambda_0 - \lambda_n}{\lambda_0 + \lambda_n}.\end{aligned}$$

(2) If $\mu\lambda_n < 1$, then

$$\begin{aligned}\rho(\boldsymbol{D}) &= \rho(\boldsymbol{I} - \mu\boldsymbol{A}^T\boldsymbol{A}) \\ &= \max\{|1-\mu\lambda_0|, |1-\mu\lambda_n|\} \\ &= \max\{\mu\lambda_0 - 1, 1 - \mu\lambda_n\} \\ &= \begin{cases} 1 - \mu\lambda_n, & \dfrac{1}{\lambda_0} < \mu < \dfrac{2}{\lambda_0 + \lambda_n}, \\ \mu\lambda_0 - 1, & \dfrac{2}{\lambda_0 + \lambda_n} \leq \mu < \dfrac{2}{\lambda_0}. \end{cases}\end{aligned}$$



(a) Step 0.

(b) Step 1.

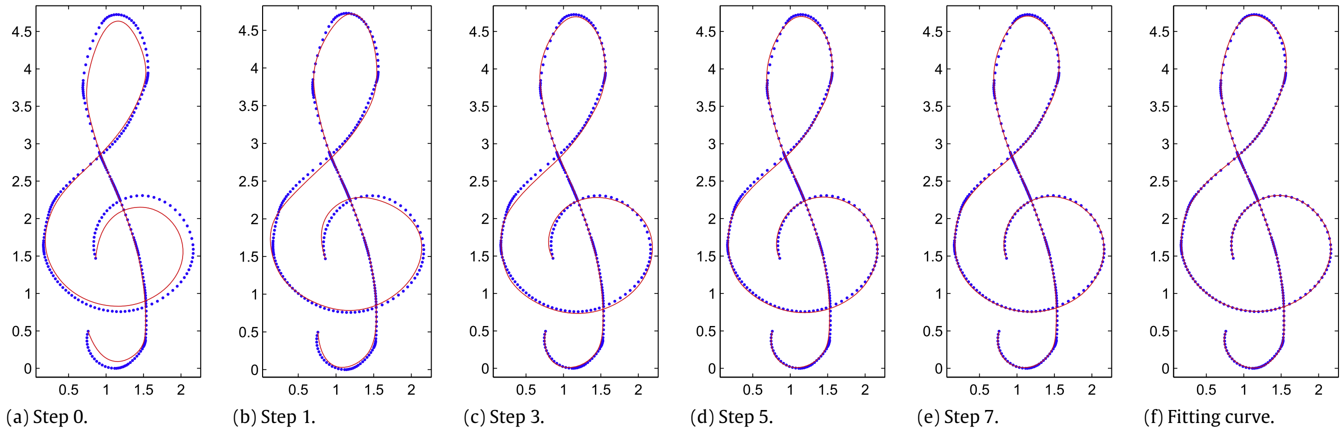
(c) Step 3.

(d) Step 5.

(e) Step 7.

(f) Fitting curve.

Fig. 4. A point set of 201 points sampled from a six-degree Bézier curve with random noise is fitted by a cubic B-spline curve with 20 control points.



(a) Step 0.

(b) Step 1.

(c) Step 3.

(d) Step 5.

(e) Step 7.

(f) Fitting curve.

Fig. 5. A point set of 305 points is fitted by a cubic B-spline curve with 30 control points.

Obviously, $\rho(\boldsymbol{D})$ will reach the minimum $\frac{\lambda_0-\lambda_n}{\lambda_0+\lambda_n}$ when $\mu = \frac{2}{\lambda_0+\lambda_n}$. This completes the proof. □

3.2. *A practical method for selecting an appropriate weight μ*

Although $\mu_{tb} = \frac{2}{\lambda_0+\lambda_n}$ provides the fastest convergence rate (Theorem 3.1) in theory, it needs a large amount of computation to calculate the greatest and smallest eigenvalues. To avoid the computation of the eigenvalues, we propose a simple method for determining the weight μ.

For the $(m+1)\times(n+1)$ matrix $\boldsymbol{A}$ defined in (2), let $\boldsymbol{A}^T\boldsymbol{A} = \{a_{i,j}\}_{0,0}^{m,n}$, where $a_{i,j} = \sum_{k=0}^{m} B_i(t_k)B_j(t_k)$. Together with $\sum_{i=0}^{n} B_i(t_k) = 1$, we have,

$$\sum_{j=0}^{n} a_{i,j} = \sum_{j=0}^{n}\left[\sum_{k=0}^{m} B_i(t_k)B_j(t_k)\right]$$
$$= \sum_{k=0}^{m} B_i(t_k)\left[\sum_{j=0}^{n} B_j(t_k)\right] = \sum_{k=0}^{m} B_i(t_k) := c_i.$$

It means that c_i is the sum of the ith row elements of $\boldsymbol{A}^T\boldsymbol{A}$. Therefore, $\lambda_0 \le \max_i\{c_i\} := C$ $(i = 0, 1, \ldots, n)$, and, $\frac{2}{C} < \frac{2}{\lambda_0}$. Then we define the following weight μ for practical applications:

$$\mu = \frac{2}{C}. \tag{20}$$

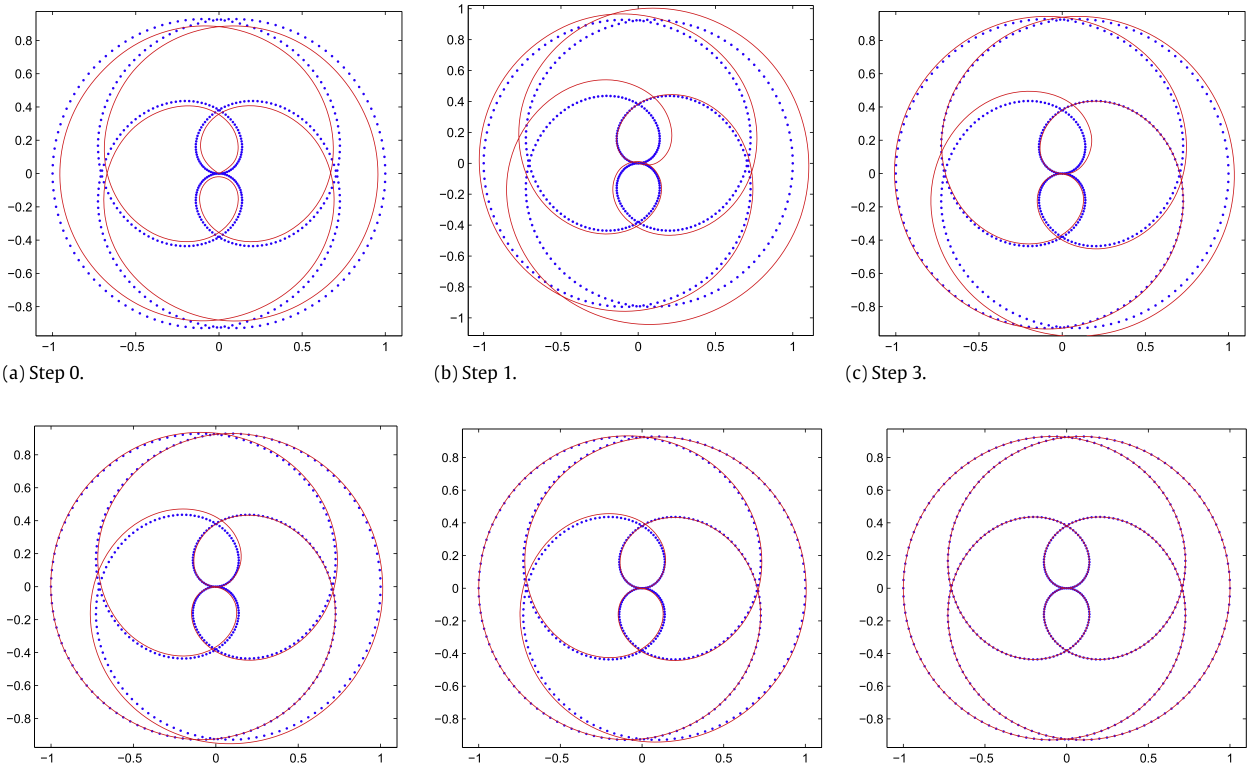
Let $D_k = \sum_{i=0}^{n} \|\boldsymbol{P}_i^k - \boldsymbol{P}_i^\infty\|$, we plot D_k $(k = 1, 2, \ldots, 30)$ for six examples presented in Section 4.2 and two weights $\mu = \frac{2}{C}$ and $\mu_{tb} = \frac{2}{\lambda_0+\lambda_n}$ in Fig. 2. Fig. 2 show that the convergence rates of the two iterative methods with the two weights $\mu = \frac{2}{C}$ and $\mu_{tb} = \frac{2}{\lambda_0+\lambda_n}$ are similar.

4. Implementation and examples

In this section, we test the LSPIA by cubic B-spline curves because of its simplicity and wide range of applications in CAD. In Section 4.1, the implementation details for the LSPIA and cubic B-spline curves are elucidated. In the following, six representative examples are demonstrated in Section 4.2.

4.1. *Implementation*

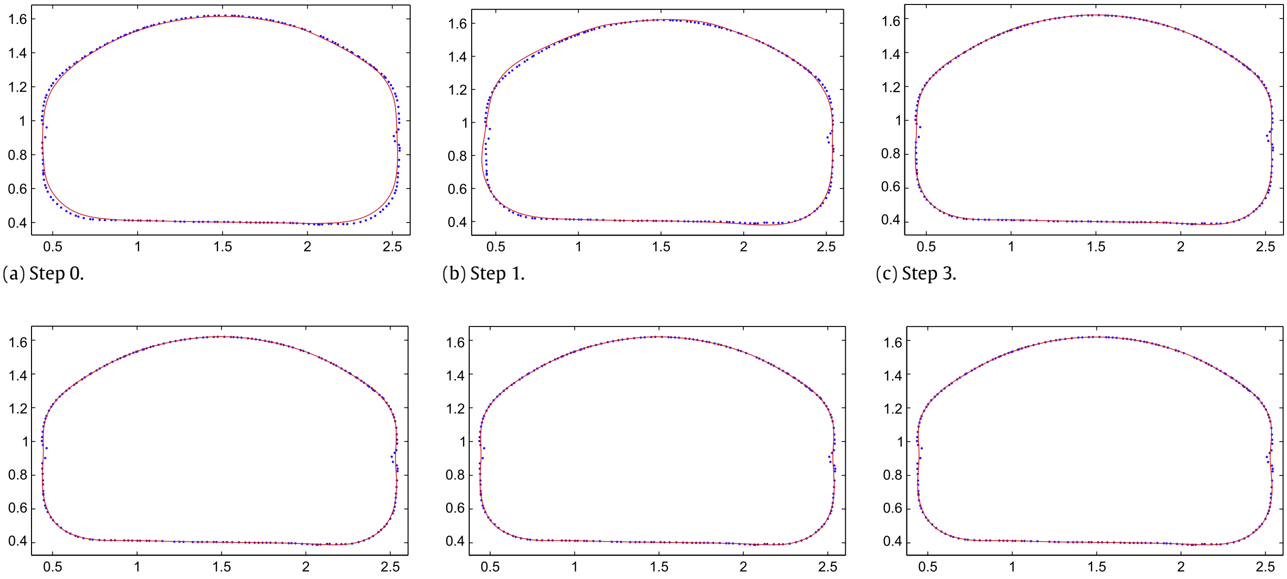
Given an ordered point set $\{\boldsymbol{Q}_i\}_{i=0}^{m}$, we assign the parameters $\{t_i\}_{i=0}^{m}$ for $\{\boldsymbol{Q}_i\}_{i=0}^{m}$ using the normalized accumulated chord



(a) Step 0. (b) Step 1. (c) Step 3.

(d) Step 5. (e) Step 7. (f) Fitting curve.

Fig. 6. A point set of 501 points is fitted by a cubic B-spline curve with 50 control points.



(a) Step 0. (b) Step 1. (c) Step 3.

(d) Step 5. (e) Step 7. (f) Fitting curve.

Fig. 7. A point set of 205 points is fitted by a cubic B-spline curve with 30 control points.

parameterization method, that is [18],

$$t_0 = 0, \qquad t_m = 1,$$
$$t_i = t_{i-1} + \frac{\|\mathbf{Q}_i - \mathbf{Q}_{i-1}\|}{D} \quad (i = 1, 2, \ldots, m-1), \tag{21}$$

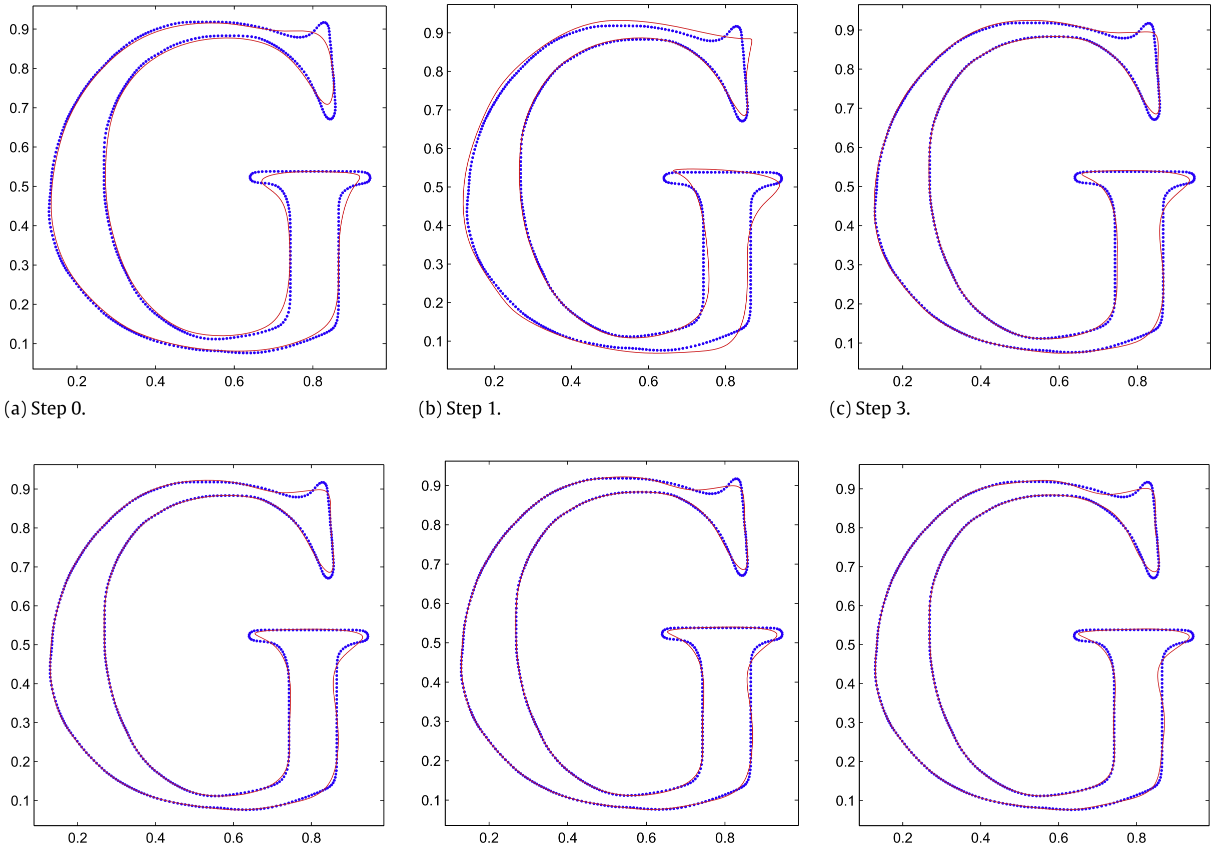
where $D = \sum_{i=0}^{m} \|\mathbf{Q}_i - \mathbf{Q}_{i-1}\|$ is the total chord length.

Moreover, the knot vector for the cubic B-spline fitting curve $\boldsymbol{P}(t) = \sum_{i=0}^{n} N_{i,3}(t)\boldsymbol{P}_i$, is defined as $\{0, 0, 0, 0, \bar{t}_4, \bar{t}_5, \ldots, \bar{t}_n, 1, 1, 1, 1\}$, where,

$$\bar{t}_{j+3} = (1-\alpha)t_{i-1} + \alpha t_i, \quad j = 1, \ldots, n-3,$$
$$i = [jd], \qquad \alpha = jd - i, \quad \text{and} \quad d = \frac{m+1}{n-2}. \tag{22}$$

Here, $[jd]$ is the largest integer not larger than jd [18].

Finally, though the LSPIA method can be started with arbitrary initial control points, an appropriate selection of initial control points makes the LSPIA converge quickly. In our implementation,



(a) Step 0. (b) Step 1. (c) Step 3.

(d) Step 5. (e) Step 7. (f) Fitting curve.

Fig. 8. A point set of 577 points is fitted by a cubic B-spline curve with 50 control points.

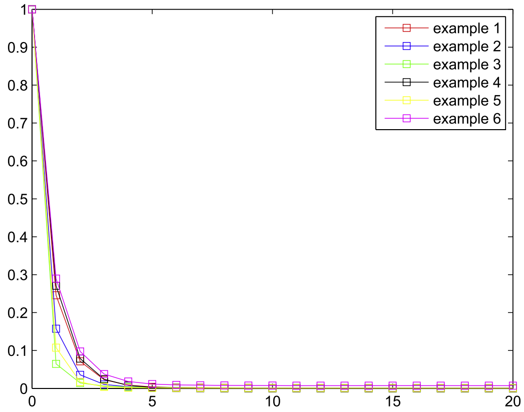


Fig. 9. E_k for the six examples (scaled such that $E_0 = 1$).

we select the initial control points $\{\boldsymbol{P}_i\}_{i=0}^{n}$ as

$$\begin{aligned} &\boldsymbol{P}_0 = \boldsymbol{Q}_0, \qquad \boldsymbol{P}_n = \boldsymbol{Q}_m \\ &\boldsymbol{P}_i = \boldsymbol{Q}_{f(i)}, \quad (i = 1, 2, \ldots, n-1) \end{aligned} \tag{23}$$

where $f(i) = \left[\frac{(m+1)i}{n}\right]$.

4.2. Examples

In this section, we present six representative examples to demonstrate the efficiency and validity of the LSPIA presented in Section 2. The point sets in the six examples are, respectively:

- Example 1: 205 points measured and smoothed from an airfoil-shape data;

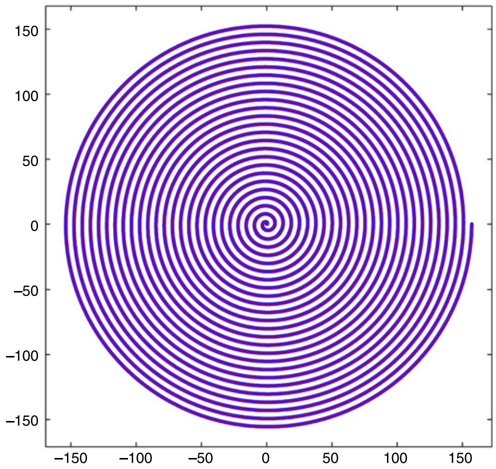


Fig. 10. Fitting sampled 100 001 points from Archimedes spiral $\rho = \theta$ $(0 \le \theta \le 40\pi)$.

- Example 2: 201 points sampled from a six-degree Bézier curve with random noise;
- Example 3: 305 points derived from a subdivision curve generated by incenter subdivision scheme [19];
- Example 4: 501 points sampled uniformly from an analytic curve whose polar coordinate equation is $r = \sin\frac{\theta}{4} (\theta \in [0, 8\pi])$;
- Example 5: 205 points with features measured and smoothed from a cross-section of a mouse;
- Example 6: 577 points with features measured and smoothed from a G-shape font.

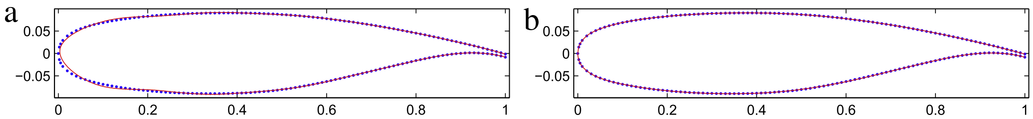


Fig. 11. Fitting curves for the data set presented in example 1 using LSPIA and incremental method. (a) Limit curve for LSPIA; (b) limit curve for incremental LSPIA.

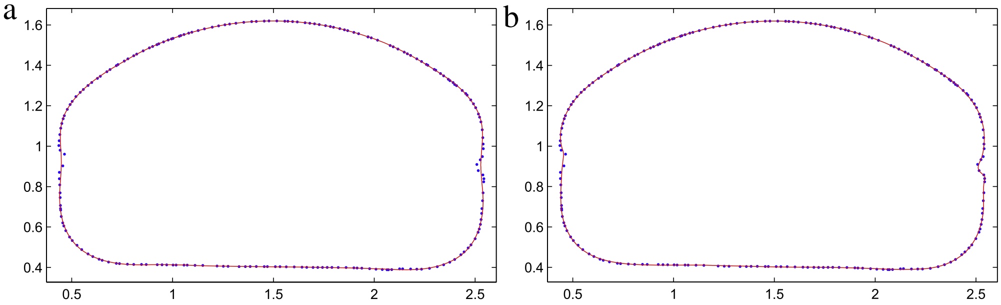


Fig. 12. Fitting curves for the data set presented in example 5 using LSPIA and incremental method. (a) Limit curve for LSPIA; (b) limit curve for incremental LSPIA.

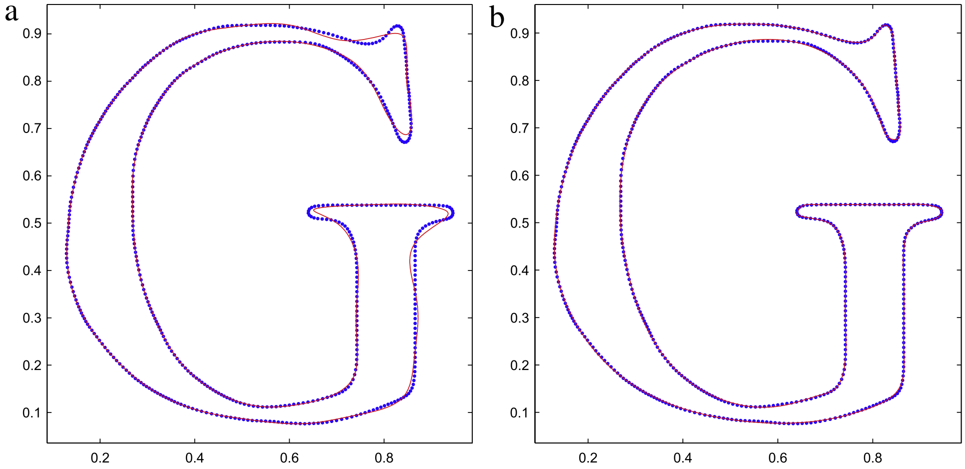


Fig. 13. Fitting curves for the data set presented in example 6 using LSPIA and incremental method. (a) Limit curve for LSPIA; (b) limit curve for incremental LSPIA.

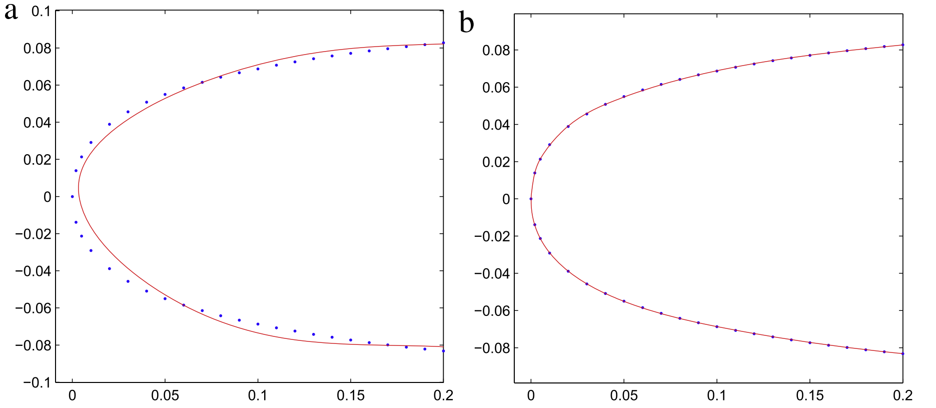


Fig. 14. Zoom in for a detail view of examples illustrated in Fig. 11. (a) Limit curve for LSPIA curve; (b) limit curve for incremental LSPIA.

Table 1
E_k ($k = 0, 1, \ldots, 8$) and E_∞ for examples 1–6.

	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_∞
Em 1	0.743594	0.182764	0.053274	0.017004	0.006330	0.003022	0.001921	0.001513	0.001022
Em 2	257.8242	40.61075	9.274884	2.713619	1.088914	0.607258	0.433126	0.357284	0.252452
Em 3	34.72645	2.248036	0.506507	0.217729	0.125941	0.084237	0.061761	0.048410	0.015905
Em 4	22.36867	6.055901	1.760404	0.543135	0.176884	0.060525	0.021700	0.008167	0.000113
Em 5	2.612842	0.281235	0.044178	0.010435	0.004671	0.003335	0.002876	0.002659	0.002214
Em 6	1.896372	0.549350	0.184882	0.071486	0.034600	0.022081	0.017575	0.015799	0.013494

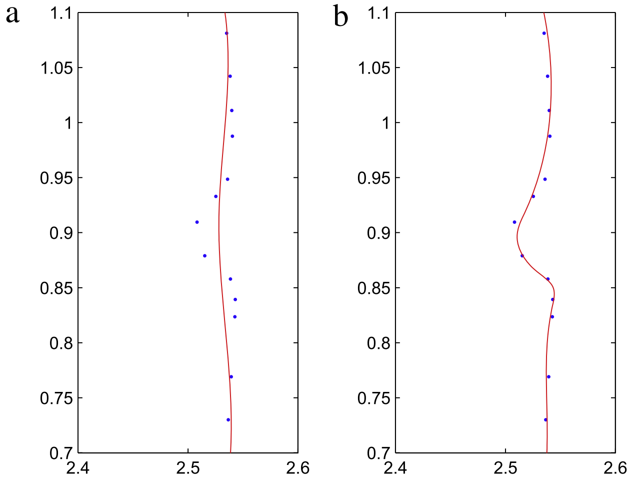


Fig. 15. Zoom in for a detailed view of examples illustrated in Fig. 12. (a) Limit curve for LSPIA curve; (b) limit curve for incremental LSPIA.

With LSPIA, the above six point sets are fitted using cubic B-spline curves with 20, 20, 30, 50, 30, 50 control points, respectively. The point sets and fitting curves are plotted in Figs. 3–8, where the point sets and initial cubic B-spline curves are in (a), the cubic B-spline curves after 1, 3, 5, 7 iterative steps are in (b)–(e), and the limit fitting curves are presented in (f). From Figs. 3–8, we can see that, LSPIA constructs a series of curves which approximate the given data points progressively.

Moreover, we list the fitting errors

$$E_k = \sum_{j=0}^{m} \left\| \boldsymbol{Q}_j - \sum_{i=0}^{n} B_i(t_j)\boldsymbol{P}_i^k \right\|^2, \tag{24}$$

after iteration step $k = 0, 1, \ldots, 8$ and E_∞ in Table 1, and plot $E_k, k = 0, 1, \ldots, 20$ in Fig. 9, respectively. For each example, if $|E_{k+1} - E_k| < 10^{-7}$ we stop the iteration process and let the last E_k be E_∞. From Table 1 and Fig. 9, we can see that all the E_k decrease very fast in some starting steps, and $\{E_k\}$ is a rigorous monotone decreasing sequence.

5. Discussions

5.1. Fitting data sets of very large size

Special techniques are usually required in fitting a large size of data set, because it usually leads to an ill-conditioned or even singular matrix [20]. However, LSPIA can fit a large size of data set successfully. For example, by LSPIA, the 100001 points sampled from Archimedes spiral $\rho = \theta$ $(0 \le \theta \le 40\pi)$ can be fitted by a cubic B-spline with 1000 control points (see Fig. 10). On the contrary, this problem cannot be solved by Matlab using usual least square fitting method because the collocation matrix (saved as sparse matrix) is out of memory.

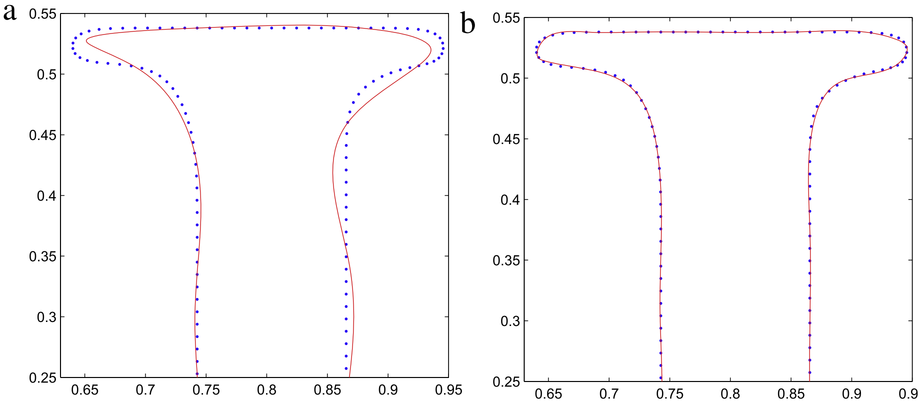
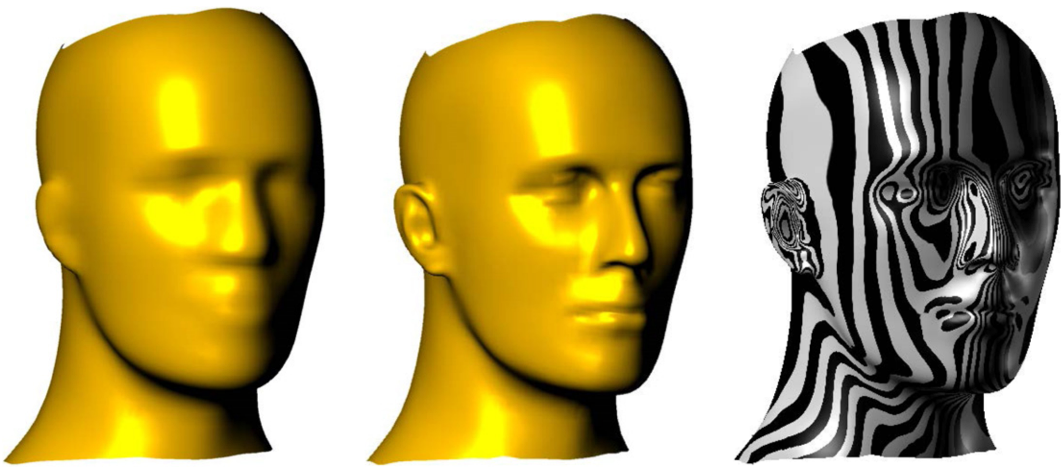


Fig. 16. Zoom in for a detailed view of examples illustrated in Fig. 13. (a) Limit curve for LSPIA curve; (b) limit curve for incremental LSPIA.



(a) Initial surface.

(b) Final surface.

(c) Zebra of limit surface.

Fig. 17. A data point grid with 121 × 161 data points is fitted by a cubic B-spline surface with the incremental data fitting method.

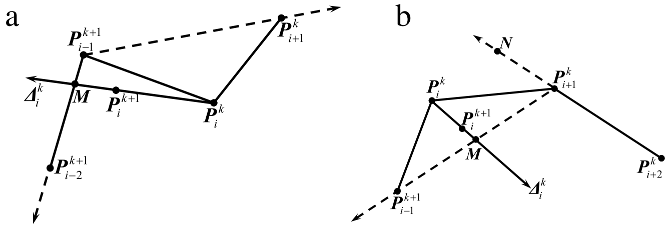


Fig. 18. Adjusting $\{\boldsymbol{P}_i^k\}$ according to shape preserving constraint.

5.2. Adjusting knot vector in iterations of LSPIA

The knot vector plays an important role in data fitting. In practice, the data points are usually fitted *incrementally*, i.e., if the fitting error is not satisfied the given tolerance, we should increase the number of control points, as well as that of knots, to improve the fitting precision. In such an *incremental* data fitting method, if the traditional LSF is employed, the last fitting result is discarded entirely and the fitting procedure should be restarted from scratch. However, using LSPIA, after adding control points and knots, the new fitting procedure can start from the last fitting result, thus saving a great amount of computation.

Specifically, if the fitting error after a round of iterations does not meet the prescribed precision, the incremental data fitting algorithm inserts a knot to an *admissible knot interval* with maximum fitting error, and restarts a new round of iterations from the fitting result of the last round of iterations. Here, a knot interval is called an *admissible* knot interval if there is at least two data points whose parameters lie in the knot interval, and the fitting error of a knot interval $[\bar{t}_i, \bar{t}_{i+1}]$ is defined as $d_i = \sum_{t_j \in [\bar{t}_i, \bar{t}_{i+1}]} \|\boldsymbol{Q}_j - \boldsymbol{P}(t_j)\|$.

Actually, if the selected admissible knot interval has only two data points $\boldsymbol{Q}_j, \boldsymbol{Q}_{j+1}$ with parameters t_j, t_{j+1}, we insert a knot $\bar{t} = \frac{1}{2}(t_j + t_{j+1})$; if the selected admissible knot interval has more than two data points $\boldsymbol{Q}_j, \ldots, \boldsymbol{Q}_{j+a}$ with parameters $t_j, \ldots, t_{j+a}$, the inserted knot is $\bar{t} = \frac{1}{2}(t_l + t_{l+1})$, where $l \in [j+1, j+a-1]$ and

$$\sum_{t_h \in [t_j, \bar{t}_{j+l}]} \|\boldsymbol{Q}_h - \boldsymbol{P}(t_h)\| \geq \frac{d_i}{2},$$

$$\sum_{t_h \in [t_l, \bar{t}_{j+a}]} \|\boldsymbol{Q}_h - \boldsymbol{P}(t_h)\| \geq \frac{d_i}{2}.$$

Table 2
Statistical data for LSPIA and EPIA.

Methods	Initial patch		Final patch		Iteration times
	#Control mesh	Error	#Control mesh	Error	
LSPIA	31 × 66	0.053347	63 × 74	0.002580	50
EPIA [3]	31 × 66	0.053347	68 × 90	0.002576	100

This incremental LSPIA algorithm stops when the number of control points is equal to a predefined number or the fitting error satisfies the predefined precision.

We test the incremental data fitting aforementioned on Examples 1, 5, 6 presented in Section 4.2, and illustrated them in Figs. 11 and 13. From them, we can see that with this incremental LSPIA algorithm, the fitting curves faithfully resemble the shape implied by the data points. In the curve segment whose curvature varies slowly, the incremental LSPIA method distributes sparse knots automatically, while in the curve segment whose curvature varies quickly, the incremental LSPIA method distributes many knots automatically to capture the details implied by the data points. To view this effect clearly, we zoom in some details of Examples 1, 6 to compare the fitting effect of LSPIA and LSPIA with knot adjusting (see Figs. 14–16). In each example in Figs. 11 and 13, respectively, we use the same number of control points for two different LSPIA methods.

We also test the incremental data fitting on a data point grid with 121 × 161 data points and illustrated them in Fig. 17. These figures show that the final surface catches the features accurately. We compare incremental LSPIA with EPIA [3] and summarize the statistical data in Table 2. From Table 2 we find that to achieve the similar fitting error from the same initial patch, the incremental LSPIA algorithm needs fewer control points and iteration times, thus performing better than EPIA.

5.3. Shape preserving fitting by LSPIA

In general, shape preserving means that the inflexion of the fitting curve is the same as that implied by the data point sequence. By LSPIA, it is easy to generate a fitting curve with shape preserving property.

First, from the given data point, we select a sequence of initial control points, which correspond to an initial B-spline curve with the desirable shape. In the iterations of LSPIA, we make the

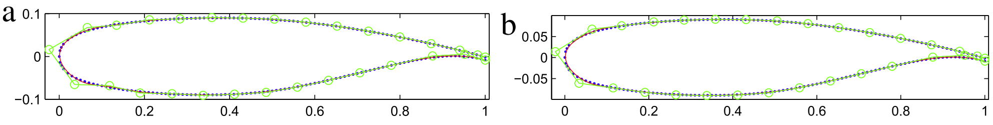


Fig. 19. Fit 205 points of example 1 with a cubic B-spline with 30 control points. (a) Least square fitting, (b) shape preserving least square fitting.

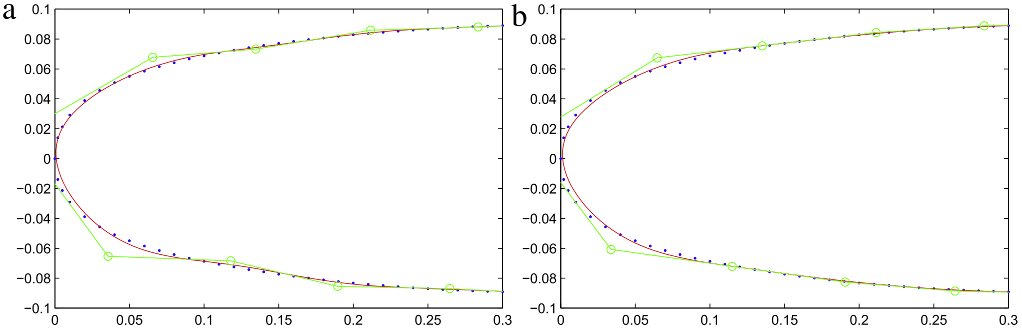
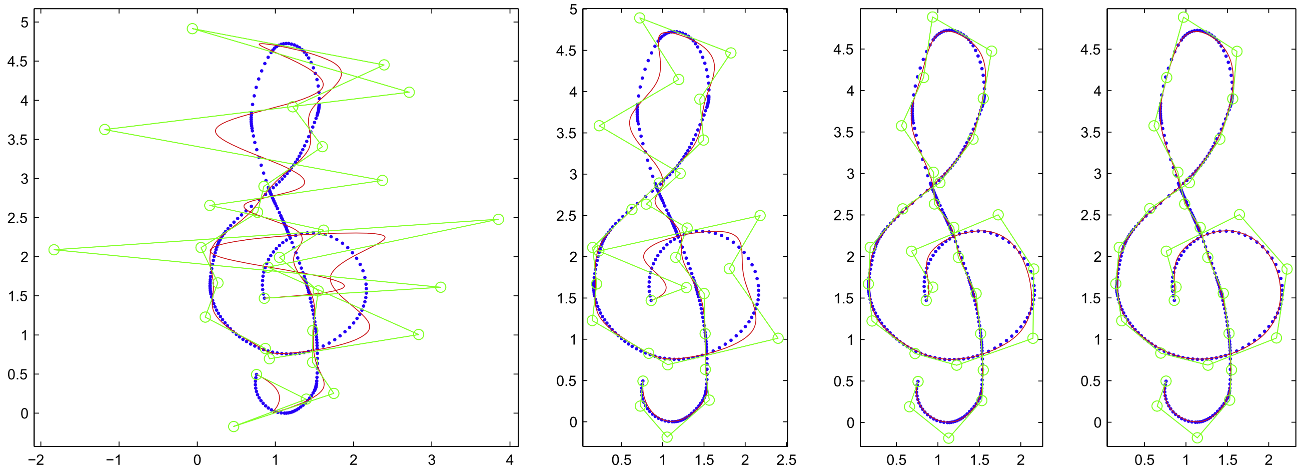


Fig. 20. Zoom in for a detail view of the examples illustrated in Fig. 19. (a) Least square fitting, (b) shape preserving least square fitting.



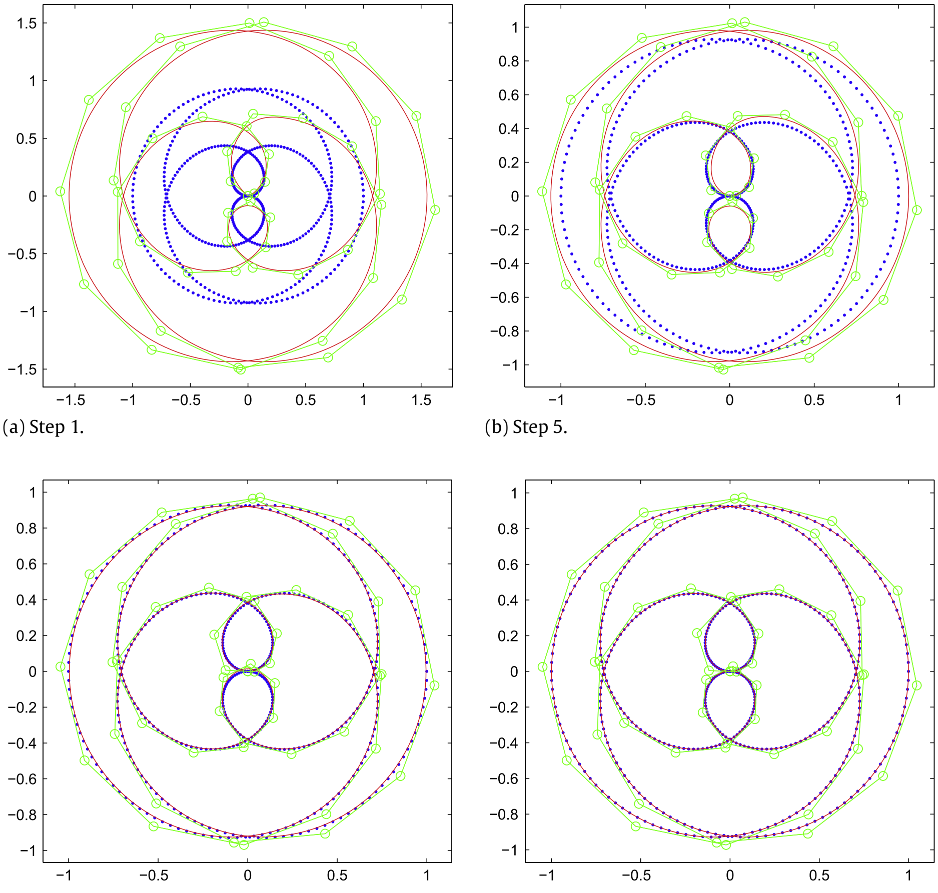
(a) Step 20.

(b) Step 40.

(c) Step 60.

(d) Step 70.

Fig. 21. LSPIA for the data of example 3 with initial control points set as: the two end control points are the same as the two end points of the input points, and the other control points are (100, 1).



(a) Step 1.

(b) Step 5.

(c) Step 10.

(d) Step 20.

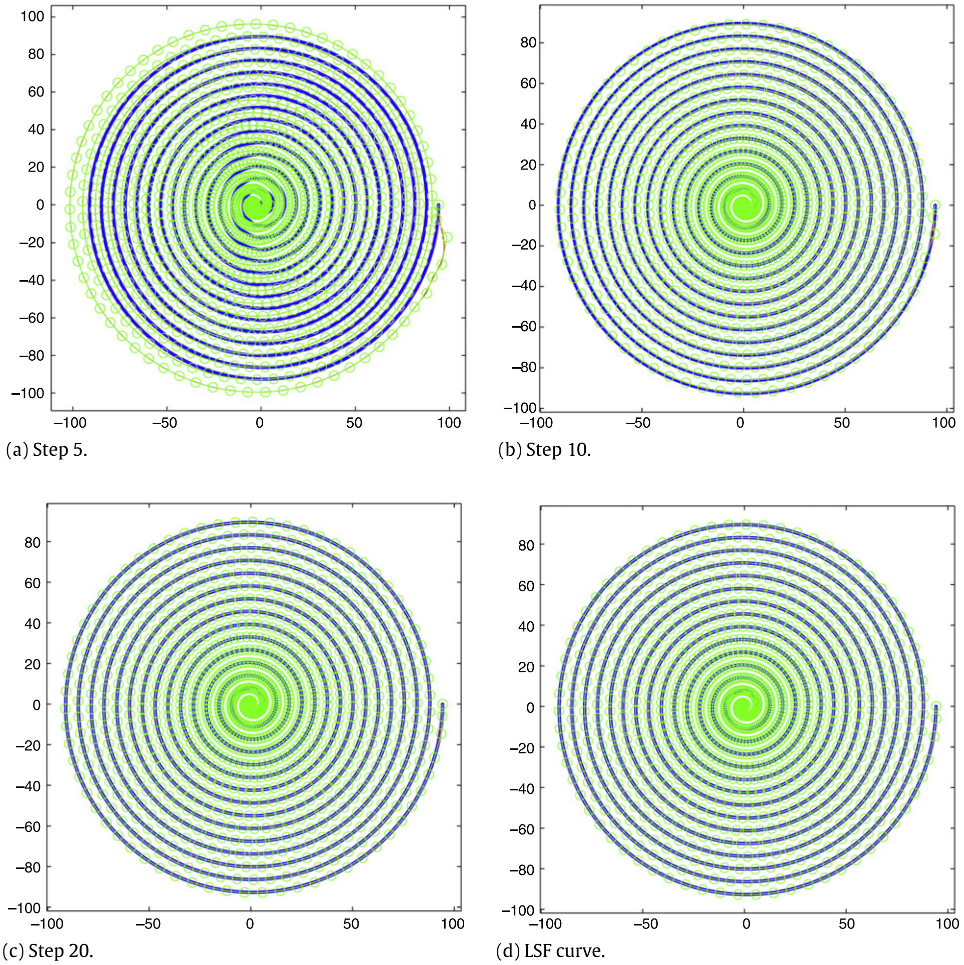
Fig. 22. LSPIA for the data of example 4 with all initial control points set as the centroid of the input points.

inflexions of the new control polygon $\{\boldsymbol{P}_i^{k+1}\}_{i=0}^n$ the same as those of the old control polygon $\{\boldsymbol{P}_i^k\}_{i=0}^n$, so the inflexions of the limit curve are the same as those of the initial curve, and then the limit fitting curve has the shape preserving property. For this purpose, we develop the following method:

1. $\{\boldsymbol{P}_i^k\}_{i=0}^n$ are updated from $i = 1$ to $i = n - 1$ one by one;
2. when updating $\boldsymbol{P}_i^k$ to $\boldsymbol{P}_{i+1}^k$, we consider two cases:

(a) as to the four consecutive points $\boldsymbol{P}_{i-2}^{k+1}, \boldsymbol{P}_{i-1}^{k+1}, \boldsymbol{P}_i^k, \boldsymbol{P}_{i+1}^k$, if the line segment with end points $\boldsymbol{P}_i^k$ and $\boldsymbol{P}_i^k + \boldsymbol{\Delta}_i^k$ has an intersection with radial line $\boldsymbol{P}_{i-1}^{k+1}\boldsymbol{P}_{i+1}^k$ or $\boldsymbol{P}_{i-1}^{k+1}\boldsymbol{P}_{i-2}^{k+1}$, we set $\boldsymbol{P}_i^{k+1} = \boldsymbol{P}_i^k + 0.8(\boldsymbol{M} - \boldsymbol{P}_i^k)$, where $\boldsymbol{M}$ is the intersection closer to $\boldsymbol{P}_i^k$ (see Fig. 18(a));

(b) as to the four consecutive points $\boldsymbol{P}_{i-1}^{k+1}, \boldsymbol{P}_i^k, \boldsymbol{P}_{i+1}^k \boldsymbol{P}_{i+2}^k$, if the line segment with end points $\boldsymbol{P}_i^k$ and $\boldsymbol{P}_i^k + \boldsymbol{\Delta}_i^k$ has



(a) Step 5.

(b) Step 10.

(c) Step 20.

(d) LSF curve.

Fig. 23. Fitting sampled 100 001 points from Archimedes spiral $\rho = \theta$ $(0 \le \theta \le 40\pi)$ with initial control points set as: the two end control points are the same as the two end points of the input points, and the other control points are the midpoint of the two end points.

intersection with radial line $\boldsymbol{P}^k_{i+1}\boldsymbol{P}^{k+1}_{i-1}$ or $\boldsymbol{P}^k_{i+1}\boldsymbol{N}$ ($\boldsymbol{N}$ is on the reverse extension of $\boldsymbol{P}^k_{i+2}\boldsymbol{P}^k_{i+1}$), we set $\boldsymbol{P}^{k+1}_i = \boldsymbol{P}^k_i + 0.8(\boldsymbol{M} - \boldsymbol{P}^k_i)$, where $\boldsymbol{M}$ is the intersection closer to $\boldsymbol{P}^k_i$ (see Fig. 18(b));

3. for $\boldsymbol{P}^k_2$ or $\boldsymbol{P}^k_{n-1}$, only $\boldsymbol{P}^{k+1}_0, \boldsymbol{P}^{k+1}_1, \boldsymbol{P}^k_2, \boldsymbol{P}^k_3$ or $\boldsymbol{P}^{k+1}_{n-3}, \boldsymbol{P}^{k+1}_{n-2}, \boldsymbol{P}^k_{n-1}, \boldsymbol{P}^k_n$ are considered.

We present an example to demonstrate the effect of shape preserving fitting by the above method. The outcomes are presented in Fig. 19 and their close-up views in Fig. 20.

5.4. Selection of the initial control points

In Remark 2.1 we point out that the initial points $\{\boldsymbol{P}^0_i\}^n_{i=0}$ can be set arbitrary. By (9) we know that the smaller the $\|\boldsymbol{P}^0 - (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q}\|$, the closer the distance between $\boldsymbol{P}_{k+1}$ and the limit $(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{Q}$. So an appropriate selection of the initial control points $\{\boldsymbol{P}^0_i\}^n_{i=0}$ will reduce the iteration times to achieve the same error. In this subsection, we present some examples to demonstrate this effect.

First we use LSPIA to fit the data in Example 3 whose bound box is about $0 \le x \le 2$, $0 \le y \le 4.7$. The two end points of the initial control points are set as the two end points of the input points, all other initial control points are set as $(100, 1)$, a point far from the input points. The results are presented in Fig. 21. From these figures we can see that after 20 iterative steps the fitting curve is not well-formed. But after more iterative steps, for example, 70 iterative steps, the distance between the fitting curve and the input points are very small. Compared with the fitting curves presented in Fig. 5, whose initial control points are selected appropriately, we find that more iteration steps are needed if the initial control points are far from the limit curves.

We also use LSPIA to fit the data in Example 4 with the initial control points set as the centroid of the input points, and the data sampled from Archimedes spiral $\rho = \theta$ $(0 \le \theta \le 40\pi)$ with the two end points set as those of the input points and other initial control points are the midpoints of the two end points. The results are presented in Figs. 22 and 23, respectively. From these figures we can see that the fitting curves converge faster than those of Fig. 21, but slower than those of Fig. 6 because the distances between the initial control points and the corresponding limit control points fall in between the two cases.

From these examples we conclude that an appropriate selection of the initial control points is important. However, as a shortcoming of LSPIA, the construction of a desirable initial patch is difficult, and the convergence of LSPIA will be made relatively slow by the undesirable initial patch.

6. Conclusion

In this paper, a new progressive and iterative approximation method is developed, whose limit is the least square fitting result of the given data points (this method is abbreviated as LSPIA). Similar as the classical PIA method, LSPIA also constructs a series of fitting curves (surfaces) by adjusting the control points iteratively. In each iteration, the difference vector for each control point is a weighted sum of some difference vectors between the data points and their corresponding points on the fitting curve (surface). Since it is able to not only fit the data set of very large size efficiently and robustly, but also reuse the last fitting result in an incremental data fitting procedure, it will have a wide range of applications in geometric design.

Acknowledgments

This work is supported by NSFC (Nos. 61003194, 60970150, 60933008, 61370166, 61379072), and the Open Project Program of the State Key Lab of CAD&CG (Grant No. A1304), Zhejiang University.

References

[1] Lin H, Wang G, Dong C. Constructing iterative non-uniform B-spline curve and surface to fit data points. Science in China Series F 2004;47:315–31.
[2] Lin H, Bao H, Wang G. Totally positive bases and progressive iteration approximation. Computers and Mathematics with Applications 2005;50:575–86.
[3] Lin H, Zhang Z. An extended iterative format for the progressive-iteration approximation. Computers & Graphics 2011;35:967–75.
[4] Qi D, Tian Z, Zhang Y, Feng J. The method of numeric polish in curve fitting. Acta Mathematica Sinica 1975;18:173–84.
[5] de Boor C. How does Agee's smoothing method work? Proceedings of the 1979 army numerical analysis and computers conference, ARO report 79-3. Army Research Office; 1979. p. 299–302.
[6] Shi L, Wang R. An iterative algorithm of nurbs interpolation and approximation. Journal of Mathematical Research and Exposition 2006;26:735–43.
[7] Lu L. Weighted progressive iteration approximation and convergence analysis. Computer Aided Geometric Design 2010;27:129–37.
[8] Lin H. Local progressive-iterative approximation format for blending curves and patches. Computer Aided Geometric Design 2010;27:322–39.
[9] Chen J, Wang GJ. Progressive-iterative approximation for triangular Bézier surfaces. Computer-Aided Design 2011;43(8):889–95.
[10] Maekawa T, Matsumoto Y, Namiki K. Interpolation by geometric algorithm. Computer-Aided Design 2007;39:313–23.
[11] Gofuku S, Tamura S, Maekawa T. Point-tangent/point-normal B-spline curve interpolation by geometric algorithms. Computer-Aided Design 2009;41:412–22.
[12] Nishiyama Y, Morioka M, Maekawa T. Loop subdivision surface fitting by geometric algorithms. In: Igarashi T, Max N, Sillion F, editors. Poster proceedings of pacific graphics 2008.
[13] Kineri Y, Wang M, Lin H, Maekawa T. B-spline surface fitting by iterative geometric interpolation/approximation algorithms. Computer-Aided Design 2012;44(7):697–708.
[14] Cheng F, Fan F, Lai S, Huang C, Wang J, Yong J. Loop subdivision surface based progressive interpolation. Journal of Computer Science and Technology 2009;24:39–46.
[15] Fan F, Cheng F, Lai S. Subdivision based interpolation with shape control. Computer-Aided Design and Applications 2008;5:539–47.
[16] Chen Z, Luo X, Tan L, Ye B, Chen J. Progressive interpolation based on Catmull–Clark subdivision surfaces. Computer Graphics Forum 2008;27:1823–7. Pacific Graphic 2008.
[17] Deng C, Ma W. Weighted progressive interpolation of Loop subdivision surfaces. Computer-Aided Design 2012;44:424–31.
[18] Piegl L, Tiller W. The NURBS book. 2nd ed. New York, USA: Springer-Verlag; 1997.
[19] Deng C, Wang G. Incenter subdivision scheme for curve interpolation. Computer Aided Geometric Design 2010;27:48–59.
[20] Pereyra V, Scherer G. Large scale least squares scattered data fitting. Applied Numerical Mathematics 2003;44:225–39.