AIMS Mathematics, 11(8): 24808–24838.
DOI: 10.3934/math.2026998
Received: 09 June 2026
Revised: 29 July 2026
Accepted: 11 August 2026
Published: 13 August 2026

https://www.aimspress.com/journal/Math

Research article

Spectral adaptivity for iterative least-squares B-spline curve and surface fitting

Xingxuan Peng* **and Yutong Li**

School of Mathematics, Liaoning Normal University, Dalian 116081, China

* **Correspondence:** Email: pengxx@lnnu.edu.cn.

Abstract: For curve and surface fitting, the least-squares progressive iterative approximation (LSPIA) method updates the control points with a constant weight. To accelerate the iteration, this paper proposes a spectrally adaptive LSPIA (denoted by SaLSPIA) method. SaLSPIA updates the control points by an adaptive weight constructed from the adjusting vectors of two consecutive iterations, so that the weight adapts to the spectral scale of the collocation matrix, while retaining the simple control-point update form of LSPIA. We establish the global convergence and an R-linear convergence rate of SaLSPIA for both the full-rank and the rank-deficient cases, including convergence to a least-squares fitting curve or surface when the corresponding fitting system is rank deficient. Numerical experiments on curve, surface, and missing-data fitting show that SaLSPIA is efficient and robust.

Keywords: least-squares fitting; progressive iterative approximation; B-spline; spectral adaptivity; curve and surface reconstruction; missing data fitting
Mathematics Subject Classification: 65D07, 65D17, 65D10, 65F10, 65F20

1. Introduction

Data fitting is a fundamental problem in computer-aided geometric design [1, 2], image processing [3], and 3D reconstruction [4]. In computer graphics and visual computing, fitting a B-spline curve or surface to large point sets underlies reverse engineering, surface reconstruction, and light detection and ranging (LiDAR) terrain modeling, for which least-squares fitting provides a simple and practical formulation. Since such data are frequently large-scale or incomplete, iterative fitting schemes with rapid and stable convergence are important.

The least-squares progressive iterative approximation (LSPIA) was proposed by Deng and Lin [5] as an extension of progressive iterative approximation (PIA) to the least-squares setting. The iterative scheme underlying PIA can be traced to the method of numeric polish for curve fitting proposed by Qi et al. [6] and independently rediscovered by de Boor [7]. Lin et al. [8] established its

convergence for non-uniform cubic B-spline curves and surfaces. In subsequent work, Lin et al. [9] extended the convergence analysis to curves and surfaces represented by normalized totally positive bases. The convergence of LSPIA in the rank-deficient case was further studied in [10]. LSPIA is characterized by its simple iterative scheme, stable convergence, and suitability for large-scale fitting where direct solvers may become numerically unstable or memory-prohibitive. It has been extended to T-splines [11], subdivision surfaces [12, 13], and constrained fitting [14]. Related developments include generalized B-spline fitting [15], a survey of geometric iterative methods [16], regularized bivariate B-spline surface fitting [17], and preconditioned fitting for triangular Bézier patches [18].

Many accelerated variants of LSPIA have been proposed. Splitting and inverse-approximation schemes include the composite LSPIA based on the Schulz iteration [19], the hyperpower LSPIA [20], and the Hermitian and skew-Hermitian splitting LSPIA [21]. Semi-iterative and conjugate-gradient methods include the asynchronous LSPIA based on the Chebyshev semi-iterative method [22] and the conjugate-gradient progressive-iterative approximation [23]. Memory and momentum variants include the LSPIA with memory [24] and the LSPIA with Polyak momentum (PmLSPIA) and the LSPIA with Nesterov momentum (NmLSPIA) [25]. Local and randomized update strategies have also been developed [26, 27]. Rios and Jüttler [28] proved that LSPIA is equivalent to a gradient descent method for the least-squares fitting objective and proposed a stochastic variant of LSPIA. The convergence of gradient methods is well known to be sensitive to the choice of step size. The Barzilai–Borwein (BB) step size [29] is one of the most representative choices. Motivated by the quasi-Newton secant equation, it determines the step size from two consecutive iterates and gradients and often achieves faster practical convergence than the classical steepest descent while keeping a low per-iteration cost. Building on this idea, we develop an adaptive update weight that varies with the iteration while retaining the simple control-point update form of LSPIA.

We call the resulting method the spectrally adaptive LSPIA (SaLSPIA) and use it for B-spline curve and surface fitting. SaLSPIA updates the control points with a dynamic weight computed at each step from two consecutive adjusting vectors. The main contributions are threefold. First, we construct an adaptive LSPIA weight through an iteration-dependent interpolation parameter m_k. Second, we establish global convergence and an R-linear convergence rate in both the full-rank and rank-deficient cases, including tensor-product surface fitting and fitting problems involving incomplete data. Third, experiments on curve, surface, missing-data, and large-scale fitting, together with ablation and sensitivity studies, confirm the efficiency and robustness of SaLSPIA.

The rest of this paper is structured as follows. The algorithm is presented in Section 2. Convergence analyses are derived in Section 3. Numerical experiments are reported in Section 4. Finally, conclusions and limitations are discussed in the last section.

2. The SaLSPIA method

This section presents the SaLSPIA method for B-spline fitting. We first describe the algorithm for curve fitting and then extend the same construction to tensor-product B-spline surface fitting.

2.1. Curve fitting

Given data points $\{Q_j\}_{j=0}^{m} \subset \mathbb{R}^d$ $(m \geq n)$ with parameters $\{t_j\}_{j=0}^{m}$, let $\{B_i(t)\}_{i=0}^{n}$ be the B-spline basis and let $A_{ji} = B_i(t_j),\ j = 0, 1, \ldots, m,\ i = 0, 1, \ldots, n$ be the collocation matrix. At the beginning of

the iteration, we select $\{P_i^{(0)}\}_{i=0}^n$ from $\{Q_j\}_{j=0}^m$ as the control point set and construct the initial B-spline fitting curve

$$C^{(0)}(t) = \sum_{i=0}^{n} B_i(t) P_i^{(0)}, \qquad t \in [t_0, t_m].$$

Set

$$\Lambda_B = \max_{0 \le i \le n} \sum_{j=0}^{m} B_i(t_j), \qquad \alpha_0 = \frac{1}{\Lambda_B}. \tag{2.1}$$

The initial difference vectors are

$$r_j^{(0)} = Q_j - C^{(0)}(t_j), \qquad j = 0, 1, \ldots, m.$$

Define

$$D_i^{(0)} = \sum_{j=0}^{m} B_i(t_j) r_j^{(0)}, \qquad i = 0, 1, \ldots, n.$$

The first control-point update is $P_i^{(1)} = P_i^{(0)} + \alpha_0 D_i^{(0)},\ i = 0, 1, \ldots, n$, and the corresponding B-spline fitting curve is

$$C^{(1)}(t) = \sum_{i=0}^{n} B_i(t) P_i^{(1)}.$$

Suppose that we have obtained the kth B-spline fitting curve

$$C^{(k)}(t) = \sum_{i=0}^{n} B_i(t) P_i^{(k)}, \qquad k \ge 1.$$

The difference vectors are $r_j^{(k)} = Q_j - C^{(k)}(t_j),\ j = 0, 1, \ldots, m$, and we set

$$D_i^{(k)} = \sum_{j=0}^{m} B_i(t_j) r_j^{(k)}, \qquad i = 0, 1, \ldots, n.$$

Then the control points of the $(k+1)$st B-spline fitting curve are given by

$$P_i^{(k+1)} = P_i^{(k)} + \alpha_k D_i^{(k)}, \qquad i = 0, 1, \ldots, n, \tag{2.2}$$

where α_k is the weight to be determined. To determine α_k, we use the information from two successive sets $\{D_i^{(k-1)}\}_{i=0}^n$ and $\{D_i^{(k)}\}_{i=0}^n$. Define

$$\rho_\ell = \sum_{i=0}^{n} \|D_i^{(\ell)}\|^2, \qquad \ell = k-1, k, \tag{2.3}$$

and

$$\zeta_k = \sum_{i=0}^{n} \left\langle D_i^{(k-1)}, D_i^{(k)} \right\rangle. \tag{2.4}$$

Set

$$b_k = \frac{\rho_{k-1} - \zeta_k}{\alpha_{k-1}}, \qquad c_k = \frac{\rho_{k-1} - 2\zeta_k + \rho_k}{\alpha_{k-1}^2}. \tag{2.5}$$

These quantities give two local weights for the next control-point correction

$$\alpha_k^{(1)} = \frac{\rho_{k-1}}{b_k}, \qquad \alpha_k^{(2)} = \frac{b_k}{c_k}. \tag{2.6}$$

Next, define

$$\mu_k = \frac{b_k}{\Lambda_B \rho_{k-1}}, \qquad \widehat{\mu}_k = \min\{\max\{\mu_k, 0\}, 1\}, \tag{2.7}$$

and

$$m_k = \max\{1 - \widehat{\mu}_k, \delta\}, \tag{2.8}$$

where $\delta \in (0, 1)$ is a prescribed constant. The parameter m_k is used to interpolate the two local weights continuously. Let $\widetilde{\alpha}_k$ be the positive root of

$$(1 - m_k)c_k\alpha^2 + (2m_k - 1)b_k\alpha - m_k\rho_{k-1} = 0. \tag{2.9}$$

Equivalently, set $B_q = (2m_k - 1)b_k$, $D_q = B_q^2 + 4m_k(1 - m_k)\rho_{k-1}c_k$. Then the positive root can be written as

$$\widetilde{\alpha}_k = \frac{2m_k\rho_{k-1}}{B_q + \sqrt{D_q}}. \tag{2.10}$$

For numerical implementation, the root in (2.10) is evaluated in a stable form. In the shorthand notation $a = \rho_{k-1}$, $b = b_k$, $c = c_k$, $m = m_k$, we have $A_q = (1 - m)c$, $B_q = (2m - 1)b$, $D_q = B_q^2 + 4A_qma$. If $A_q \le \varepsilon_{\text{saf}}$, where ε_{saf} is the safeguard tolerance used in Algorithm 1, we use the limiting value $\widetilde{\alpha}_k = a/b$. Otherwise, we compute

$$\widetilde{\alpha}_k = \begin{cases} \dfrac{2ma}{B_q + \sqrt{D_q}}, & B_q \ge 0, \\ \dfrac{\sqrt{D_q} - B_q}{2A_q}, & B_q < 0. \end{cases}$$

This is algebraically equivalent to (2.10) but avoids cancellation in the quadratic formula. Then the trial weight is set as

$$\bar{\alpha}_k = \begin{cases} \alpha_k^{(2)}, & \text{if } 1 - \widehat{\mu}_k \le \delta, \\ \widetilde{\alpha}_k, & \text{otherwise.} \end{cases} \tag{2.11}$$

The lower bound δ is only a numerical safeguard. When $1 - \widehat{\mu}_k \le \delta$, the algorithm directly uses the short endpoint $\alpha_k^{(2)}$, so the safeguard does not prevent the short BB-type step. The trial weight is then checked by a nonmonotone fitting-error criterion. Let

$$\Phi_k = \frac{1}{2}\sum_{j=0}^{m} \|r_j^{(k)}\|^2, \qquad F_k = \max_{0 \le q \le \min(k, M-1)} \Phi_{k-q},$$

where $M \ge 1$ is the memory length. For a given weight α, consider the fitting curve generated by the tentative control points $P_i^{(k)} + \alpha D_i^{(k)}$, namely

$$C_\alpha^{(k+1)}(t) = \sum_{i=0}^{n} B_i(t)\left(P_i^{(k)} + \alpha D_i^{(k)}\right).$$

Starting with $\alpha = \bar{\alpha}_k$, we accept $\alpha_k = \alpha$ if

$$\frac{1}{2}\sum_{j=0}^{m}\left\|Q_j - C_{\alpha}^{(k+1)}(t_j)\right\|^2 \le F_k - \sigma\alpha\rho_k \tag{2.12}$$

holds, where $\sigma \in (0, 1/2]$ is a prescribed constant. Otherwise, α is replaced by $\alpha/2$ until (2.12) is satisfied. Once α_k is determined, the adjusting vector for the ith control point is $\Delta_i^{(k)} = \alpha_k D_i^{(k)}$. Then the iteration (2.2) takes the LSPIA form $P_i^{(k+1)} = P_i^{(k)} + \Delta_i^{(k)}$, $i = 0, 1, \ldots, n$, and the $(k+1)$st B-spline fitting curve is

$$C^{(k+1)}(t) = \sum_{i=0}^{n} B_i(t) P_i^{(k+1)}.$$

The procedure is summarized in Algorithm 1.

Algorithm 1 SaLSPIA for B-spline curve fitting.

Require: Initial control points $\{P_i^{(0)}\}_{i=0}^n$; parameters $\sigma \in (0, 1/2]$, $M \ge 1$, and $\delta \in (0, 1)$; convergence tolerance $\varepsilon > 0$; safeguard tolerance $\varepsilon_{\text{saf}} > 0$; maximum iteration count $K_{\max} \ge 2$.

Ensure: Control points $\{P_i\}_{i=0}^n$ of the fitting curve.

Compute Λ_B by (2.1) and set $\alpha_0 \leftarrow 1/\Lambda_B$.
Compute $\{D_i^{(0)}\}$ and set $\rho_0 \leftarrow \sum_{i=0}^n \|D_i^{(0)}\|^2$.
if $\rho_0 \le \varepsilon_{\text{saf}}$ **then**
    return $\{P_i^{(0)}\}_{i=0}^n$
end if
Update $P_i^{(1)} \leftarrow P_i^{(0)} + \alpha_0 D_i^{(0)}$, $i = 0, 1, \ldots, n$.
Compute $C^{(1)}(t)$, $\{r_j^{(1)}\}$, $\{D_i^{(1)}\}$ and set $\rho_1 \leftarrow \sum_{i=0}^n \|D_i^{(1)}\|^2$ and $E_1 \leftarrow \rho_1/\rho_0$.
$k \leftarrow 1$.
while $\rho_k > \varepsilon_{\text{saf}}$ and $E_k > \varepsilon$ and $k < K_{\max}$ **do**
    Compute ζ_k, b_k, and c_k by (2.4) and (2.5).
    if $b_k \le \varepsilon_{\text{saf}}$ or $c_k \le \varepsilon_{\text{saf}}$ **then**
        $\bar{\alpha}_k \leftarrow 1/\Lambda_B$
    else
        Compute $\bar{\alpha}_k$ by (2.6)–(2.11), using the stable evaluation described after (2.10).
    end if
    $\alpha \leftarrow \bar{\alpha}_k$
    while the criterion (2.12) fails **do**
        $\alpha \leftarrow \alpha/2$
    end while
    $\alpha_k \leftarrow \alpha$
    Update $P_i^{(k+1)} \leftarrow P_i^{(k)} + \alpha_k D_i^{(k)}$, $i = 0, 1, \ldots, n$.
    Compute $C^{(k+1)}(t)$, $\{r_j^{(k+1)}\}$, $\{D_i^{(k+1)}\}$ and set $\rho_{k+1} \leftarrow \sum_{i=0}^n \|D_i^{(k+1)}\|^2$ and $E_{k+1} \leftarrow \rho_{k+1}/\rho_0$.
    $k \leftarrow k + 1$
end while
return $\{P_i^{(k)}\}_{i=0}^n$

Here, ε_{saf} prevents division by zero or near-zero quantities. The iteration terminates when the adjusting vector is negligible, and $1/\Lambda_B$ is used when b_k or c_k is too small. The parameter ε is the convergence tolerance for the relative least-squares fitting error $E_k = \rho_k/\rho_0$.

Remark 2.1. *If α_k is held constant in* (2.2)*, the iteration reduces to the classical LSPIA iteration [5]. Figure 1 illustrates the geometric interpretation for curve fitting: The weights to form $D_i^{(k)}$, which gives the LSPIA correction direction, while α_k determines the correction length.*

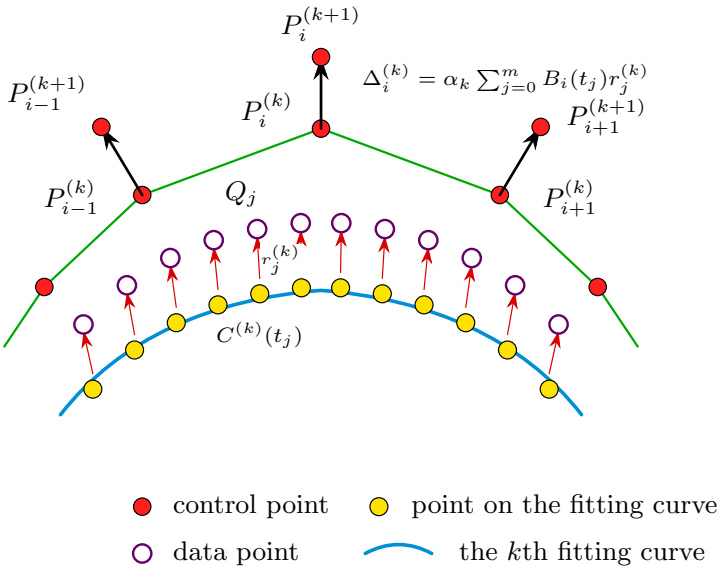


Figure 1. Geometric interpretation of the SaLSPIA method for least-squares B-spline curve fitting.

Remark 2.2. *It was shown in [28] that LSPIA is equivalent to a gradient descent method for minimizing the least-squares fitting objective. In the matrix notation used in Section 3, the corresponding LSPIA adjusting-vector matrix is*

$$d_k = A^{\top}(Q - AP_k).$$

Under this equivalence, the two local weights $\alpha_k^{(1)}$ and $\alpha_k^{(2)}$ are the counterparts of the long and short BB step sizes [29], respectively. The positive root $\widetilde{\alpha}_k$ of the quadratic equation (2.9) *interpolates continuously between $\alpha_k^{(1)}$ and $\alpha_k^{(2)}$ through the parameter m_k. The criterion* (2.12) *corresponds to the Grippo–Lampariello–Lucidi (GLL) nonmonotone line search [30].*

Remark 2.3. *In contrast to the interpolation parameter of [31], which is defined through angle information between successive iterative directions, the interpolation parameter m_k of SaLSPIA is constructed from the B-spline collocation matrix. It is obtained by scaling the scalar b_k, computed from two consecutive sets of adjusting vectors in* (2.5)*, by Λ_B. Since Λ_B provides the spectral bound $\lambda_{\max}(A^{\top}A) \le \Lambda_B$, this scaling normalizes b_k with respect to the spectral scale of the least-squares B-spline fitting problem.*

2.2. Surface fitting

The curve case can be extended to tensor-product surfaces. Let

$$\{Q_{hr} \mid h = 0, \ldots, m_u,\ r = 0, \ldots, m_v\} \subset \mathbb{R}^d$$

be the data points with parameters $\{(u_h, v_r) \mid h = 0, \ldots, m_u,\ r = 0, \ldots, m_v\}$, and let $\{B_i^u(u)\}_{i=0}^{n_u}$ and $\{B_j^v(v)\}_{j=0}^{n_v}$ be the B-spline bases in the u- and v-directions. At the beginning of the iteration, we select $\{P_{ij}^{(0)}\}_{i=0,j=0}^{n_u,n_v}$ as the control point set and construct the initial B-spline fitting surface

$$S^{(0)}(u, v) = \sum_{i=0}^{n_u} \sum_{j=0}^{n_v} B_i^u(u) B_j^v(v) P_{ij}^{(0)}, \qquad 0 \le u, v \le 1.$$

Set

$$\Lambda_B^u = \max_{0 \le i \le n_u} \sum_{h=0}^{m_u} B_i^u(u_h), \qquad \Lambda_B^v = \max_{0 \le j \le n_v} \sum_{r=0}^{m_v} B_j^v(v_r),$$

and

$$\Lambda_B^s = \Lambda_B^u \Lambda_B^v, \qquad \alpha_0 = \frac{1}{\Lambda_B^s}.$$

For $k \ge 0$, suppose that we have obtained the kth B-spline fitting surface

$$S^{(k)}(u, v) = \sum_{i=0}^{n_u} \sum_{j=0}^{n_v} B_i^u(u) B_j^v(v) P_{ij}^{(k)}.$$

The difference vectors are

$$R_{hr}^{(k)} = Q_{hr} - S^{(k)}(u_h, v_r),$$

and we set

$$D_{ij}^{(k)} = \sum_{h=0}^{m_u} \sum_{r=0}^{m_v} B_i^u(u_h) B_j^v(v_r) R_{hr}^{(k)}.$$

The first step is

$$P_{ij}^{(1)} = P_{ij}^{(0)} + \alpha_0 D_{ij}^{(0)}, \qquad i = 0, \ldots, n_u, \quad j = 0, \ldots, n_v.$$

For $k \ge 1$, define

$$\rho_\ell^s = \sum_{i=0}^{n_u} \sum_{j=0}^{n_v} \|D_{ij}^{(\ell)}\|^2, \qquad \ell = k-1, k, \tag{2.13}$$

and

$$\zeta_k^s = \sum_{i=0}^{n_u} \sum_{j=0}^{n_v} \left\langle D_{ij}^{(k-1)}, D_{ij}^{(k)} \right\rangle. \tag{2.14}$$

Set

$$b_k^s = \frac{\rho_{k-1}^s - \zeta_k^s}{\alpha_{k-1}}, \qquad c_k^s = \frac{\rho_{k-1}^s - 2\zeta_k^s + \rho_k^s}{\alpha_{k-1}^2}.$$

The surface trial weight $\bar{\alpha}_k$ is obtained from (2.6)–(2.11) by replacing

$$(\rho_{k-1}, b_k, c_k, \Lambda_B) \quad \text{with} \quad (\rho_{k-1}^s, b_k^s, c_k^s, \Lambda_B^s).$$

Let

$$\Phi_k^s = \frac{1}{2}\sum_{h=0}^{m_u}\sum_{r=0}^{m_v}\|R_{hr}^{(k)}\|^2, \qquad F_k^s = \max_{0\le q\le \min(k,M-1)} \Phi_{k-q}^s.$$

For a given weight α, consider the fitting surface generated by the tentative control points $P_{ij}^{(k)} + \alpha D_{ij}^{(k)}$, namely

$$S_\alpha^{(k+1)}(u,v) = \sum_{i=0}^{n_u}\sum_{j=0}^{n_v} B_i^u(u)B_j^v(v)(P_{ij}^{(k)} + \alpha D_{ij}^{(k)}).$$

Starting with $\alpha = \bar{\alpha}_k$, we accept $\alpha_k = \alpha$ if

$$\frac{1}{2}\sum_{h=0}^{m_u}\sum_{r=0}^{m_v}\left\|Q_{hr} - S_\alpha^{(k+1)}(u_h, v_r)\right\|^2 \le F_k^s - \sigma\alpha\rho_k^s \tag{2.15}$$

holds. Otherwise, α is replaced by $\alpha/2$ until (2.15) is satisfied. Once α_k is determined, the adjusting vector for the (i, j)th control point is

$$\Delta_{ij}^{(k)} = \alpha_k D_{ij}^{(k)}.$$

Then the control points are updated by

$$P_{ij}^{(k+1)} = P_{ij}^{(k)} + \Delta_{ij}^{(k)}, \qquad i = 0,\dots,n_u, \quad j = 0,\dots,n_v,$$

and the $(k+1)$st B-spline fitting surface is

$$S^{(k+1)}(u,v) = \sum_{i=0}^{n_u}\sum_{j=0}^{n_v} B_i^u(u)B_j^v(v)P_{ij}^{(k+1)}.$$

For surface fitting with missing data, let Ω denote the index set of the observed parameter pairs. In this case, all sums over the data indices h and r in the adjusting-vector and fitting-error computations are restricted to $(h, r) \in \Omega$.

Remark 2.4. *Remarks 2.1 and 2.2 also hold for the surface case with the substitutions used above.*

3. Convergence analysis of SaLSPIA

In this section, we first analyze the convergence of SaLSPIA for B-spline curve fitting and then extend the analysis to tensor-product B-spline surface fitting. Both the full-rank and rank-deficient cases are considered.

Arrange the data points and the control points as

$$Q = [Q_0, Q_1, \dots, Q_m]^\top \in \mathbb{R}^{(m+1)\times d}, \qquad P_k = [P_0^{(k)}, P_1^{(k)}, \dots, P_n^{(k)}]^\top \in \mathbb{R}^{(n+1)\times d}.$$

Here,

$$A \in \mathbb{R}^{(m+1)\times(n+1)}.$$

Let

$$H = A^\top A, \qquad r^{(k)} = Q - AP_k, \qquad d_k = A^\top r^{(k)}.$$

Then d_k collects the vectors $D_i^{(k)}$ used in the LSPIA control-point correction. The curve iteration in Section 2.1 can be written as

$$P_{k+1} = P_k + \alpha_k d_k. \tag{3.1}$$

The fitting error is denoted by

$$\Phi(P) = \frac{1}{2}\|Q - AP\|_F^2.$$

Since

$$d_k = A^\top Q - HP_k,$$

we have

$$d_k = d_{k-1} - \alpha_{k-1} H d_{k-1}, \qquad k \geq 1. \tag{3.2}$$

Therefore, the quantities used in the construction of the weights can be written in matrix form as

$$\rho_{k-1} = \|d_{k-1}\|_F^2, \qquad \zeta_k = \langle d_{k-1}, d_k\rangle_F, \qquad \rho_k = \|d_k\|_F^2, \tag{3.3}$$

and

$$b_k = \langle d_{k-1}, Hd_{k-1}\rangle_F, \qquad c_k = \|Hd_{k-1}\|_F^2. \tag{3.4}$$

Hence, the two local weights in (2.6) are

$$\alpha_k^{(1)} = \frac{\|d_{k-1}\|_F^2}{\langle d_{k-1}, Hd_{k-1}\rangle_F}, \qquad \alpha_k^{(2)} = \frac{\langle d_{k-1}, Hd_{k-1}\rangle_F}{\|Hd_{k-1}\|_F^2}. \tag{3.5}$$

In the following analysis, we consider the case $d_k \neq 0$. If $d_k = 0$, then

$$A^\top(Q - AP_k) = 0,$$

and the normal equation has already been satisfied.

Lemma 3.1. *Assume that $d_{k-1} \neq 0$ and $\langle d_{k-1}, Hd_{k-1}\rangle_F > 0$. Then* (2.9) *has a unique positive root, given by* (2.10)*. Moreover, the trial weight satisfies*

$$\alpha_k^{(2)} \leq \bar{\alpha}_k \leq \alpha_k^{(1)}. \tag{3.6}$$

Proof. Let $a = \rho_{k-1}$, $b = b_k$, $c = c_k$, and $m = m_k$. Since $a > 0$, $b > 0$, and $c > 0$, Eq (2.9)

$$(1-m)c\alpha^2 + (2m-1)b\alpha - ma = 0$$

has exactly one positive root when $0 < m < 1$ because the product of its two roots is negative. When $m = 1$, it reduces to $b\alpha - a = 0$, whose positive solution is a/b. Thus, (2.10) follows by solving the quadratic equation and taking the positive root.

By the Cauchy–Schwarz inequality, $ac - b^2 \geq 0$. Let

$$\gamma = ac - b^2, \qquad D(m) = b^2 + 4m(1-m)\gamma.$$

The positive root reads

$$\widetilde{\alpha}(m) = \frac{2ma}{(2m-1)b + \sqrt{D(m)}},$$

which satisfies

$$\lim_{m\to 0^+}\widetilde{\alpha}(m)=\frac{b}{c},\qquad \widetilde{\alpha}(1)=\frac{a}{b}.$$

Direct differentiation gives

$$\frac{d\widetilde{\alpha}}{dm}=\frac{2a}{((2m-1)b+\sqrt{D(m)})^2}\cdot\frac{b^2+2m\gamma-b\sqrt{D(m)}}{\sqrt{D(m)}}.$$

Since

$$(b^2+2m\gamma)^2-b^2D(m)=4m^2\gamma ac\geq 0,$$

we have $\frac{d\widetilde{\alpha}}{dm}\geq 0$. Evaluating at $m=m_k$ therefore gives

$$\frac{b}{c}\leq\widetilde{\alpha}_k\leq\frac{a}{b},$$

i.e.,

$$\alpha_k^{(2)}\leq\widetilde{\alpha}_k\leq\alpha_k^{(1)}.$$

Together with the definition of $\bar{\alpha}_k$ in (2.11), we obtain (3.6). □

Lemma 3.2. *Let $H=A^\top A$, and let Λ_B be defined by* (2.1)*. Then*

$$\lambda_{\max}(H)\leq\Lambda_B.$$

Proof. Since $B_i(t_j)\geq 0$, all entries of H are nonnegative. By the partition of unity of the B-spline basis,

$$\sum_l H_{il}=\sum_j B_i(t_j)\sum_l B_l(t_j)=\sum_j B_i(t_j).$$

Hence,

$$\|H\|_\infty=\max_i\sum_l H_{il}=\Lambda_B.$$

Since H is symmetric and positive semidefinite,

$$\lambda_{\max}(H)=\rho(H)\leq\|H\|_\infty=\Lambda_B.$$

□

3.1. Full-rank case

Assume that A has full column rank. Then $H=A^\top A$ is positive definite. Let

$$0<\lambda_{\min}\leq\lambda_i(H)\leq\lambda_{\max}$$

be the eigenvalues of H, and let

$$P^*=(A^\top A)^{-1}A^\top Q$$

be the control-point matrix of the least-squares fitting curve

$$C^*(t)=\sum_{i=0}^{n}B_i(t)P_i^*.$$

For the control-point error $\mathcal{E}_k=P_k-P^*$, the SaLSPIA iteration gives

$$\mathcal{E}_{k+1}=(I-\alpha_k H)\mathcal{E}_k. \tag{3.7}$$

Lemma 3.3. *For $k \geq 1$ with $d_{k-1} \neq 0$, the trial weight generated by Algorithm 1 satisfies*

$$\frac{1}{\Lambda_B} \leq \bar{\alpha}_k \leq \frac{1}{\lambda_{\min}}. \tag{3.8}$$

Proof. If the safeguard branch is active, Algorithm 1 sets

$$\bar{\alpha}_k = \frac{1}{\Lambda_B}.$$

Since

$$\lambda_{\min} \leq \lambda_{\max} \leq \Lambda_B,$$

the result follows immediately.

Otherwise, $\bar{\alpha}_k$ is determined by the BB-type interpolation. Since H is positive definite, the Rayleigh quotient at $d_{k-1} \neq 0$ satisfies

$$\lambda_{\min}\|d_{k-1}\|_F^2 \leq \langle d_{k-1}, Hd_{k-1}\rangle_F \leq \lambda_{\max}\|d_{k-1}\|_F^2.$$

This gives

$$\frac{1}{\lambda_{\max}} \leq \alpha_k^{(1)} \leq \frac{1}{\lambda_{\min}}.$$

Moreover,

$$\|Hd_{k-1}\|_F^2 = \langle d_{k-1}, H^2 d_{k-1}\rangle_F.$$

Since

$$\lambda_{\min} H \leq H^2 \leq \lambda_{\max} H,$$

we have

$$\frac{1}{\lambda_{\max}} \leq \alpha_k^{(2)} \leq \frac{1}{\lambda_{\min}}.$$

It follows from Lemma 3.1 that

$$\frac{1}{\lambda_{\max}} \leq \bar{\alpha}_k \leq \frac{1}{\lambda_{\min}}.$$

Finally, since $\lambda_{\max} \leq \Lambda_B$,

$$\frac{1}{\Lambda_B} \leq \frac{1}{\lambda_{\max}},$$

which proves (3.8) in both branches. □

Lemma 3.4. *Assume that $\sigma \in (0, 1/2]$. Then the halving process in the fitting-error criterion terminates after finitely many steps, and the resulting weights satisfy*

$$\frac{1-\sigma}{\Lambda_B} \leq \alpha_k \leq \frac{1}{\lambda_{\min}}, \qquad k \geq 0. \tag{3.9}$$

Proof. For any $\alpha > 0$,

$$\Phi(P_k + \alpha d_k) = \Phi(P_k) - \alpha\|d_k\|_F^2 + \frac{\alpha^2}{2}\|Ad_k\|_F^2.$$

By Lemma 3.2,

$$\|Ad_k\|_F^2 = \langle d_k, Hd_k\rangle_F \le \Lambda_B\|d_k\|_F^2.$$

Therefore,

$$\Phi(P_k + \alpha d_k) \le \Phi(P_k) - \alpha\left(1 - \frac{\alpha\Lambda_B}{2}\right)\|d_k\|_F^2.$$

If $\alpha \le 2(1-\sigma)/\Lambda_B$, then

$$\Phi(P_k + \alpha d_k) \le \Phi(P_k) - \sigma\alpha\|d_k\|_F^2 \le F_k - \sigma\alpha\|d_k\|_F^2.$$

Hence, the fitting-error criterion is satisfied for all sufficiently small α, and the halving process terminates.

For $k = 0$, $\alpha_0 = 1/\Lambda_B \ge (1-\sigma)/\Lambda_B$.

For $k \ge 1$, Lemma 3.3 gives

$$\bar{\alpha}_k \ge \frac{1}{\Lambda_B}.$$

Set

$$\eta = \frac{1-\sigma}{\Lambda_B}.$$

From the preceding estimate, every $\alpha \le 2\eta$ satisfies the fitting-error criterion. Let $h \ge 0$ denote the number of halvings performed by the line search, with $\alpha_k = 2^{-h}\bar{\alpha}_k$. If $h = 0$, then

$$\alpha_k = \bar{\alpha}_k \ge 1/\Lambda_B \ge \eta.$$

If $h \ge 1$, then the previous trial weight $2\alpha_k$ is rejected. Since every weight not exceeding 2η is accepted, this rejection implies $2\alpha_k > 2\eta$. Hence, $\alpha_k > \eta$. Therefore,

$$\alpha_k \ge \frac{1-\sigma}{\Lambda_B}, \qquad k \ge 1.$$

Together with $\alpha_0 = 1/\Lambda_B \ge (1-\sigma)/\Lambda_B$, this proves the lower bound in (3.9).

The upper bound follows because the halving process only decreases the trial weight, and Lemma 3.3 gives $\bar{\alpha}_k \le 1/\lambda_{\min}$ for $k \ge 1$. Also, $\alpha_0 = 1/\Lambda_B \le 1/\lambda_{\min}$ by Lemma 3.2. □

Theorem 3.5. *Let A be of full column rank. Then the control points generated by SaLSPIA satisfy*

$$\lim_{k\to\infty} P_k = P^*, \qquad \lim_{k\to\infty} d_k = 0.$$

Consequently, the B-spline fitting curve sequence $\{C^{(k)}(t)\}_{k=0}^{\infty}$ converges to the least-squares fitting curve $C^*(t)$.

Proof. Let

$$F_k = \max_{0 \le j \le \min(k, M-1)} \Phi(P_{k-j}).$$

By the fitting-error criterion,

$$\Phi(P_{k+1}) \le F_k - \sigma \alpha_k \|d_k\|_F^2.$$

Every term in the window defining F_{k+1} is bounded above by F_k, so $\{F_k\}$ is nonincreasing. Since $\Phi(P) \ge \Phi(P^*)$, $\{F_k\}$ is bounded below and therefore has a limit.

Using Lemma 3.4,

$$\Phi(P_{k+1}) \le F_k - \frac{\sigma(1-\sigma)}{\Lambda_B} \|d_k\|_F^2.$$

For each block $W_l = \{lM, lM+1, \ldots, (l+1)M-1\}$,

$$F_{(l+1)M} \le F_{lM} - \frac{\sigma(1-\sigma)}{\Lambda_B} \min_{k \in W_l} \|d_k\|_F^2.$$

Summing over l yields

$$\sum_{l=0}^{\infty} \min_{k \in W_l} \|d_k\|_F^2 < \infty.$$

Thus, there exists a subsequence $\{k_l\}$ such that $\|d_{k_l}\|_F \to 0$. Since

$$d_k = A^\top Q - H P_k = H(P^* - P_k)$$

and H is positive definite, we have $P_{k_l} \to P^*$.

We now extend this convergence to the full sequence. By Lemma 3.4,

$$\|P_{k+1} - P^*\|_F \le \|P_k - P^*\|_F + \alpha_k \|d_k\|_F \le \left(1 + \frac{\lambda_{\max}}{\lambda_{\min}}\right) \|P_k - P^*\|_F.$$

Hence, for each fixed $j \in \{0, 1, \ldots, M-1\}$, $P_{k_l+j} \to P^*$ as $l \to \infty$, and therefore $\Phi(P_{k_l+j}) \to \Phi(P^*)$. Consequently, $F_{k_l+M-1} \to \Phi(P^*)$. Since $\{F_k\}$ is nonincreasing, its limit is $\Phi(P^*)$. Together with

$$\Phi(P^*) \le \Phi(P_k) \le F_k,$$

the squeeze theorem yields $\Phi(P_k) \to \Phi(P^*)$. Since the least-squares solution is unique in the full-rank case, $P_k \to P^*$ and $d_k = H(P^* - P_k) \to 0$.

Finally,

$$C^{(k)}(t) - C^*(t) = \sum_{i=0}^{n} B_i(t)(P_i^{(k)} - P_i^*).$$

Since the B-spline basis is bounded on $[t_0, t_m]$, the convergence of the control points implies the convergence of the fitting curve sequence to $C^*(t)$. □

3.2. R-linear convergence estimate

Theorem 3.6. *Let A be of full column rank. Then the SaLSPIA fitting curve sequence converges R-linearly to the least-squares fitting curve. There exists a constant $C > 0$ such that*

$$\|d_k\|_F \le C\theta^k \|d_0\|_F, \qquad k \ge 0,$$

where

$$\theta = 1 - \frac{(1-\sigma)\lambda_{\min}}{\Lambda_B} \in (0,1).$$

Consequently,

$$\|P_k - P^*\|_F \le \frac{C\lambda_{\max}}{\lambda_{\min}} \theta^k \|P_0 - P^*\|_F.$$

Proof. The fitting error is $\Phi(P) = \frac{1}{2}\|Q - AP\|_F^2$, and $d_k = A^\top Q - HP_k$ with $H = A^\top A$. We verify that the weights generated by SaLSPIA satisfy Property B in [32, Theorem 1] for the sequence $\{d_k\}$. Stacking the d coordinate columns of the control-point matrix into a single vector, the iteration $P_{k+1} = P_k + \alpha_k d_k$ is of the form treated in [32, Theorem 1], with associated matrix $I_d \otimes H$, where d is the coordinate dimension of the data points. The Frobenius inner product then coincides with the Euclidean inner product of the stacked vectors. Since $I_d \otimes H$ is symmetric positive definite, it admits an orthogonal diagonalization. If some eigenvalues are repeated, the corresponding eigenspace components can be grouped together, and the argument in [32, Theorem 1] applies to the resulting distinct eigenvalue groups. Therefore, the theorem applies to $I_d \otimes H$.

We choose

$$M_1 = \frac{\Lambda_B}{1-\sigma}, \qquad \psi(z) \equiv 1, \qquad v(0) = 0, \quad v(k) = k-1 \ (k \ge 1),$$

and take $m = 2$ in Property B, the smallest window admitting $v(k) = k - 1$. By Lemma 3.4,

$$\lambda_{\min} \le \alpha_k^{-1} \le \frac{\Lambda_B}{1-\sigma} = M_1, \qquad k \ge 0,$$

which is condition (i). For condition (ii), we first note that $\bar{\alpha}_k \le \alpha_k^{(1)}$ in both branches. If the safeguard branch is inactive, this follows from Lemma 3.1. If the safeguard branch is active, then

$$\bar{\alpha}_k = \frac{1}{\Lambda_B} \le \frac{1}{\lambda_{\max}} \le \alpha_k^{(1)}.$$

Since the line search only decreases the trial weight, we obtain

$$\alpha_k \le \bar{\alpha}_k \le \alpha_k^{(1)} = \frac{\|d_{k-1}\|_F^2}{\langle d_{k-1}, Hd_{k-1}\rangle_F}, \qquad k \ge 1,$$

which is condition (ii) with $\psi(z) \equiv 1$ and $v(k) = k - 1$ after stacking all columns of d_{k-1}. For the initial index,

$$\alpha_0 = \frac{1}{\Lambda_B} \le \frac{1}{\lambda_{\max}} \le \frac{\|d_0\|_F^2}{\langle d_0, Hd_0\rangle_F},$$

where the first inequality follows from Lemma 3.2. Hence, Property B holds. Applying [32, Theorem 1] gives

$$\|d_k\|_F \le C\left(1 - \frac{(1-\sigma)\lambda_{\min}}{\Lambda_B}\right)^k \|d_0\|_F.$$

Since $d_k = H(P^* - P_k)$, we have

$$\|P_k - P^*\|_F \le \frac{\|d_k\|_F}{\lambda_{\min}}, \qquad \|d_0\|_F \le \lambda_{\max}\|P_0 - P^*\|_F.$$

Combining these inequalities gives the control-point error bound. □

Remark 3.7. *Although Theorem 3.6 establishes R-linear convergence, the numerical performance of SaLSPIA is observed to be better. A representative comparison between the observed decay of $\|d_k\|_F/\|d_0\|_F$ and the theoretical rate factor θ^k is reported in Table 1. A theoretical proof of a higher-order rate is not available, which is a known limitation of the convergence analysis of BB-type methods (see, e.g., [33, 34]).*

Table 1. Observed normalized decay of $\|d_k\|_F$ for SaLSPIA in Example 4.3. Since the constant C in the theoretical estimate is an existence constant, we report the rate factor θ^k, where $\theta = 1 - (1-\sigma)\lambda_{\min}/\Lambda_B$.

k	$\|d_k\|_F/\|d_0\|_F$	θ^k
0	1.000e+00	1.000e+00
5	1.046e−01	9.656e−01
10	4.294e−02	9.323e−01
15	3.929e−03	9.003e−01
20	1.974e−03	8.693e−01
22	9.489e−04	8.572e−01

3.3. Rank-deficient case

We now consider the case in which the collocation matrix A is rank deficient, with $H = A^\top A$ singular. Let

$$\mathcal{R} = \text{range}(A^\top), \qquad \mathcal{N} = \ker(A),$$

and let $\Pi_{\mathcal{R}}$ and $\Pi_{\mathcal{N}}$ be the orthogonal projections onto $\mathcal{R}$ and $\mathcal{N}$, respectively. Since $\mathcal{R}$ and $\mathcal{N}$ are orthogonal complements, each control-point matrix admits the columnwise orthogonal decomposition

$$P_k = P_k^{\mathcal{R}} + P_k^{\mathcal{N}}, \qquad P_k^{\mathcal{R}} = \Pi_{\mathcal{R}} P_k, \quad P_k^{\mathcal{N}} = \Pi_{\mathcal{N}} P_k.$$

Let $\lambda_{\min}^+$ be the smallest positive eigenvalue of $H = A^\top A$. Here and below, the superscript † denotes the Moore–Penrose (M–P) pseudoinverse. Let $P^* = A^\dagger Q$ be the least-squares solution with the minimum Frobenius norm. Since $\text{range}(A^\dagger) = \text{range}(A^\top)$, the columns of P^* belong to $\mathcal{R}$.

Lemma 3.8. *For all $k \ge 0$,*

$$d_k \in \mathcal{R}, \qquad P_k^{\mathcal{N}} = P_0^{\mathcal{N}}.$$

Proof. Since $d_k = A^\top r^{(k)}$, we have $d_k \in \text{range}(A^\top) = \mathcal{R}$. Because $\mathcal{R}$ and $\mathcal{N}$ are orthogonal complements, $\Pi_{\mathcal{N}} d_k = 0$. Applying $\Pi_{\mathcal{N}}$ to the update $P_{k+1} = P_k + \alpha_k d_k$ gives

$$P_{k+1}^{\mathcal{N}} = P_k^{\mathcal{N}} + \alpha_k \Pi_{\mathcal{N}} d_k = P_k^{\mathcal{N}}.$$

Therefore, $P_k^{\mathcal{N}} = P_0^{\mathcal{N}}$ for all $k \geq 0$. □

Lemma 3.9. *Assume that $\sigma \in (0, 1/2]$. In the rank-deficient case, the weights generated by the fitting-error criterion satisfy*

$$\frac{1-\sigma}{\Lambda_B} \leq \alpha_k \leq \frac{1}{\lambda_{\min}^+}, \qquad k \geq 0.$$

Proof. By Lemma 3.8, $d_{k-1} \in \mathcal{R}$. On $\mathcal{R}$, the matrix H is positive definite with eigenvalues in $[\lambda_{\min}^+, \lambda_{\max}]$. Repeating the safeguard-aware argument of Lemma 3.3 on $\mathcal{R}$ gives

$$\frac{1}{\Lambda_B} \leq \bar{\alpha}_k \leq \frac{1}{\lambda_{\min}^+}, \qquad k \geq 1.$$

The proof of Lemma 3.4 then applies with $\lambda_{\min}$ replaced by $\lambda_{\min}^+$, yielding the stated bounds. □

Theorem 3.10. *Let A be rank deficient. Then the control points generated by SaLSPIA satisfy*

$$\lim_{k\to\infty} \|d_k\|_F = 0, \qquad \lim_{k\to\infty} P_k = P^* + P_0^{\mathcal{N}}.$$

Consequently, the generated B-spline fitting curve sequence converges to a least-squares fitting curve. The limiting control-point matrix $P^* + P_0^{\mathcal{N}}$ is a least-squares solution and generally depends on the initial value through $P_0^{\mathcal{N}}$. It coincides with the minimum-Frobenius-norm M–P pseudoinverse solution P^* precisely when $P_0^{\mathcal{N}} = 0$. Moreover, the convergence is R-linear. There exists a constant $C > 0$ such that

$$\|d_k\|_F \leq C(\theta^+)^k \|d_0\|_F, \qquad k \geq 0,$$

where

$$\theta^+ = 1 - \frac{(1-\sigma)\lambda_{\min}^+}{\Lambda_B} \in (0, 1).$$

Proof. By Lemma 3.8, the iteration changes only the $\mathcal{R}$-component of the control points, while $P_k^{\mathcal{N}} = P_0^{\mathcal{N}}$ for all k. On $\mathcal{R}$, the matrix H is positive definite with the smallest positive eigenvalue $\lambda_{\min}^+$. Since the columns of P^* and $P_k^{\mathcal{R}}$ belong to $\mathcal{R}$, we have

$$d_k = A^\top Q - H P_k = H(P^* - P_k^{\mathcal{R}}).$$

Using Lemma 3.9, the proof of Theorem 3.5 applies on $\mathcal{R}$, giving $d_k \to 0$ and $P_k^{\mathcal{R}} \to P^*$. Combined with $P_k^{\mathcal{N}} = P_0^{\mathcal{N}}$, this yields $P_k \to P^* + P_0^{\mathcal{N}}$. Since P^* satisfies the normal equation and $AP_0^{\mathcal{N}} = 0$, the limit $P^* + P_0^{\mathcal{N}}$ also satisfies the normal equation and is therefore a least-squares control-point solution. If $P_0^{\mathcal{N}} \neq 0$, the limiting control points differ from P^*, although they produce the same fitted values at the sampled parameter values. In general, they may define a different B-spline curve away from those parameter values.

For the R-linear estimate, the verification of Property B in the proof of Theorem 3.6 applies on the subspace of control-point matrices with columns in $\mathcal{R} = \mathrm{range}(A^\top)$, which is invariant by Lemma 3.8. There, H is positive definite with smallest positive eigenvalue $\lambda^+_{\min}$, so, by Lemma 3.9, the bound on α_k^{-1} holds with $\lambda_{\min}$ replaced by $\lambda^+_{\min}$, while the safeguard-aware comparison established in the proof of Theorem 3.6 gives $\alpha_k \le \alpha_k^{(1)}$ on $\mathcal{R}$. Hence, [32, Theorem 1] gives

$$\|d_k\|_F \le C\left(1 - \frac{(1-\sigma)\lambda^+_{\min}}{\Lambda_B}\right)^k \|d_0\|_F.$$

This proves the stated R-linear estimate. □

3.4. Convergence for surface fitting

We now show that the tensor-product surface case can be reduced to the curve case by vectorization. Let $A_u \in \mathbb{R}^{(m_u+1)\times(n_u+1)}$, $A_v \in \mathbb{R}^{(m_v+1)\times(n_v+1)}$ be the collocation matrices in the u- and v-directions. Suppose that the data points take values in $\mathbb{R}^d$. For each coordinate $\ell = 1, \dots, d$, let $Q^{(\ell)} \in \mathbb{R}^{(m_u+1)\times(m_v+1)}$, $P_k^{(\ell)} \in \mathbb{R}^{(n_u+1)\times(n_v+1)}$ denote the corresponding data matrix and control mesh, respectively. Define

$$\widetilde{Q} = [\,\mathrm{vec}(Q^{(1)}), \dots, \mathrm{vec}(Q^{(d)})] \in \mathbb{R}^{(m_u+1)(m_v+1)\times d},$$

$$\widetilde{P}_k = [\,\mathrm{vec}(P_k^{(1)}), \dots, \mathrm{vec}(P_k^{(d)})] \in \mathbb{R}^{(n_u+1)(n_v+1)\times d},$$

and set

$$A_s = A_v \otimes A_u \in \mathbb{R}^{(m_u+1)(m_v+1)\times(n_u+1)(n_v+1)}.$$

Using

$$\mathrm{vec}(A_u P_k^{(\ell)} A_v^\top) = (A_v \otimes A_u)\,\mathrm{vec}(P_k^{(\ell)}),$$

the surface fitting problem can be written in the matrix form

$$\widetilde{R}_k = \widetilde{Q} - A_s \widetilde{P}_k, \qquad \widetilde{D}_k = A_s^\top \widetilde{R}_k,$$

and the SaLSPIA surface iteration becomes

$$\widetilde{P}_{k+1} = \widetilde{P}_k + \alpha_k \widetilde{D}_k.$$

Therefore, it has exactly the same form as the curve iteration, with A, P_k, Q, and Λ_B replaced by A_s, $\widetilde{P}_k$, $\widetilde{Q}$, and Λ_B^s, respectively. Moreover, since the ℓth column of $\widetilde{D}_k$ equals $\mathrm{vec}(D_{ij}^{(k)})$ for coordinate ℓ,

$$\rho_k^s = \|\widetilde{D}_k\|_F^2, \qquad \zeta_k^s = \langle \widetilde{D}_{k-1}, \widetilde{D}_k \rangle_F,$$

which coincide with the surface quantities defined in (2.13)-(2.14). Since

$$\widetilde{D}_k = \widetilde{D}_{k-1} - \alpha_{k-1} H_s \widetilde{D}_{k-1},$$

where

$$H_s = A_s^\top A_s = (A_v^\top A_v) \otimes (A_u^\top A_u),$$

we also have

$$b_k^s = \langle \widetilde{D}_{k-1}, H_s \widetilde{D}_{k-1} \rangle_F, \qquad c_k^s = \|H_s \widetilde{D}_{k-1}\|_F^2.$$

It remains to verify the spectral bound used in the convergence analysis. By Lemma 3.2 applied in the two parametric directions,

$$\lambda_{\max}(A_u^\top A_u) \le \Lambda_B^u, \qquad \lambda_{\max}(A_v^\top A_v) \le \Lambda_B^v.$$

Because the eigenvalues of a Kronecker product are the pairwise products of the eigenvalues of its factors,

$$\lambda_{\max}(H_s) = \lambda_{\max}(A_u^\top A_u)\lambda_{\max}(A_v^\top A_v) \le \Lambda_B^u \Lambda_B^v = \Lambda_B^s,$$

where $\Lambda_B^s = \Lambda_B^u \Lambda_B^v$ is the surface constant defined in Section 2.2.

If A_s has full column rank, equivalently if both A_u and A_v have full column rank, then H_s is positive definite. Hence, the full-rank convergence results for curves apply directly to the vectorized surface iteration. In particular, the generated tensor-product B-spline surface sequence converges *R*-linearly to the least-squares fitting surface, where

$$\theta_s = 1 - \frac{(1-\sigma)\lambda_{\min}(H_s)}{\Lambda_B^s} \in (0,1).$$

We next consider surface fitting with missing data. Let $\Omega \subseteq \{0,\dots,m_u\} \times \{0,\dots,m_v\}$ denote the set of observed parameter pairs, and let $S_\Omega \in \mathbb{R}^{|\Omega| \times (m_u+1)(m_v+1)}$ be the row-selection matrix that extracts the observations indexed by Ω from the complete tensor-product grid. Define

$$A_\Omega = S_\Omega A_s = S_\Omega (A_v \otimes A_u), \qquad \widetilde{Q}_\Omega = S_\Omega \widetilde{Q}.$$

Then the residual and adjusting-vector matrices for the observed data are

$$\widetilde{R}_k^\Omega = \widetilde{Q}_\Omega - A_\Omega \widetilde{P}_k, \qquad \widetilde{D}_k^\Omega = A_\Omega^\top \widetilde{R}_k^\Omega,$$

and the SaLSPIA iteration becomes

$$\widetilde{P}_{k+1} = \widetilde{P}_k + \alpha_k \widetilde{D}_k^\Omega.$$

Thus, the missing-data surface iteration again has the same algebraic form as the curve iteration, with A replaced by A_Ω.

Let

$$H_\Omega = A_\Omega^\top A_\Omega = A_s^\top S_\Omega^\top S_\Omega A_s.$$

Since S_Ω is a row-selection matrix,

$$0 \le S_\Omega^\top S_\Omega \le I.$$

Consequently,

$$0 \le H_\Omega \le H_s,$$

and hence

$$\lambda_{\max}(H_\Omega) \le \lambda_{\max}(H_s) \le \Lambda_B^s.$$

Suppose that A_Ω is rank deficient, and define

$$\mathcal{R}_\Omega = \operatorname{range}(A_\Omega^\top), \qquad \mathcal{N}_\Omega = \ker(A_\Omega).$$

The update direction

$$\widetilde{D}_k^{\Omega} = A_{\Omega}^{\top}(\widetilde{Q}_{\Omega} - A_{\Omega}\widetilde{P}_k)$$

always lies in $\mathcal{R}_{\Omega}$. Therefore, the $\mathcal{N}_{\Omega}$-component of the initial control mesh is invariant. Moreover, H_{Ω} is positive definite on $\mathcal{R}_{\Omega}$, with smallest positive eigenvalue $\lambda_{\min}^{+}(H_{\Omega})$. Thus, the rank-deficient convergence argument for curves applies with

$$(A, H, \Lambda_B, \lambda_{\min}^{+}) \quad \text{replaced by} \quad (A_{\Omega}, H_{\Omega}, \Lambda_B^s, \lambda_{\min}^{+}(H_{\Omega})).$$

Consequently, $\widetilde{D}_k^{\Omega} \to 0$, and the generated surface sequence converges R-linearly to a least-squares fitting surface, with rate factor

$$\theta_{\Omega}^{+} = 1 - \frac{(1-\sigma)\lambda_{\min}^{+}(H_{\Omega})}{\Lambda_B^s} \in (0, 1).$$

The limiting control mesh is a least-squares solution and generally depends on the initial value through its invariant null-space component. It coincides with the minimum-Frobenius-norm M–P pseudoinverse solution $A_{\Omega}^{\dagger}\widetilde{Q}_{\Omega}$ when the initial null-space component vanishes.

4. Numerical experiments

In this section, we compare the performance of the proposed SaLSPIA with LSPIA [5], asynchronous LSPIA (ALSPIA) [22], LSPIA with memory (MLSPIA) [24], and NmLSPIA [25] on several curve and surface fitting problems. For rank-deficient least-squares fitting systems arising from missing data, we additionally compare SaLSPIA with the diagonal-weighted LSPIA variant of Lin et al. [10] (LSPIA-Lin2018).

All experiments use cubic B-spline bases ($p = 3$). Following the standard practice in B-spline data fitting [1,35], we construct the chord-length parameterization and clamped knot vectors as in Eqs (9.5) and (9.69) of Piegl and Tiller [1], and assemble the collocation matrix via the de Boor evaluation algorithm. The LSPIA, ALSPIA, MLSPIA, and NmLSPIA methods employ the recommended parameters given in the respective references. For every convergent method for which CPU time is reported, we perform five repeated runs under the same MATLAB setting and report the mean elapsed time together with the sample standard deviation.

As noted by Deng and Lin [5], LSPIA can be started with arbitrary initial control points, and an appropriate selection can reduce the number of iterations. For full-rank systems, the limiting least-squares curve or surface is unique and does not depend on the initial control points. In the rank-deficient case, Theorem 3.10 shows that only the null-space component of the initial control points may affect the limiting control mesh, while the sampled fitted values and the relative least-squares fitting error E_k are unaffected. To ensure a fair comparison, all methods use the same Deng–Lin subsampling initialization strategy [5, Eq (23)].

Following [22], we use the relative least-squares fitting error

$$E_k = \frac{\left\|A^{\top}Q - A^{\top}AP^{(k)}\right\|_F^2}{\left\|A^{\top}Q - A^{\top}AP^{(0)}\right\|_F^2} \tag{4.1}$$

for curves. For tensor-product surfaces, the corresponding relative least-squares fitting error is

$$E_k = \frac{\sum_{\ell=1}^{d} \left\| A_u^\top \left(Q^{(\ell)} - A_u P_k^{(\ell)} A_v^\top \right) A_v \right\|_F^2}{\sum_{\ell=1}^{d} \left\| A_u^\top \left(Q^{(\ell)} - A_u P_0^{(\ell)} A_v^\top \right) A_v \right\|_F^2}, \tag{4.2}$$

Here, $Q^{(\ell)} - A_u P_k^{(\ell)} A_v^\top$ is the residual matrix of the ℓth coordinate component, and $A_u^\top (Q^{(\ell)} - A_u P_k^{(\ell)} A_v^\top) A_v$ is the corresponding normal-equation residual. Thus, (4.2) is the tensor-product surface counterpart of (4.1).

The experiments are terminated when E_k drops below 10^{-6} or the iteration count reaches 10^4. The safeguard tolerance in Algorithm 1 is set to $\varepsilon_{\text{saf}} = 10^{-30}$ in all experiments. The final value of E_k is reported as E_∞.

All experiments are performed on a MacBook Pro with an Intel Core i9-9980HK 2.40 GHz processor and 32 GB memory using MATLAB R2021b. Examples 4.1, 4.2, 4.4, 4.5, and 4.8 are taken from `http://paulbourke.net/geometry/`. Example 4.3 is available from `https://github.com/giacomoorsi/ProgressiveIterationApproximation`. Example 4.6 is available from `https://github.com/dieghernan/tidyterra`, originally provided by Toitū Te Whenua Land Information New Zealand. Example 4.7 is available from `https://github.com/XuejiaoYuan/LSPIA`.

Example 4.1. *We sample* $m+1 = 15{,}001$ *points uniformly from the blob-shaped curve defined in polar coordinates by*

$$r = 2 + 4\cos(2\theta + \pi/4) + \cos(3\theta + \pi/4), \qquad x = r\cos\theta, \quad y = r\sin\theta, \qquad 0 \le \theta \le 2\pi.$$

The number of control points is $n + 1 = 3{,}001$.

Example 4.2. *We sample* $m + 1 = 10{,}001$ *points uniformly from the helix curve*

$$x = 10\cos(\theta\pi/3), \quad y = 10\sin(\theta\pi/3), \quad z = \theta\pi/3, \quad 0 \le \theta \le 2\pi.$$

The number of control points is $n + 1 = 2{,}001$.

Example 4.3. *We use* $m + 1 = 1{,}000$ *points sampled from a reindeer contour dataset. The number of control points is* $n + 1 = 288$.

Example 4.4. *We sample* 81×81 *points uniformly from Dini's surface*

$$x = \cos\theta_1 \sin\theta_2, \quad y = \sin\theta_1 \sin\theta_2, \quad z = \cos\theta_2 + \ln(\tan(\theta_2/2)) + \theta_1,$$

where $0 \le \theta_1 \le 4\pi$ *and* $0.1 \le \theta_2 \le \pi - 0.1$. *The control mesh size is* 31×31.

Example 4.5. *We sample* 121×121 *points uniformly from the peaks function*

$$f(\theta_1, \theta_2) = 3(1-\theta_1)^2 e^{-\theta_1^2 - (\theta_2+1)^2} - 10\left(\frac{\theta_1}{5} - \theta_1^3 - \theta_2^5\right) e^{-\theta_1^2 - \theta_2^2} - \frac{1}{3} e^{-(\theta_1+1)^2 - \theta_2^2},$$

where $-3 \le \theta_1 \le 3$ *and* $-4 \le \theta_2 \le 4$. *The control mesh size is* 41×41.

Example 4.6. *We use* $860 \times 600 = 516{,}000$ *points from the high-resolution LiDAR digital elevation model of the Maungawhau/Mt. Eden volcano in Auckland, New Zealand. The data are sampled on a* 1 m *grid with elevations from* 76.51 m *to* 195.65 m*. The control mesh size is* 121×121.

Example 4.7. *We use* 269 *points sampled from a G-shaped scattered closed curve. After four regions are removed,* 206 *points remain for fitting. The number of control points is* $n + 1 = 101$.

Example 4.8. *We sample* 101×101 *points uniformly from Franke's function on* $[0, 1] \times [0, 1]$,

$$f(x, y) = 0.75e^{-((9x-2)^2+(9y-2)^2)/4} + 0.75e^{-(9x+1)^2/49-(9y+1)/10} + 0.50e^{-((9x-7)^2+(9y-3)^2)/4} - 0.20e^{-(9x-4)^2-(9y-7)^2}.$$

After four local elliptical regions are removed, 8,627 *points remain for fitting. The control mesh size is* 41×41.

We report the ranks and effective condition numbers of the collocation matrices in Table 2. The numerical ranks reported in the table were computed using the default MATLAB rank tolerance. For rank-deficient cases, the usual condition number is infinite, so we use $\kappa_{\text{eff}}(A) = \sigma_{\max}(A)/\sigma_r(A)$, where $r = \text{rank}(A)$ and $\sigma_r(A)$ is the smallest nonzero singular value.

Table 2. Ranks and effective condition numbers of the collocation matrices.

Example	Setting	Observed/control size	Rank	κ_{eff}
Example 4.1	full curve	$(15001, 3001)$	$3001/3001$	4.72
Example 4.2	full curve	$(10001, 2001)$	$2001/2001$	4.72
Example 4.3	full curve	$(1000, 288)$	$288/288$	1.06×10^1
Example 4.4	full surface	$(81 \times 81,\ 31 \times 31)$	$961/961$	2.12×10^1
Example 4.5	full surface	$(121 \times 121,\ 41 \times 41)$	$1681/1681$	2.12×10^1
Example 4.6	full surface	$(860 \times 600,\ 121 \times 121)$	$14641/14641$	2.22×10^1
Example 4.1	missing curve	$(267, 101)$	$96/101$	6.48×10^5
Example 4.2	missing curve	$(458, 201)$	$189/201$	1.03×10^8
Example 4.7	missing curve	$(206, 101)$	$90/101$	1.22×10^4
Example 4.8	missing surface	$(8627, 1681)$	$1639/1681$	1.98×10^9

4.1. Parameter sensitivity

Before the performance comparison, we first fix the algorithm parameters and verify their robustness. We examine the sensitivity of SaLSPIA to the user-defined parameters σ, M, and δ on a full-rank curve example (Curve Example 4.3), a rank-deficient missing-data example (Example 4.7), and a full-rank surface example (Peaks Example 4.5), thereby covering both curve and surface fitting in the full-rank and rank-deficient settings. In all other experiments of this paper, we use the same default setting $\sigma = 10^{-4}$, $M = 10$, and $\delta = 10^{-8}$. The nonmonotone line-search framework follows [30], while the choices $\sigma = 10^{-4}$ and $M = 10$ are consistent with the numerical setting used in [36].

SaLSPIA is insensitive to σ and δ. Over $\sigma \in \{1/2, 10^{-1}, 10^{-2}, 10^{-4}, 10^{-6}\}$, the iteration count is essentially unchanged on all three examples, varying only at the extreme value $\sigma = 1/2$ on Example 4.7; varying δ over $[10^{-10}, 10^{-2}]$ gives identical results throughout. The default values therefore lie in a stable range.

The memory length M has a more visible effect, as reported in Table 3. The results show that $M = 10$ attains the minimum iteration count (IT) for all three examples and substantially reduces unnecessary halvings, supporting the default choice $M = 10$.

Table 3. Sensitivity to the memory length M with $\sigma = 10^{-4}$ and $\delta = 10^{-8}$ fixed.

Example		$M = 1$	$M = 3$	$M = 5$	$M = 10$	$M = 20$
Curve Example 4.3	E_∞	2.44×10^{-7}	4.59×10^{-7}	3.43×10^{-7}	9.00×10^{-7}	8.06×10^{-7}
	IT	28	25	24	22	22
	Halvings	12	6	6	1	0
Example 4.7	E_∞	8.50×10^{-7}	6.40×10^{-7}	5.58×10^{-7}	9.42×10^{-7}	9.42×10^{-7}
	IT	18	20	20	17	17
	Halvings	7	5	4	0	0
Peaks Example 4.5	E_∞	5.01×10^{-7}	5.73×10^{-7}	5.73×10^{-7}	5.73×10^{-7}	5.73×10^{-7}
	IT	8	8	8	8	8
	Halvings	1	0	0	0	0

4.2. Fitting data sets

For the curve-fitting cases in Examples 4.1–4.3, whose collocation matrices are of full column rank, Table 4 shows that all methods reach comparable E_∞, while SaLSPIA requires the fewest iterations and the lowest CPU time. The fitting curve obtained by SaLSPIA for Example 4.3 is displayed in Figure 2.

To illustrate the conservativeness of the theoretical rate in Remark 3.7, Table 1 gives a representative comparison between the observed decay of $\|d_k\|_F$ and the theoretical rate factor θ^k in Example 4.3.

For the surface-fitting cases in Examples 4.4–4.6, the numerical results are listed in Table 5. In particular, on the large-scale volcano model with 516,000 points, SaLSPIA achieves the smallest iteration count and CPU time among the five methods, converging in 13 iterations with a CPU time of 0.2750 ± 0.0072 s. In this large-scale surface example, the full Kronecker matrix $A_s = A_v \otimes A_u$ is not formed explicitly. Only A_u and A_v are stored, and each iteration is carried out through the tensor-product products $A_u P_k^{(\ell)} A_v^\top$ and $A_u^\top \left(Q^{(\ell)} - A_u P_k^{(\ell)} A_v^\top\right) A_v$. Since the fitting surfaces constructed by the five methods are virtually identical, we only show the bi-cubic B-spline fitting surfaces obtained by SaLSPIA in Figure 3 for Examples 4.4 and 4.5. For the volcano model in Example 4.6, the bi-cubic B-spline fitting surfaces obtained by SaLSPIA at iterations $k = 0$, 7, and 13 are displayed in Figure 4.

Table 4. The numerical results of E_∞, IT, and CPU time for curve fitting in Examples 4.1–4.3. CPU time (s) is reported as the mean ± sample standard deviation over five runs. Each column header $(m + 1,\ n + 1)$ lists the numbers of data points and control points.

Method		(15001, 3001)	(10001, 2001)	(1000, 288)
LSPIA	E_∞	9.34×10^{-7}	9.65×10^{-7}	9.89×10^{-7}
	IT	77	77	180
	CPU	3.6011 ± 0.0761	1.5080 ± 0.0112	0.0083 ± 0.0013
ALSPIA	E_∞	9.42×10^{-7}	6.29×10^{-7}	9.98×10^{-7}
	IT	6	7	125
	CPU	0.2657 ± 0.0034	0.1289 ± 0.0044	0.0061 ± 0.0009
MLSPIA	E_∞	7.34×10^{-7}	7.74×10^{-7}	8.15×10^{-7}
	IT	25	25	41
	CPU	1.6999 ± 0.0048	0.7373 ± 0.0159	0.0035 ± 0.0003
NmLSPIA	E_∞	9.74×10^{-7}	3.60×10^{-7}	9.41×10^{-7}
	IT	5	6	31
	CPU	0.4478 ± 0.0074	0.2312 ± 0.0045	0.0037 ± 0.0005
SaLSPIA	E_∞	8.16×10^{-7}	9.15×10^{-7}	9.00×10^{-7}
	IT	4	4	22
	CPU	0.2113 ± 0.0041	0.0901 ± 0.0025	0.0023 ± 0.0004

Table 5. The numerical results of E_∞, IT, and CPU for surface fitting in Examples 4.4–4.6. CPU time (s) is reported as the mean ± sample standard deviation over five runs. Each column header $(m_u + 1,\ m_v + 1,\ n_u + 1,\ n_v + 1)$ lists the data grid size and the control mesh size in the two parametric directions.

Method		(81, 81, 31, 31)	(121, 121, 41, 41)	(860, 600, 121, 121)
LSPIA	E_∞	9.76×10^{-7}	9.94×10^{-7}	9.94×10^{-7}
	IT	213	1091	1262
	CPU	0.0601 ± 0.0026	0.5386 ± 0.0202	39.3780 ± 1.5238
ALSPIA	E_∞	9.19×10^{-7}	8.70×10^{-7}	9.74×10^{-7}
	IT	37	15	33
	CPU	0.0099 ± 0.0004	0.0089 ± 0.0019	1.0432 ± 0.0144
MLSPIA	E_∞	6.92×10^{-7}	8.39×10^{-7}	8.99×10^{-7}
	IT	82	109	114
	CPU	0.0149 ± 0.0007	0.0366 ± 0.0022	2.4163 ± 0.0595
NmLSPIA	E_∞	8.66×10^{-7}	8.72×10^{-7}	9.60×10^{-7}
	IT	30	17	23
	CPU	0.0067 ± 0.0007	0.0063 ± 0.0004	0.6023 ± 0.0165
SaLSPIA	E_∞	3.37×10^{-7}	5.73×10^{-7}	4.42×10^{-7}
	IT	14	8	13
	CPU	0.0032 ± 0.0006	0.0032 ± 0.0010	0.2750 ± 0.0072

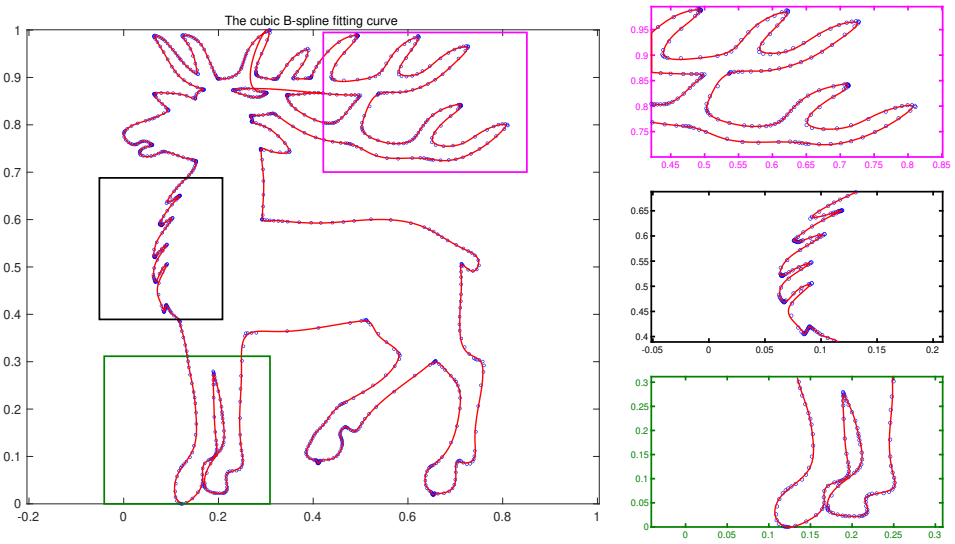


Figure 2. The cubic B-spline fitting curve for Example 4.3 (reindeer contour).

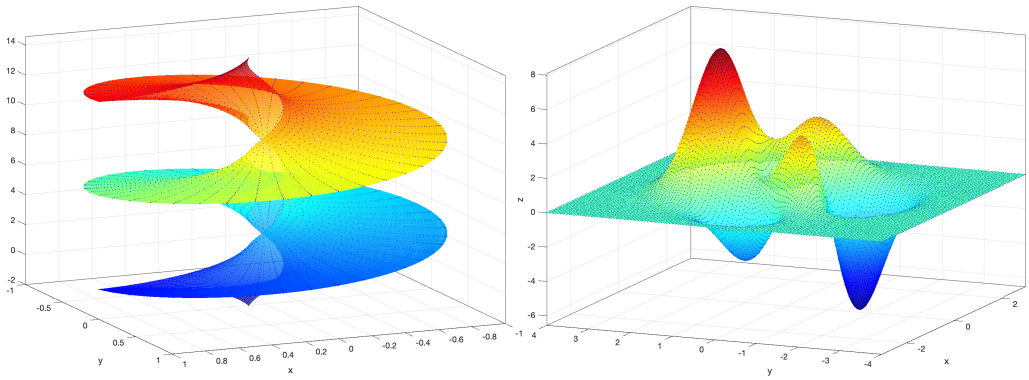
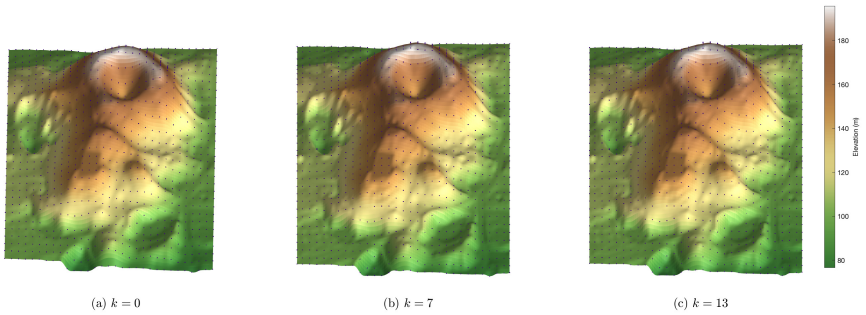


Figure 3. The bi-cubic B-spline SaLSPIA fitting surfaces for Examples 4.4 (left) and 4.5 (right).



(a) $k = 0$ (b) $k = 7$ (c) $k = 13$

Figure 4. The bi-cubic B-spline fitting surfaces obtained by SaLSPIA for Example 4.6 at iterations $k = 0$, 7, and 13.

To complement the algebraic metric E_∞ with direct geometric fitting errors, Table 6 reports representative root mean square (RMS) and maximum pointwise residuals for SaLSPIA.

Table 6. Representative geometric residuals of SaLSPIA.

Example	Type	RMS	Max
Example 4.3	curve	1.048×10^{-3}	6.255×10^{-3}
Example 4.6	surface	9.883×10^{-2}	9.298×10^{-1}

4.3. Missing data fitting

In least-squares fitting problems, missing data, holes, or insufficiently sampled regions may lead to a rank-deficient collocation matrix [10, 22]; whether the system is rank-deficient depends on the sampling and the knot configuration, and is verified numerically here. In this subsection, we compare SaLSPIA with LSPIA, ALSPIA, and LSPIA-Lin2018 on both curve and surface fitting with missing regions. The rank-deficient data/control sizes and ranks are summarized in Table 2.

We test four missing-data cases: three curves and one surface. For Examples 4.1 and 4.2, reduced sampled versions of the same curves are used, with four and three holes, respectively; Example 4.7 is a G-shaped closed curve with four removed regions; and Example 4.8 is the Franke surface with four local elliptical holes (Figures 5(a), 6(a), 7(a), and 8(a), where the missing data are shown as red circles). The numerical results are reported in Table 7. In every case, LSPIA fails to converge within 10^4 iterations, whereas LSPIA-Lin2018, ALSPIA, and SaLSPIA all converge, with SaLSPIA requiring the fewest iterations. Besides the relative error E_∞, we also report the held-out geometric errors (Hole RMS and Hole Max) evaluated at the removed data points to assess the reconstruction quality inside the missing regions; these are of the same order as those of the existing rank-deficient fitting methods. As shown in Figures 5(b), 6(b), 7(b), and 8(b), SaLSPIA fits the data well across the missing regions, confirming the robustness of the adaptive weight strategy in the rank-deficient setting.

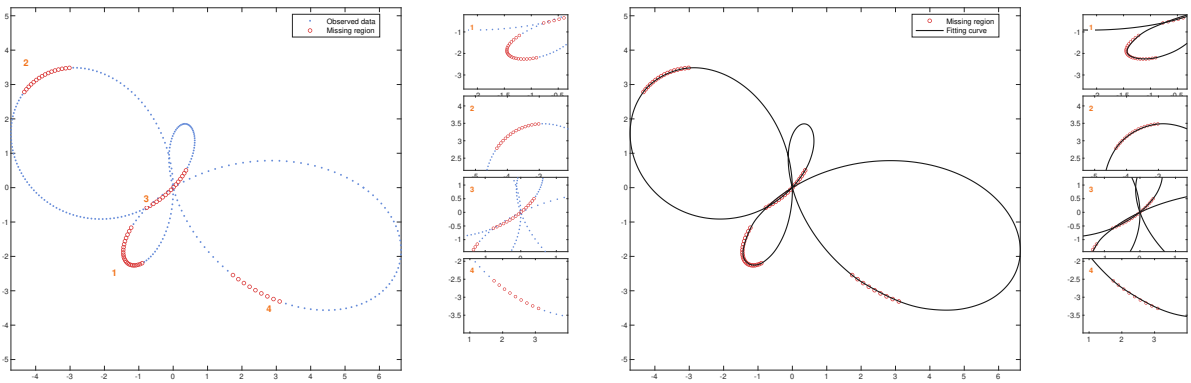


Figure 5. The initial data points with missing data and the fitting curve for Example 4.1 and their focal locations.

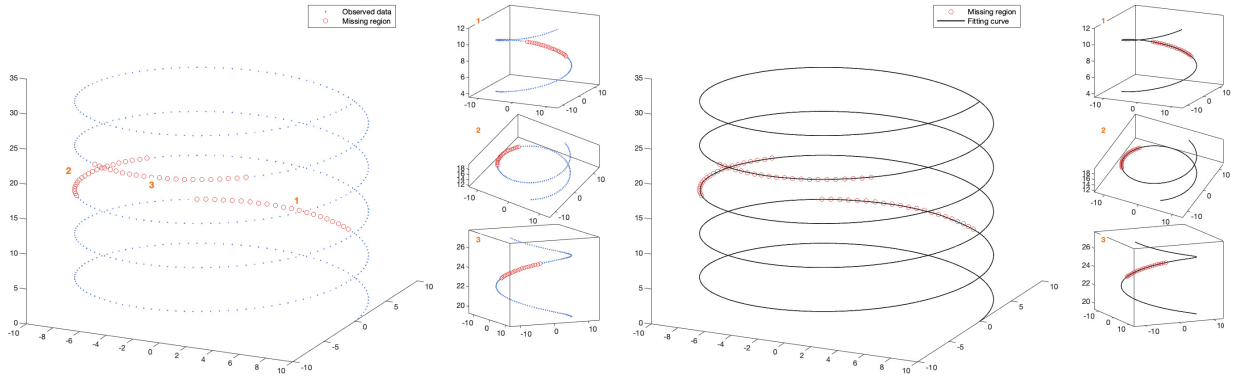


Figure 6. The initial data points with missing data and the fitting curve for Example 4.2 and their focal locations.

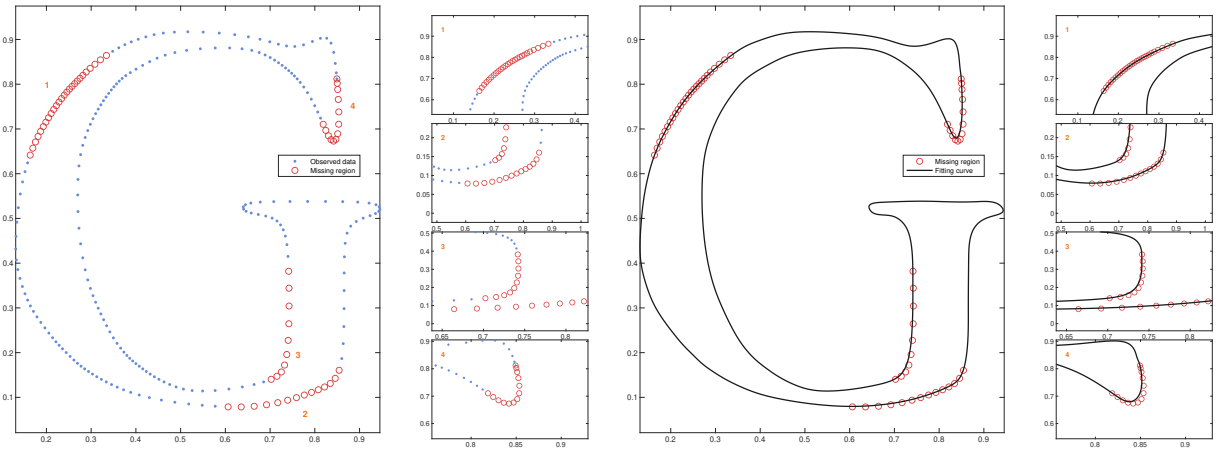


Figure 7. The initial data points with missing data and the fitting curve for Example 4.7 and their focal locations.

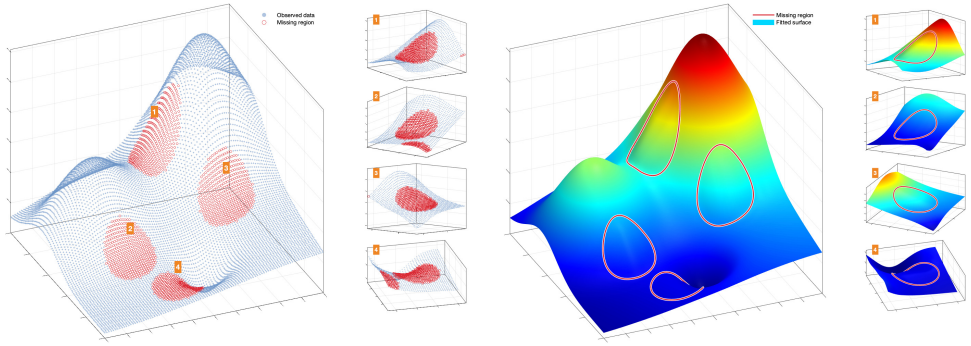


Figure 8. The initial data points with missing data and the bi-cubic B-spline fitting surface for Example 4.8 and the numbered zoom-ins of each hole.

Table 7. The numerical results of E_∞, IT, CPU, and held-out geometric errors (Hole RMS, Hole Max) for the rank-deficient curve and surface fitting (missing data). CPU time (s) is reported as the mean ± sample standard deviation over five runs.

Example	Method	IT	CPU (s), mean ± SD	E_∞	Hole RMS	Hole Max
Example 4.1	LSPIA	$> 10^4$	#	#	–	–
	LSPIA-Lin2018	18	0.000802 ± 0.000959	9.35×10^{-7}	7.273×10^{-2}	1.807×10^{-1}
	ALSPIA	38	0.000944 ± 0.000071	9.62×10^{-7}	1.188×10^{-1}	2.272×10^{-1}
	SaLSPIA	14	0.000446 ± 0.000056	4.40×10^{-7}	1.154×10^{-1}	2.202×10^{-1}
Example 4.2	LSPIA	$> 10^4$	#	#	–	–
	LSPIA-Lin2018	26	0.001005 ± 0.000124	8.70×10^{-7}	4.829×10^{-1}	1.149
	ALSPIA	32	0.000881 ± 0.000134	9.06×10^{-7}	6.488×10^{-1}	1.198
	SaLSPIA	14	0.000717 ± 0.000127	7.51×10^{-7}	6.424×10^{-1}	1.196
Example 4.7	LSPIA	$> 10^4$	#	#	–	–
	LSPIA-Lin2018	32	0.000724 ± 0.000123	9.85×10^{-7}	1.711×10^{-2}	4.209×10^{-2}
	ALSPIA	73	0.001291 ± 0.000250	9.87×10^{-7}	1.892×10^{-2}	4.146×10^{-2}
	SaLSPIA	17	0.000572 ± 0.000126	9.42×10^{-7}	1.894×10^{-2}	4.146×10^{-2}
Example 4.8	LSPIA	$> 10^4$	#	#	–	–
	LSPIA-Lin2018	40	0.459535 ± 0.004027	9.34×10^{-7}	2.059×10^{-2}	8.085×10^{-2}
	ALSPIA	53	0.625100 ± 0.002840	9.85×10^{-7}	2.980×10^{-2}	8.441×10^{-2}
	SaLSPIA	14	0.241874 ± 0.002510	7.61×10^{-7}	2.973×10^{-2}	8.441×10^{-2}

#: failed to converge within 10^4 iterations; –: held-out error is not reported for the non-convergent method.

4.4. Ablation study

We further isolate the contribution of the spectrally adaptive weight together with its interpolation mechanism (see Table 8). The effect of the memory length M has already been examined in the parameter sensitivity study (see Table 3) and is not repeated here.

Table 8. Ablation of the spectrally adaptive weight on three curve examples.

Method		Curve Example 4.2	Curve Example 4.3	Curve Example 4.7
	data/control	(10001, 2001)	(1000, 288)	(206, 101)
	rank	full	full	90
Long endpoint $\alpha_k^{(1)}$	E_∞	8.38×10^{-7}	7.76×10^{-7}	4.24×10^{-7}
	IT	4	24	21
	Halvings	0	0	2
Short endpoint $\alpha_k^{(2)}$	E_∞	2.48×10^{-7}	9.37×10^{-7}	9.45×10^{-7}
	IT	5	22	19
	Halvings	0	0	0
SaLSPIA	E_∞	9.15×10^{-7}	9.00×10^{-7}	9.42×10^{-7}
	IT	4	22	17
	Halvings	0	1	0

To further visualize the ablation effect on Example 4.7, Figure 9 plots the best-so-far convergence envelope for the three weight configurations.

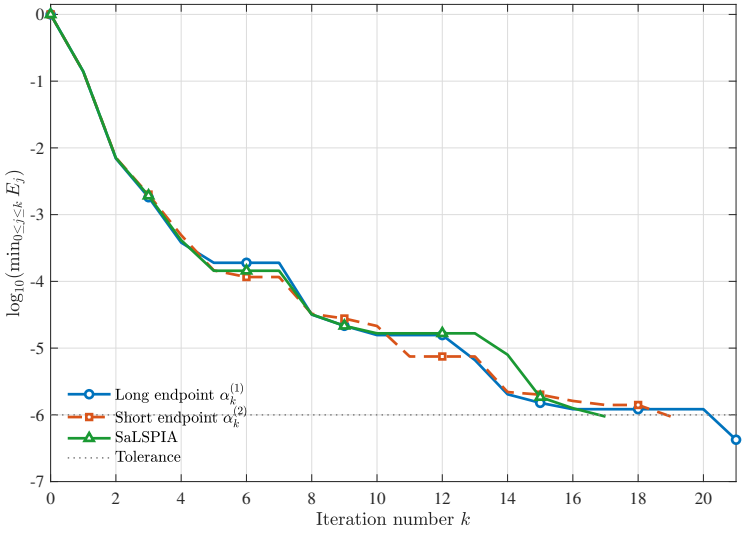


Figure 9. Best-so-far convergence envelopes of the three weight configurations in Example 4.7.

When the safeguard branch is inactive, the trial weight $\bar{\alpha}_k$ is obtained by interpolating between the long endpoint $\alpha_k^{(1)}$ and the short endpoint $\alpha_k^{(2)}$ through the interpolation parameter m_k. By Lemma 3.1, $\alpha_k^{(2)} \le \bar{\alpha}_k \le \alpha_k^{(1)}$. When the safeguard branch is active, Algorithm 1 instead uses $\bar{\alpha}_k = 1/\Lambda_B$. To show the effect of this adaptive choice, we compare three weight configurations on three curve examples in Table 8, namely the weight fixed at the long endpoint $\alpha_k^{(1)}$, the weight fixed at the short endpoint $\alpha_k^{(2)}$, and SaLSPIA itself. All three use the same GLL safeguard.

The long endpoint $\alpha_k^{(1)}$ produces a more aggressive control-point update and may trigger halvings, whereas the short endpoint $\alpha_k^{(2)}$ is more conservative and may require additional iterations. As shown in Table 8 and Figure 9, the interpolated weight enables SaLSPIA to reach the prescribed tolerance earlier than either endpoint configuration on Example 4.7. This result indicates that the adaptive interpolation provides a favorable balance between update size and safeguarding.

5. Conclusions

We have proposed SaLSPIA for B-spline curve and surface fitting, in which an adaptive weight is constructed from the adjusting vectors of two consecutive iterations. We have established global convergence and an R-linear convergence rate for both full-rank and rank-deficient fitting systems, including tensor-product surface fitting and fitting with missing data. Numerical experiments demonstrate that, on the tested problems, SaLSPIA requires fewer iterations and less CPU time than LSPIA and several accelerated variants, while remaining robust for rank-deficient systems arising from missing regions.

Although the present analysis guarantees R-linear convergence, the faster convergence behavior observed in the numerical experiments has not yet been fully characterized theoretically. Future

work will extend SaLSPIA to T-spline and subdivision surface fitting and seek sharper theoretical convergence estimates.

Author contributions

Xingxuan Peng: Conceptualization, methodology, formal analysis, writing–original draft, writing–review & editing; Yutong Li: Methodology, software, validation, writing–original draft, writing–review & editing. All authors have read and agreed to the published version of the manuscript.

Use of Generative-AI tools declaration

The authors declare that no generative AI tools were used in the writing, analysis, preparation, or revision of this manuscript.

Acknowledgments

This work was supported by the Scientific Research Project of Liaoning Normal University (No. 20250097).

Data and code availability

The complete MATLAB R2021b implementation, including SaLSPIA and all comparison methods, the drivers for Tables 1–8, the computational results used in Figures 2–9, the input data used in the experiments, and the generated CSV/LaTeX tables containing CPU means and sample standard deviations, is publicly available in the GitHub repository (`https://github.com/Lily11898/SaLSPIA-Reproducibility`). No external non-bundled data are required to run the included experiments. The provenance and redistribution terms of each bundled data file are documented in `DATA_SOURCES.md` within the repository.

Conflict of interest

The authors declare no conflicts of interest.

References

1. L. Piegl, W. Tiller, *The NURBS Book*, Springer Berlin, Heidelberg, 1997. http://dx.doi.org/10.1007/978-3-642-59223-2
2. J. Hoschek, D. Lasser, *Fundamentals of computer aided geometric design*, A. K. Peters, Ltd., 1993.
3. Y. Zhang, P. Wang, F. Bao, X. Yao, C. Zhang, H. Lin, A single-image super-resolution method based on progressive-iterative approximation, *IEEE Trans. Multimedia*, **22** (2020), 1407–1422. http://dx.doi.org/10.1109/TMM.2019.2943750
4. A. Hashemian, S. F. Hosseini, An integrated fitting and fairing approach for object reconstruction using smooth NURBS curves and surfaces, *Comput. Math. Appl.*, **76** (2018), 1555–1575. http://dx.doi.org/10.1016/j.camwa.2018.07.007

5. C. Deng, H. Lin, Progressive and iterative approximation for least squares B-spline curve and surface fitting, *Comput.-Aided Des.*, **47** (2014), 32–44. http://dx.doi.org/10.1016/j.cad.2013.08.012
6. D. Qi, Z. Tian, Y. Zhang, J. B. Feng, The method of numeric polish in curve fitting, *Acta Math. Sin.*, **18** (1975), 173–184.
7. C. de Boor, How does Agee's smoothing method work? *Proceedings of the 1979 Army Numerical Analysis and Computers Conference*, ARO Report 79-3, 1979, 299–302.
8. H. Lin, G. Wang, C. Dong, Constructing iterative non-uniform B-spline curve and surface to fit data points, *Sci. China Ser.: Inf. Sci.*, **47** (2004), 315–331. http://dx.doi.org/10.1360/02yf0529
9. H. W. Lin, H. J. Bao, G. J. Wang, Totally positive bases and progressive iteration approximation, *Comput. Math. Appl.*, **50** (2005), 575–586. http://dx.doi.org/10.1016/j.camwa.2005.01.023
10. H. Lin, Q. Cao, X. Zhang, The convergence of least-squares progressive iterative approximation for singular least-squares fitting system, *J. Syst. Sci. Complex.*, **31** (2018), 1618–1632. http://dx.doi.org/10.1007/s11424-018-7443-y
11. H. Lin, Z. Zhang, An efficient method for fitting large data sets using T-splines, *SIAM J. Sci. Comput.*, **35** (2013), A3052–A3068. http://dx.doi.org/10.1137/120888569
12. Z. Chen, X. Luo, L. Tan, B. Ye, J. Chen, Progressive interpolation based on Catmull-Clark subdivision surfaces, *Comput. Graph. Forum*, **27** (2008), 1823–1827. http://dx.doi.org/10.1111/j.1467-8659.2008.01328.x
13. C. Deng, W. Ma, Weighted progressive interpolation of Loop subdivision surfaces, *Comput.-Aided Des.*, **44** (2012), 424–431. http://dx.doi.org/10.1016/j.cad.2011.12.001
14. Q. Chang, W. Ma, C. Deng, Constrained least square progressive and iterative approximation (CLSPIA) for B-spline curve and surface fitting, *Vis. Comput.*, **40** (2024), 4427–4439. http://dx.doi.org/10.1007/s00371-023-03090-8
15. L. Zhang, X. Ge, J. Tan, Least square geometric iterative fitting method for generalized B-spline curves with two different kinds of weights, *Vis. Comput.*, **32** (2016), 1109–1120. http://dx.doi.org/10.1007/s00371-015-1170-3
16. H. Lin, T. Maekawa, C. Deng, Survey on geometric iterative methods and their applications, *Comput.-Aided Des.*, **95** (2018), 40–51. http://dx.doi.org/10.1016/j.cad.2017.10.002
17. M. Liu, B. Li, Q. Guo, C. Zhu, P. Hu, Y. Shao, Progressive iterative approximation for regularized least square bivariate B-spline surface fitting, *J. Comput. Appl. Math.*, **327** (2018), 175–187. http://dx.doi.org/10.1016/j.cam.2017.06.013
18. C. Liu, X. Han, J. Li, Preconditioned progressive iterative approximation for triangular Bézier patches and its application, *J. Comput. Appl. Math.*, **366** (2020), 112389. http://dx.doi.org/10.1016/j.cam.2019.112389
19. A. Ebrahimi, G. B. Loghmani, A composite iterative procedure with fast convergence rate for the progressive-iteration approximation of curves, *J. Comput. Appl. Math.*, **359** (2019), 1–15. http://dx.doi.org/10.1016/j.cam.2019.03.025
20. S. Sajavičius, Hyperpower least squares progressive iterative approximation, *J. Comput. Appl. Math.*, **422** (2023), 114888. http://dx.doi.org/10.1016/j.cam.2022.114888
21. S. Channark, P. Kumam, J. Martínez-Moreno, W. Jirakitpuwapat, Hermitian and skew-Hermitian splitting method on a progressive-iterative approximation for least squares fitting, *AIMS Math.*, **7** (2022), 17570–17591. http://dx.doi.org/10.3934/math.2022967

22. N. C. Wu, C. Liu, Asynchronous progressive iterative approximation method for least squares fitting, *Comput. Aided Geom. Des.*, **111** (2024), 102295. http://dx.doi.org/10.1016/j.cagd.2024.102295
23. Y. F. Hamza, H. W. Lin, Conjugate-gradient progressive-iterative approximation for Loop and Catmull-Clark subdivision surface interpolation, *J. Comput. Sci. Technol.*, **37** (2022), 487–504. http://dx.doi.org/10.1007/s11390-020-0183-1
24. Z. D. Huang, H. D. Wang, On a progressive and iterative approximation method with memory for least square fitting, *Comput. Aided Geom. Des.*, **82** (2020), 101931. http://dx.doi.org/10.1016/j.cagd.2020.101931
25. C. Liu, N. C. Wu, J. Li, L. Hu, Two novel iterative approaches for improved LSPIA convergence, *Comput. Aided Geom. Des.*, **111** (2024), 102312. http://dx.doi.org/10.1016/j.cagd.2024.102312
26. Z. Yao, Q. Hu, Accelerated local progressive-iterative approximation methods for curve and surface fitting, *Vis. Comput.*, **41** (2025), 5979–5993. http://dx.doi.org/10.1007/s00371-024-03764-x
27. Z. Yao, Q. Hu, Efficient surface fitting via randomized Gauss-Seidel LSPIA: a novel iterative approach, *Vis. Comput.*, **42** (2026), 172. http://dx.doi.org/10.1007/s00371-025-04266-0
28. D. Rios, B. Jüttler, LSPIA, (stochastic) gradient descent, and parameter correction, *J. Comput. Appl. Math.*, **406** (2022), 113921. http://dx.doi.org/10.1016/j.cam.2021.113921
29. J. Barzilai, J. M. Borwein, Two-point step size gradient methods, *IMA J. Numer. Anal.*, **8** (1988), 141–148. http://dx.doi.org/10.1093/imanum/8.1.141
30. L. Grippo, F. Lampariello, S. Lucidi, A nonmonotone line search technique for Newton's method, *SIAM J. Numer. Anal.*, **23** (1986), 707–716. http://dx.doi.org/10.1137/0723046
31. X. Xu, A parameterized Barzilai-Borwein method via interpolated least squares, *J. Ind. Manag. Optim.*, **21** (2025), 6246–6269. http://dx.doi.org/10.3934/jimo.2025130
32. X. R. Li, Y. K. Huang, A note on R-linear convergence of nonmonotone gradient methods, *J. Oper. Res. Soc. China*, **13** (2025), 313–325. http://dx.doi.org/10.1007/s40305-023-00468-2
33. Y. H. Dai, L. Z. Liao, R-linear convergence of the Barzilai and Borwein gradient method, *IMA J. Numer. Anal.*, **22** (2002), 1–10. http://dx.doi.org/10.1093/imanum/22.1.1
34. Y. H. Dai, R. Fletcher, On the asymptotic behaviour of some new gradient methods, *Math. Program.*, **103** (2005), 541–559. http://dx.doi.org/10.1007/s10107-004-0516-9
35. H. Park, J. H. Lee, B-spline curve fitting based on adaptive curve refinement using dominant points, *Comput. Aided Des.*, **39** (2007), 439–451. http://dx.doi.org/10.1016/j.cad.2006.12.006
36. E. G. Birgin, J. M. Martínez, M. Raydan, Nonmonotone spectral projected gradient methods on convex sets, *SIAM J. Optim.*, **10** (2000), 1196–1211. http://dx.doi.org/10.1137/S1052623497330963

AIMS Press