Sequential B-Spline Surface Construction Using Multiresolution Data Clouds

B-spline surfaces are widely used in engineering practices as a flexible and efficient mathematical model for product design, analysis, and assessment. In this paper, we propose a new sequential B-spline surface construction procedure using multiresolution measurements. At each iterative step of the proposed procedure, we first update knots vectors based on bias and variance decomposition of the fitting error and then incorporate new data into the current surface approximation to fit the control points using Kalman filtering technique. The asymptotical convergence property of the proposed procedure is proved under the framework of sieves method. Using numerical case studies, the effectiveness of the method under finite sample is tested and demonstrated.

Keywords: B-spline surface, Kalman filter, sieves method, asymptotic convergence

1 Introduction
With the rapid development of computer aided design and engineering, surface construction, i.e., creating a closed-form mathematical model from discrete measurement points, has been widely used in various application areas, such as aerospace, biomedical, architecture, archaeology, film and entertainment industries, and consumer product industries, to facilitate product design, analysis, and assessment. Mathematically, surface construction can be viewed as fitting a function \( f(x,y) \) from the measured data cloud \( D \equiv \{(x_1,y_1,z_1), (x_2,y_2,z_2), \ldots, (x_n,y_n,z_n)\} \) to minimize the difference between \( f(x,y) \) and the underlying true surface \( g(x,y) \). In order to fit \( f(x,y) \), both the structure and the parameters of \( f(x,y) \) need to be determined. As of the structure, in computer aided design (CAD) and computer aided engineering, B-spline surface functions and their generalized form NURBS (Non-uniform rational B-spline) [1–4] are widely used due to their mathematical elegance and geometrical flexibility. Indeed, most available commercial CAD/CAM software packages adopt the B-Spline models as the basic mathematical tool to describe the design geometry. Thus, in this paper, we will focus on the problem of B-spline surface construction.

For an effective algorithm of B-spline surface construction, the following capabilities are highly desirable: (i) Capability of adjusting both the knots values and the control points. To completely determine a B-spline surface, both the knots values, which determine the basis functions used in the fitted function, and the control points, which determine the coefficients of the basis functions, are needed. Thus, with the capability of adjusting both, we can have the highest flexibility in the surface construction. However, knots values adjustment is generally a very challenging task and only limited works are available in the literature. (ii) Capability of dealing with multiresolution data. Due to the fast development of sensing technology, we can often obtain a huge amount of data with different resolutions. For example, in dimension control of manufacturing, it is now common to use the coordinate measure machine (CMM) with both mechanical probes and optical scanning probes. The CMM with mechanical probes can attain high-resolution and high-accuracy measurements but the operation is usually time consuming, while CMM with optical scanner can obtain large volume of data in a short time but the measurement is usually less accurate. To fully utilize the available data, it is highly desired that the surface construction algorithm can handle heterogeneous data. (iii) Capability of sequential updating. Different from the nonsequential algorithm (sometimes also called “offline” algorithm), which constructs the surface at once based on all the measurement data, the sequential algorithm starts from an initial surface and updates the surface model sequentially when extra measurements are available. Clearly, the sequential algorithm allows the change in sampling plan and is more flexible and computational efficient. In fact, the nonsequential algorithm can be viewed as a special case of the sequential algorithm.

According to the desired capabilities listed above, the available B-spline surface fitting algorithms, to the best of our knowledge, are presented in Table 1. It can be seen that currently there lacks a B-Spline surface construction method that possesses all the three desired capabilities. Furthermore, for the algorithms with the sequential updating capability, there usually lack the discussions on the convergence of the algorithm. To fill this gap, in this paper we propose a sequential B-spline surface construction framework using multiresolution measurement data, which cannot only update the control points but also adjust the knots vectors of the fitted B-spline surface model. The underlying theoretical foundation is the bias and variance decomposition of the fitting errors, of which the bias is mainly determined by the knots vectors and the variance is mainly determined by the control points. By updating both the control points and the knots vectors, we are aiming at reducing the overall fitting errors. The basic procedures of the proposed method are as follows: whenever new measurement data are available, we update the knots vectors first, followed by the control points updating using Kalman filter. The proposed method chooses knots vectors adaptively and sequentially, which provides a neat solution to the otherwise very difficult knots selection problem. By using the results of sieves method [15], we can analyze the asymptotic convergence property of the proposed B-spline surface construction procedure. Furthermore, by adopting Kalman filter to update the control points, our method can efficiently combine multiresolution data to get better fitting performance.

The remainder of this paper is organized as follows. Section 2 presents the problem formulation and a brief introduction to B-spline surface and Kalman filter. Section 3 presents the
proposed procedures for sequential B-spline surface construction, including the procedures for updating both the control points and the knots vectors, the convergence discussion, and the finite sample realization. Then, we illustrate the effectiveness of the proposed method through numerical studies and compare its performance with existing works in Sec. 4. Finally, we conclude our paper in Sec. 5.

2 Problem Formulation and Mathematical Backgrounds

2.1 Problem Formulation. Assume the underlying true surface is $g(x,y)$. The data cloud is denoted as $D = \{(x_1,y_1),z_1\}, \{(x_2,y_2),z_2\}, \ldots, \{(x_n,y_n),z_n\}$, where $z_i = g(x_i,y_i) + \xi_i$, with $\xi_i$ denoting the measurement error and assuming the expectation of $\xi_i$ is zero, i.e., $E[\xi] = 0$. The goal is to construct an approximation surface $f_D(x,y)$ based on the dataset $D$, which will perform well not only on $D$ but will also achieve good generalization properties, meaning that it will have small approximation error on locations that are not included in $D$. In this paper, we adopt the B-spline surface for the structure of $f_D$ and use the mean squared error function $H_D(x,y)$ to evaluate the approximation performance as

$$H_D(x,y) = E_z [(f_D(x,y) - z)^2] , \text{ where } f_D(x,y) \in \Pi(u,v,p,q) \tag{1}$$

where $E_z$ denotes the expectation with respect to the distribution of $z$, the subscript $D$ in $H_D(x,y)$ and $f_D(x,y)$ means they are dependent on dataset $D$. $\Pi(u,v,p,q)$ denotes the linear space of B-spline surfaces with knots vectors $u$ and $v$ and degree parameters $p$ and $q$. More detailed introduction to $\Pi(u,v,p,q)$ is presented in Sec. 2.2. Equation (1) can be decomposed into

$$H_D(x,y) = E_z [(z - g(x,y))^2] + E_D (f_D(x,y) - g(x,y))^2 \tag{2}$$

The first term in Eq. (2) does not depend on $D$ and it measures the amount of variance of $z$ caused by the measurement noise $\xi$ at location $(x,y)$. The second term is a random variable depending on $D$ since $f_D(x,y)$ is an approximation based on $D$. To evaluate the generalization property, we take the expectation on both sides of Eq. (2) with respect to all possible datasets $D$ of size $n$, and get

$$H(x,y) = E_z [(z - g(x,y))^2] + E_D (f_D(x,y) - g(x,y))^2 \tag{3}$$

$H(x,y)$ is called the integrated mean squared error function (IMSEF). To further analyze the approximation performance under IMSEF, we decompose $E_D (f_D(x,y) - g(x,y))^2$ to the bias and variance terms [16] as

$$E_D (f_D(x,y) - g(x,y))^2 = (E_D [f_D(x,y)] - g(x,y))^2 + E_D (f_D(x,y) - E_D [f_D(x,y)])^2 \tag{4}$$

The first term is the squared bias, the amount by which the average estimation $E_D (f_D(x,y))$ differs from the true surface. Since $E_D (f_D(x,y)) \in \Pi(u,v,p,q)$, the bias term essentially measures the distance between $g(x,y)$ and the space $\Pi(u,v,p,q)$, denoted as $\text{dist}(g(x,y), \Pi)$. As the complexity of $\Pi(u,v,p,q)$ grows, the bias term could approach zero for B-spline space [2]. The second term is the estimation variance, which will increase along the increase of the dimension of $\Pi(u,v,p,q)$ and decrease along with the increase of the sample size $n$ [17]. Intuitively, if $\Pi(u,v,p,q)$ is fixed, then the bias term in Eq. (4) is fixed. The second term in Eq. (4) can be reduced by using more measurement data points. On the other hand, if the measurement dataset $D$ is fixed, then reducing the bias term through the increase of complexity of $\Pi(u,v,p,q)$ will increase the variance term [17]. Thus, we need to strike a balance between the variance and the bias in surface construction.

In this paper, we propose a sequential B-spline surface construction method that allows the updates in both the $\Pi(u,v,p,q)$ and $D$ during the construction process. In this way, both the bias and variance term could decrease and hence the approximation performance improves. This method is sequential in the sense that we finish the surface construction in multiple iterative steps. At each step, we assume an additional measurement set is available. Specifically, let $D_k$ denote the dataset available at the $k$th iteration, where $D_k \supset D_{k-1}$ and the number of data points in $D_k$ is $n_k$. We first construct B-spline surface space $\Pi_k$ such that (i) $\Pi_k \supset \Pi_{k-1}$ and (ii) $\text{dist}(g(x,y), \Pi_k) \leq \text{dist}(g(x,y), \Pi_{k-1})$. And then, we adopt the Kalman filter technology to identify $f_D$ in $\Pi_k$ to minimize the integrated mean squared error

$$f_D(x,y) = \arg \min_{f \in \Pi_k} H(x,y)$$

$$= \arg \min_{f \in \Pi_k} \left\{E_z [(z - g(x,y))^2] + E_D (f(x,y) - g(x,y))^2 \right\}$$

(5)

Several important points need to be mentioned about the sequence $\{\Pi_k\}$. First, the $\Pi_k$ does not necessarily change at each iteration. The rule to decide whether to construct new $\Pi_k$ will be discussed in Sec. 3.3. Second, the change of $\Pi_k$ from $\Pi_{k-1}$ is achieved through the adjustment of the knots vectors $u$ and $v$, while the degree of the spline basis functions $p$ and $q$ are fixed. Third, if the true surface $g(x,y)$ is known, then the sequence $\{\Pi_k\}$ could be constructed with monotonically decreasing distance from $g(x,y)$. However, in reality, $g(x,y)$ is approximately known and only the upper bound of $\text{dist}(g(x,y), \Pi_k)$ could be estimated. Therefore, instead of reducing $\text{dist}(g(x,y), \Pi_k)$ directly, we try to reduce its upper bound. In summary, the objective of our sequential B-spline surface construction method could be decomposed into two tasks, namely constructing a sequence $\{\Pi_k\}$ with nonincreasing upper bound of the distance from the unknown true surface and constructing a sequence $\{f_{D_k}\}$ with decreasing approximation error. The details of these two tasks are discussed in Secs. 3.1 and 3.2.

<table>
<thead>
<tr>
<th>Control point fitting</th>
<th>Knot adjustment</th>
<th>Multiresolution data</th>
<th>Sequential updating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forsey’s hierarchical B-spline refinement (1988) [5]</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Piegl’s modifying algorithm (1989) [6]</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Sarkar’s optimization algorithm (1991) [7]</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Greiner’s algorithm (1996) [8]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Chen’s direct highline modification (1997) [9]</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lee’s BA algorithm (1997) [10]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Weiss’s advanced fitting algorithm (2002) [12]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Huang’s closed loop construction (2007) [13]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Huang’s Kalman filter algorithm (2009) [14]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

Table 1 Summary of existing B-spline surface fitting algorithms
respectively. Sections 2.2 and 2.3 provide a brief review of the theory and notations of B-spline surface and Kalman filter. Readers familiar with these materials can jump to Sec. 3 directly.

2.2 Introduction to the B-Spline Surfaces. A B-spline surface \( f(x,y) \) is defined as the tensor product of two B-spline curves along \( x \) and \( y \) directions, i.e.,

\[
f(x,y) = \sum_{i=0}^{r-p-1} \sum_{j=0}^{r-q-1} P_{ij} N_i(x) N_j(y)\]

where \( N_i(x) \) is the \( i \)th basis function of degree \( p \) defined on knots vector \( u = (u_0, u_1, \ldots, u_r) \) along \( x \) direction; \( N_j(y) \) is the \( j \)th basis function of degree \( q \) defined on knots vector \( v = (v_0, v_1, \ldots, v_s) \) along \( y \) direction; \( P_{ij} \) is the \( i \)th \( j \)th control point.

The set of all functions that can be expressed as Eq. (6) forms the linear B-spline surface space \( \Pi(u, v, p, q) \), which is formally defined as

\[
\Pi(u, v, p, q) := \left\{ \sum_{i=0}^{r-p-1} \sum_{j=0}^{r-q-1} P_{ij} N_i(x) N_j(y), P_{ij} \in \mathbb{R}, \forall i, j \right\}
\]

The complexity of \( \Pi(u, v, p, q) \) depends on the four parameters \( u, v, p, q \). Increasing \( p \) or \( q \) will increase the dimension of basis functions and finer \( u \) or \( v \) increase the dimension of surface space. In general, we can fit any continuous surface to any degree of accuracy by either increasing the \( p \) or \( q \), or by enlarging \( u \) and \( v \) [2]. Conventionally, the range for \( x \) and \( y \) is normalized to \( [0, 1] \) and in the following, we use this default setting for all analysis.

2.3 Introduction to the Kalman Filter. Kalman filter has been widely used to obtain statistically optimal estimates of the internal states of a system based on the measurements of the system’s external behavior. More precisely, the internal states of a system are a collection of variables and are assumed to evolve in the following way:

\[
x_k = F_k x_{k-1} + B_k u_k + w_k, w_k \sim N(0, Q_k)
\]

where \( x_k, x_{k-1} \) are the true states at time \( k \) and \( k-1 \), \( u_k \) is the system input and \( w_k \) is the process noise. At time \( k \), the measurement \( z_k \) of the true state \( x_k \) is

\[
z_k = H_k x_k + v_k, v_k \sim N(0, R_k)
\]

When the new measurement \( z_k \) is available, Kalman filter recursively updates the estimates of internal states by infusing new measurements using the following recursive equations

\[
x_{k+1} = (I - K_k H_k) x_k + (I - K_k H_k) B_k u_k + K_k z_k
\]

\[
\Delta x_{k+1} = (I - K_k H_k) F_k \Delta x_k + F_k T_k + (I - K_k H_k) Q_k
\]

where \( I \) is the identity matrix, \( \Delta x_k \) is the error covariance matrix of \( x_k \), i.e., \( \Delta x_k = \text{cov}(x_k - \bar{x}_k) \), and \( K_k \) is the optimal Kalman gain that yields minimum mean square error estimator of the internal state \( x_k \)

\[
K_k = F_k \Delta x_k - F_k H_k S_k^{-1} Q_k H_k^T + R_k
\]

\[
S_k = H_k (F_k \Delta x_k - F_k Q_k) H_k^T + R_k
\]

Using Kalman filter, both the internal state estimator and its variance \( \Delta x_k \) are updated incrementally in the sense that only the information in the new observation which cannot be counted for in the previous estimates is incorporated.

![Fig. 1. Basic flow of the sequential B-spline surface construction](image)

3 Sequential B-Spline Surface Construction

This section presents the details of our proposed methodology and analyzes its convergence property. The overall flow of the sequential B-spline surface construction is shown in Fig. 1.

Sections 3.1 and 3.2 introduce how to construct sequence \( \{\Pi_i\} \) and \( \{f_n\} \), respectively; Sec. 3.3 addresses the convergence issues of the proposed method and shows a finite sample realization of the proposed method.

3.1 Constructing \( \{\Pi_i\} \) Sequence Based on Refining Knots Vectors. According to Sec. 2.2, given \( p \) and \( q \), the B-spline surface space \( \Pi \) is uniquely determined by knots vectors \( u \) and \( v \). Therefore constructing sequence \( \{\Pi_i\} \) is equivalent to construct sequences of knots vectors \( \{u_i, v_i\} \). As mentioned in Sec. 2.1, the constructed sequence \( \{\Pi_i\} \) should satisfy two conditions, namely, (i) \( \{\Pi_i\} \supseteq \{\Pi_{i-1}\} \) and (ii) the upper bound of distance \( g(x, y) \), \( \Pi_k \) is nonincreasing. We need to transform these two conditions into equivalent requirements for the knots vectors \( \{u_i, v_i\} \). For the first condition, it is straightforward because we have \( \Pi_k \supseteq \Pi_{k-1} \Rightarrow (u_i \supseteq u_{i-1}, v_i \supseteq v_{i-1}) \) due to the knots insertion property of B-splines [2]. For the second condition, we need to understand first how the knots vectors influence the distance. According to De Boor [18], for univariate case, given knots vector \( u = (u_0, u_1, \ldots, u_r) \) with \( u_0 = \cdots = u_0 = 0 < u_{p+1} \leq \cdots \leq u_{r-p} = \cdots = u_r \), for any function \( h \in C^{k+1}(k = 0, 1, \ldots, p+1) (C^{k+1}) \) is the class of functions having \( k \) continuous derivatives), the distance between \( h \) and the B-Spline curve space \( \Pi(u, p, q) \) in the knot interval \( [u_i, u_{i+1}], (i = p, p+1, \ldots, r-p-1) \), is bounded from above by

\[
\text{dist}(h, \Pi(u, p)) \leq \text{const}_{p,k} \|D^k h\|_{L_2}
\]

where \( \text{const}_{p,k} \) is a constant related with \( p \) and \( k \); \( I_j \) is the interval \([u_i - k + 2; u_i + k + 1]\) with its length \( |I_j| \); \( |I_j| \) is the interval \([v_j - q + 1; v_j + q - 1]\) with its length \( |I_j| \). The proof of this theorem is straightforward, we omit the details here. For readers interested, please refer to De Boor [2] and Plienik [19].

According to this theorem, we can reduce \( |I_j| \) or \( |I_j| \) by adding more knots in areas where \( \|D^q x\| \) or \( \|D^q y\| \) is relatively large. The procedures for knots vectors refinement are as follows
In general, the variance will increase as the dimension of the space. In theory, the dimension of B-spline surface space should be accurate enough when they are used to determine where to add knots. Eq. (14) can only be estimated, it does not reduce the effective, dimension of the space $D_f$. We will like to point out that although the bound of distance in functional space is better. Otherwise small the largest extent. In practice, we can set $\alpha = 0.5$, which results in satisfying results without overwhelming optimization computations.

**Knots vectors refinement algorithm:**

S0: Calculate $\sup \left( |U^i| \left| \partial_{x^i} f_{D_0} \right| + |U^j| \left| \partial_{y^j} f_{D_0} \right| \right)_{i,j\in k}$ for each area $[u_i, u_{i+1}] \times [v_j, v_{j+1}]$ (i.e., $p, p + 1, ..., p + \gamma - 1; q, q + 1, ..., q + \gamma - 1$).

S1: Identify the area $[u_i, u_{i+1}] \times [v_j, v_{j+1}]$ where $\sup \left( |U^i| \left| \partial_{x^i} f_{D_0} \right| + |U^j| \left| \partial_{y^j} f_{D_0} \right| \right)_{i,j\in k}$ has the largest value.

S2: Check whether the dimension of B-spline surface after adding one more knot (either in u or v direction) exceeds M?

S2.1: If no, insert $\bar{u} = \lambda_1 u_i + (1 - \lambda_1) u_{i+1}$ or $\bar{v} = \lambda_2 v_j + (1 - \lambda_2) v_{j+1}$ whichever has a larger knot span, i.e., if $|U^i|$ is larger than $|U^j|$, then insert $\bar{u}$, otherwise insert $\bar{v}$.

S2.2: If yes, keep the vectors unchanged.

END

where $\phi_{r+1/2} = \partial f_{D_0}$ on $[u_i, u_{i+1}]$ and define the quotient 0/0 to be zero. The estimation of $|\partial_{x^i} f_{D_0}|$ can follow the same way. We would like to point out that although the bound of distance in Eq. (14) can only be estimated, it does not reduce the effectiveness of the proposed knots refinement scheme because we only need to identify the regions that have relatively larger error where we can add more knots. In fact, it is known that the approximation accuracy of the B-spline surface is not significantly influenced by the precise locations of the knots, rather, it is influenced by the distributions of knots [2]. As a result, we do not need to know the precise values of these approximation errors. Even though the bounds are estimated based on the fitted surface, their values shall be accurate enough when they are used to determine where to add the new knots. Furthermore, the fitted surface is estimated in the whole surface region using all the available data, and hence the impact of local measurement noise on the bound estimation can be significantly reduced in this method.

The second issue is how to determine the parameter $M$. Parameter $M$ is a parameter to control the dimension of the B-spline surface space. In theory, the dimension of B-spline surface space should increase such that the fitted surface could approach the true surface (i.e., the bias approaches zero). However, as mentioned in Sec. 2.1, the variance will increase as the dimension of $P$ increases and the increase of variance might surpass the reduction of bias due to knots insertion. Therefore, we need to exercise more caution about whether to add knots. Typically bigger $M$ will result in higher dimension and thus smaller bias but larger variance, and vice versa. When the size of dataset is large and the true surface is relatively smooth, then big $M$ is better. Otherwise small $M$ is preferable. More detailed analysis of this issue will be presented in Sec. 3.3.

The third issue is to determine the parameters $\lambda_1$ and $\lambda_2$. These two parameters decide the locations of newly added knots. They could be optimized to reduce the approximation error to the largest extent. In practice, we can set $\lambda_1 = 0.5$, which results in satisfying results without overwhelming optimization computations.

### 3.2 Constructing $f_{D_0}$

As illustrated in Fig. 1, after the functional space $\Pi_0$ has been updated at the $k$th iteration step, the B-spline surface $f_{D_k}$ needs to be constructed. Because $\Pi_k$ is already determined, updating $f_{D_k}$ is equivalent to updating the control points.

This problem can be formulated as: Given initial state $f_{D_0}(x, y) = \arg \min_{f \in \Pi_0} H(x, y)$, find $f_{D_k}(x, y) = \arg \min_{f \in \Pi_k} H(x, y)$ based on the enlarged dataset $D_k$. If $\Pi_k$ is the same as $\Pi_{k-1}$, then the above problem degrades to the problem of constructing the B-spline surface for sequentially collected data in a fixed functional space $\Pi(u, v, p, q)$ (i.e., $\Pi_0$). The solution to this problem has already been established by using Kalman filter method [15]. In this Kalman filter method, the control points are defined as the internal states of the Kalman filter and the state space model is given as

$$P_k = P_{k-1}$$

$$z_k(x, y) = N^*_u(x)P_{k-1}N^*_v(y) + v_k, v_k \sim N(0, \sigma^2_k)$$

where $P_k$ is the control points matrix at the $k$th step with $P_{k-1}$ as its $(i, j)$th entry and $z_k$ is the measurement data at the $k$th step on location $(x, y)$, $v_k$ is the measurement error with standard deviation $\sigma^2_k$. Given B-spline surface space $\Pi(u, v, p, q)$, $N^*_u(x)$, and $N^*_v(y)$ are basis vectors evaluated at $x$ and $y$, respectively, i.e.,

\[ N^*_u(x) = \left[ N_0(u, x), N_1(u, x), ..., N_{p-1}(u, x) \right] \text{ and } N^*_v(y) = \left[ N_0(v, y), N_1(v, y), ..., N_{q-1}(v, y) \right]. \]

The model in Eqs. (16) and (17) is a linear model regarding the internal states $P_k$ and thus it can be easily rewritten into the format of Eqs. (8) and (9). Using the relations in Eqs. (10) and (11), we can incorporate the new data to update the control points. Together with the given fixed space $\Pi(u, v, p, q)$, we can then obtain the updated B-spline surface $f$. This Kalman filter procedure is an optimal procedure capable of providing minimum variance unbiased estimators of the control points. If $\Pi_k$ is different from $\Pi_{k-1}$, then the Kalman filter method cannot be applied directly. In this situation, $f_{D_k}$, and the associated control points $P_{k-1}$ are not the optimal solution in $\Pi_k$ that minimizes the integrated mean squared error based on dataset $D_{k-1}$ (although they are optimal in $\Pi_{k-1}$). Therefore, Kalman updating based on $P_{k-1}$ cannot guarantee to get the optimal $P_k$ in $\Pi_k$ under the enlarged dataset $D_k$. Instead, we need to first find $f_{D_{k-1}}$ that...
\( f_{D_{k-1}}(x, y) = \arg \min_{f \in \Pi_k} \left\{ E[z - g(x, y)]^2 + E_{D_{k-1}}(f(x, y) - g(x, y))^2 \right\} \)

(18)

and then extract the associated control points \( P'_{k-1} \) to be used in the Kalman updating step. In this research, we propose to use \( f_{D_{k-1}} \) as the initial solution and apply a numerical optimization method, Levenberg–Marquardt algorithm [20], to obtain the solution for \( f_{D_{k-1}} \). As we have discussed in Sec. 2.1, the objective function of Eq. (18) can be decomposed into three parts as follows:

\[
E[z - g(x, y)]^2 + (E_{D_{k-1}}(f(x, y) - g(x, y))^2 + E_{D_{k-1}}(f(x, y) - E_{D_{k-1}}(f(x, y)))^2
\]

(19)

The first two terms are fixed and the only term left to be minimized in Eq. (19) would be the variance term. Because \( \Pi_k \) is linear in terms of control points, the problem of solving \( f_{D_{k-1}} \) becomes the typical least squares fitting problem. As proved in Nocedal and Wright [20], Levenberg–Marquardt algorithm converges to the optimal solution.

To summarize, at the \( k \)th step, if \( \Pi_k \) is the same as \( \Pi_{k-1} \), then the optimal control points \( P_k \) and the corresponding B-Spline surface \( f_k \) can be obtained through Kalman updating from \( P_{k-1} \); otherwise, extra optimization step is need to adjust \( P_{k-1} \) into \( P_k \) and then \( P_k \) can be obtained through Kalman updating from \( P_{k-1} \).

3.3 Convergence Analysis and Finite Sample Realization. The above proposed procedure is a sequential procedure. It is natural to ask if it is possible that \( f_k \) converges to the true underlying function \( g(x, y) \) as sample size \( n_k \to \infty \) and if yes, what are the conditions needed. We shall address these questions in this section.

The proposed method actually follows the basic principle of “method of sieves” introduced by Grenander [15]. In statistics, sieves method is a class of nonparametric methods which progressively use more complex models to estimate an unknown function as more data are collected. Particularly for B-spline basis functions, Kohler [21] established the general convergence theory of least square estimation over data-dependent multivariate B-spline spaces. Therefore, we can utilize their results to analyze the convergence property of our proposed method. The general convergence theorem is stated as follows:

**Theorem 2.** Let \( \Pi_k(u, v, p, q) \) denotes B-spline surface space with knots vectors \( u_k = (u_0, u_1, \ldots, u_p) \) and \( v_k = (v_0, v_1, \ldots, v_q) \) obtained based on the measurement dataset \( D_k \) of an underlying true surface function \( g(x, y) \) and \( n_k \) is the number of data points in \( D_k \). Assume \( f_{D_k}(x, y) = \arg \min_{f \in \Pi_k} H(x, y) = \sum_{i=0}^{p-1} \sum_{j=0}^{q-1} P_{i,j}(D_{k,0} \times \mathbb{N}(x) \times \mathbb{N}(y), v_k, u_k, v_{sk}) \) and truncated in the range of \( [-\beta_k, \beta_k] \), then a sequence \( f_{D_k}(x, y) \) converges to \( g(x, y) \) almost surely if the following conditions are satisfied

\begin{align}
(i) & \quad \beta_k \to \infty \quad n_k \to \infty \\
(ii) & \quad \beta_k \cdot n_k \cdot (\beta_k^4 | n_k^{(1-\delta)}) \to 0 \quad n_k \to \infty \quad \text{for some} \quad \delta > 0 \\
(iii) & \quad \bigcup_{i=0,1} \left\{ [u_i, u_{i+1}] \cap [-L, L] \right\} \to \emptyset \\
(iv) & \quad \bigcup_{i=0,1} \left\{ [v_{i,j}, v_{i,j+1}] \cap [-L, L] \right\} \to \emptyset \quad n_k \to \infty \\
& \quad \text{for every} \quad L, \gamma > 0
\end{align}

(20)

(21)

(22)

(23)

The proof of this general theorem can be found in Ref. [21].

Intuitively, conditions (i) and (iii) require the surface to be bounded, which is always the case when fitting real surfaces. Therefore, we will not discuss them in detail. Condition (ii) limits the increasing rate of the knots as compared with the increasing rate of the sample size. The intuition of this requirement is to make sure the sample size is large enough compared to the number of variable such that the variance term in the integrated mean squared error is small. Condition (iv) regulates the way of refining the knots. Roughly speaking, it requires that the intervals between every two knots go to zero as the sample size goes to infinity.

According to Theorem 2, for our procedure to be converging, we need to limit the knots increasing rate to satisfy the condition (ii) and let the recursive knots refinement satisfy the condition (iv). For condition (ii), we could add knots only when this condition is satisfied. For condition (iv), we could continuously refine knots vectors, as \( n_k \to \infty \) and then both \( r_k \) and \( s_k \to \infty \). In addition, each time refining knots, we reduce the size of the largest knots interval according to Eq. (14). Therefore, as \( r_k \) and \( s_k \to \infty \), the intervals between any two knots will go to zero and the proposed procedure will satisfy condition (iv). Under the guidance of this general theory, we developed a realization of our proposed procedure which satisfies the above convergence conditions as \( n_k \to \infty \) and also yields good approximation under finite sample size. The pseudo algorithm is as follows:

**Sequential B-spline surface construction algorithm:**

**Case 1:** Set \( u_0 = (u_0, u_1, \ldots, u_{2p+1}) \) with \( u_0 = \ldots = u_0 = 0 \), \( u_0 + 1 = 0.5 \), \( u_{p} \) \(= \ldots = u_{p} = 1 \), \( v_0 = (v_0, v_1, \ldots, v_{2q+1}) \) with \( v_0 = \ldots = v_0 = 0 \), \( v_0 + 1 = 0.5 \), \( v_{q} \) \(= \ldots = v_{q} = 1 \). Using either Kalman filter or weighted least square, fit the initial B-spline surface approximation \( f_0 \) over initial dataset \( D_0 \) and \( n_0 \) is the number of data points in \( D_0 \).

**Case 2:** Run Levenberg–Marquardt algorithm with knots vectors \( u_0 + 1 \) and \( v_0 \) as the initial state, run Kalman filter to incorporate the new data to get new surface approximation \( f_{D_1} \). Go to S2.7.

**Case 2.7:** Using \( f_{D_1} \) as the initial state, run Kalman filter to incorporate the new data to get new surface approximation \( f_{D_1} \). Go to S2.8.

**Case 2.8:** Truncate \( f_{D_1} \) to the range of \( (\ldots (N_2(n_{k+1} - 0.1)(N_2(n_{k+1} - 0.1)(r_{k+1} - p - 1) - (s_{k+1} - q - 1)), (s_{k+1} - q - 1) - (r_{k+1} - p - 1)) \).

**Case 2.9:** \( k = k + 1 \), go to S2.1.

**Case 2.2:** Obtain the original dataset from \( D_{k+1} \) by including the new data.

**Case 2.3:** Identify the area \( [u_i, u_{i+1}] \times [v_j, v_{j+1}] \) where Eq. (14) has the largest bound. Check whether the dimension after updating \( u_i \) or \( v_j \), which is \((r_{k-1}-p)(s_{k-1}-q) - 1 \) or \((r_{k-1}-p)-1(s_{k-1}-q) - 1 \) exceeds \( N_1(n_{k+1} - 0.1) \) and \( (r_{k-1}-p)-1(s_{k-1}-q) - 1 \) \(< N_1(n_{k+1} - 0.1) \). Insert \( v_{sk} = 0.5(v_{sk} + 1) \) and update \( u_i \) \(= \) \( (u_0, u_1, \ldots, u_i, u_{i+1}, \ldots, u_p) \) and update \( u_i \) \(= \) \( (u_0, u_1, \ldots, u_i, u_{i+1}, \ldots, u_p) \) and \( v_{sk} \) \(= \) \( (v_0, v_1, \ldots, v_{sk}, v_{sk+1}, \ldots, v_q) \).

**Case 3:** Run Levenberg–Marquardt algorithm with knots vectors \( u_i + 1 \) and \( v_j + 1 \) on dataset \( D_{k+1} \) to find \( f_{D_{k+1}} \).

**Case 3.2:** Using \( f_{D_{k+1}} \) as the initial state, run Kalman filter to incorporate the new data to get new surface approximation \( f_{D_{k+1}} \). Go to S2.8.
In the above realization, to control the increase rate of the dimension of B-spline surface space, we only refine knots if the dimensions of the knots vectors after knots refining satisfy the constraints stated in §2.4, where $N_1$ is a fixed large number. This rule makes the procedure satisfy the requirement of Theorem 2 condition (ii) and guarantees good approximation performance when sample size is small. At the last step, after using Kalman filter updating, we truncate the surface approximation $f_{	ext{Kal}}$ such that the surface follows in $[-N_2/(n_k+1)/((r_k+1-p-1)\cdot (x_k+1-q-1)), N_2/(n_k+1)/((r_k+1-p-1)\cdot (x_k+1-q-1))]$. This rule makes the procedure satisfy the requirement of Theorem 2 condition (iii). Again, $N_2$ is a fixed large number that guarantees good approximation performance when sample size is small.

4 Numerical Study

In the numerical study, cubic B-spline surface ($p = 3, q = 3$) are used. Both $N_1$ and $N_2$ are fixed at $10^3$. Four representative surfaces are used to show the effectiveness of our method. In Sec. 4.1, we use the Franke’s test surface to illustrate why knots updating is necessary and important. In Sec. 4.2, we apply our method on a bivariate Gaussian probability density function to demonstrate the adaptive knots allocation property of our proposed method. In Secs. 4.3 and 4.4, we compared the approximation performance of our method with De Boor’s [2] surface fitting method using optimally distributed knots vectors on both smooth surfaces and surfaces with sharp local changes. In all simulation studies, multi-resolution data are used for surface construction. For the initial sample, the measurement error has a larger standard deviation, while for the following sequential samples, the deviation is halved. Since the true surface $g$ is known in all these sections, we evaluate the root mean squared error (RMSE), defined as

$$\sqrt{\frac{1}{m} \sum_{i=1}^{m} (g(x_i,y_i) - f_{\text{Kal}}(x_i,y_i))^2},$$

to measure the surface fitting performance.

4.1 Updating Knots for Franke’s Test Surface. The Franke’s test function is defined as

$$g(x,y) = \frac{3}{4} e^{-((9x-2)^2+(9y-2)^2)/4} + \frac{3}{4} e^{-((9x+1)^2/49-(9y+1)^2/10)} + \frac{1}{2} e^{-((9x-7)^2+(9y-3)^2)/4} - \frac{1}{5} e^{-(9x-4)^2-(9y-7)^2}$$

(24)

For better visual quality, we enlarge the original Franke’s function, i.e., we use 10 $g(x,y)$ as our test function here. Initially, we sampled data points from a uniformly distributed $11 \times 11$ grid in the range of $[0,1] \times [0,1]$. The corresponding measurement error $\xi$ is normally distributed with standard deviation of 1. To illustrate the importance of knots updating, we compare the approximation performance of our method with $u_0=(0,0,0,0,0.5,1,1,1,1)$ and $v_0=(0,0,0,0,0.5,1,1,1,1)$ and Huang’s method [14]. In Huang’s method, it only updates the control points of the surface when new data arrive. However, the knots vectors are determined at the very beginning without subsequent updates. For a fair comparison, we start Huang’s method with the same initial knots vectors and provide both approaches with the same sequentially sampled data. We ran both methods for 20 iterations, and at each iteration we added 20 randomly sampled new data points. The measurement errors of these new data points are normally distributed with a smaller standard deviation, i.e., $0.5$.

Figure 2 shows the true surface and initial noisy data points measured from the true surface. Figure 3 shows the initial fitting and the error surface of the initial fitting. Figure 4 shows the final approximation of our method and Huang’s method [14] after 20 iterations. The knots vectors of our method after 20 iterations are $u = (0,0,0,0,0.25,0.5,0.75,1,1,1,1)$ and $v = (0,0,0,0.125,0.25,$
0.375, 0.5, 0.75, 1, 1, 1), while Huang’s method cannot update knots and hence the knots vectors remain as the initial ones. To evaluate the approximation performance, the RMSE of both approximations in Fig. 4 are also calculated. We select a uniformly distributed $101 \times 101$ grid as evaluation locations. Therefore, the RMSEs are 

\[
\text{RMSE} = \sqrt{\frac{1}{101 \times 101} \sum_{i=1}^{101} \sum_{j=1}^{101} (g(x_i, y_j) - f_{20}(x_i, y_j))^2},
\]

where $g(x, y)$ is the true surface, $f_{20}(x, y)$ is the approximation surface after 20 iterations.

The calculated RMSE for our method is 0.1486, while the RMSE for Huang’s method is 0.3728. The improvement is 60.13%. From both the RMSE comparison and the visual illustrations in Fig. 4, we can conclude that updating knots is necessary and beneficial, especially when the initial information about the unknown surface is insufficient to determine good knots locations. Moreover, we can also see that the distributions of knots vectors determined by our method are consistent with the true surface structures, namely, the locations of knots are close to the local maximum or local minimum of the true surface. According to De Boor [2], who states that more knots should be placed at the places where the true function has larger high order derivatives, our knots updating procedure can successfully achieve this goal. In all, this case shows that our proposed method can not only incorporate new data points to update the control points but also place knots adaptively and efficiently when the true surface is unknown.

Fig. 4 Final approximation surfaces after 20 iterations, proposed method (left), Huang’s method (right)

Fig. 5 True surface and noisy initial sample (represented as “+” symbols) (left), top view of the surface (right)

Fig. 6 Final fitting of the proposed method
The measurement errors of these new data points are normally distributed with halved standard deviation, i.e., 0.5. The final knots vectors are as shown in Eq. (26). Compared with Fig. 5 (right), it is clear that our method can allocate knots at the locations where sharp changes take place and hence need more knots than smooth areas.

\[
\mathbf{u} = (0,0,0,0,0.125,0.1875,0.25,0.3125,0.375,0.5,0.75,1,1,1,1) \\
\mathbf{v} = (0,0,0,0,0.125,0.25,0.375,0.5,0.75,1,1,1,1)
\]

(26)

Since the probability density function should be non-negative, we truncate the negative part of the final fitting. The result is shown in Fig. 6. Except some small wiggles around the boundary areas, our method provides a satisfying approximation of the true surface.

### 4.3 Comparison With De Boor’s Surface Fitting Method on a Smooth Surface

In Secs. 4.3 and 4.4, we compare the approximation performance of our method with De Boor’s method. In De Boor’s method, the optimal knots locations are determined according to the initial sample data points without further updating, and the new data are incorporated into the current surface approximation with knots vectors fixed. Only the initial sample data points are used in De Boor’s method for optimal knots allocation because that method highly depends on the number and locations of the sample points to allocate the knots. Thus, if a very large dataset is used for knot allocation, we will end up with very high dimensions of knots vectors. This will cause overfitting as well as computational inefficiency. However, in the proposed method, the dimensions of knots are well controlled even for large dataset.

First, we compare the two methods on a relatively smooth surface, defined as follows:

\[
g(x,y) = 10 \cdot \sin(2(x - 0.5) + 2(y - 0.5)) + 10 \cdot \cos(2(x - 0.5) - 6(y - 0.5))
\]

(27)

Initially, we sample data points from a uniformly distributed 11 \times 11 grid in the range of [0, 1] \times [0, 0.3]. This also corresponds to the sharp peaks of the surface in Fig. 5 (left).

The initial knots vectors are \(\mathbf{u}_0 = (0,0,0,0,0.5,1,1,1,1)\) and \(\mathbf{v}_0 = (0,0,0,0,0.5,1,1,1,1)\). We run our algorithm for 40 iterations and at each iteration we added 20 randomly sampled new data points. The measurement errors of these new data points are

\[
\begin{align*}
\sigma_x &= [0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.10] \\
\sigma_y &= [0.00, 0.11, 0.22, 0.33, 0.44, 0.55, 0.66, 0.77, 0.88, 0.99]
\end{align*}
\]

\[
\sigma_z = 0.1
\]

The initial knots are optimized according to the initial noisy data points.

The corresponding measurement error \(\xi\) is normally distributed with standard deviation 1. Figure 5 (right) is a top view of true surface, which shows the location where density concentrates and the variance direction. As we can see, the density mainly spreads out in the range of \([0, 0.5]\) \times \([0, 0.3]\). This also corresponds to the sharp peaks of the surface in Fig. 5 (left).

The initial knots vectors are \(\mathbf{u}_0 = (0,0,0,0,0.5,1,1,1,1)\) and \(\mathbf{v}_0 = (0,0,0,0,0.5,1,1,1,1)\). We run our algorithm for 40 iterations and at each iteration we added 20 randomly sampled new data points. The measurement errors of these new data points are normally distributed with halved standard deviation, i.e., 0.5. The final knots vectors are as shown in Eq. (26). Compared with Fig. 5 (right), it is clear that our method can allocate knots at the locations where sharp changes take place and hence need more knots than smooth areas.

\[
\mathbf{u} = (0,0,0,0,0.125,0.1875,0.25,0.3125,0.375,0.5,0.75,1,1,1,1) \\
\mathbf{v} = (0,0,0,0,0.125,0.25,0.375,0.5,0.75,1,1,1,1)
\]

(26)

Since the probability density function should be non-negative, we truncate the negative part of the final fitting. The result is shown in Fig. 6. Except some small wiggles around the boundary areas, our method provides a satisfying approximation of the true surface.

![Fig. 7 Smooth surface. “+” symbols stand for initial noisy measurement data.](image)

![Fig. 8 Initial fitting of two methods: proposed method (left), De Boor’s method (right).](image)
and update them sequentially. Figure 8 shows the initial fitting performance of both methods. As we can see the initial fitting of our method underfits the data, since it does not fully capture the curvature structures of the true surface. The initial fitting of De Boor’s method overfits the data and it has many unnecessary wrinkles. As mentioned before, De Boor’s knots allocation method highly depends on the number and locations of the sample points. As we can see from this case study, if the initial sampled data
have the same distribution along both \( u \) and \( v \) direction (which is very common for sampling scheme), the optimized knots along \( u \) and \( v \) direction are also the same. However, in many cases, the surface structures along these two directions are quite different and require different knots vectors settings.

Similar to Sec. 4.1, we ran our method for 20 iterations and randomly sampled 20 new data points at each iteration. The measurement errors for newly sampled data points are normally distributed with standard deviation 0.5 (half of the original noise deviation). The final knots vectors determined by our method are \( u = (0, 0.0, 0.25, 0.5, 0.625, 0.75, 0.875, 1, 1, 1, 1) \) and \( v = (0, 0.0, 0.25, 0.5, 0.75, 1, 1, 1, 1) \). The knots along \( u \) direction are denser than that of \( v \) direction, which is consistent with the changes of the true surface along these two directions. Figure 9 shows the final approximated surface of our method and De Boor’s method.

To clearly illustrate the approximation performances, we also plot the initial error surface and the final error surface for both our method and De Boor’s method in Figs. 10 and 11, respectively. Their RMSEs are also calculated based on a 101 \( \times \) 101 uniformly distributed grid. The results are summarized in Table 2. Comparing the final RMSE results of two methods, we can find out that our proposed method is 31.8% better than De Boor’s method. It is worth mentioning again that in De Boor’s method we only use the initial dataset rather than the whole dataset to determine the optimal knots locations. This is because when the dataset is large, the number of optimal knots selected by De Boor’s method is also very large. In this case study, the optimal knots vectors obtained using all the data after 20 iterations have 201 unique values along both \( u \) direction and \( v \) direction. This extremely high dimension could cause overfitting and computational issues for surface construction.

4.4 Comparison With De Boor’s Surface Fitting Method on the Sinc Surface. To evaluate the performance of our method comprehensively, we also compared with De Boor’s method on a surface with abrupt local changes. One challenge of fitting this kind of surface is that the knots vectors should be located appropriately with denser knots around rough areas and sparse knots around smooth areas. Otherwise, the approximation performance of the fitted surface would vary significantly across the whole surface. Since our proposed method updates the knots vectors dynamically according to the approximation error, it could adaptively locate knots. To illustrate the adaptive property of our method, we choose the following surface, which is defined by a bivariate sinc function as follows:

\[
g(x, y) = \frac{\sin \left( \sqrt{\left(16(x - 0.5)\right)^2 + \left(16(y - 0.5)\right)^2 + 2^{-52}} \right)}{\sqrt{\left(16(x - 0.5)\right)^2 + \left(16(y - 0.5)\right)^2 + 2^{-52}}} \tag{28}
\]

The small value \( 2^{-52} \) added in both denominator and numerator is to avoid the indeterminate 0/0 at location (0.5, 0.5). Figure 12 shows the true surface and the initial sampled data. For the initial sample, we sampled data points from a uniformly distributed 15 \( \times \) 15 grid in the range of \([0, 1] \times [0, 1]\). The corresponding measurement error \( \xi \) is still normally distributed but with standard deviation 0.1 (approximately 10% of the biggest function value). As we can see from Fig. 12, the surface has a belt shaped depression and achieves its peak value at \([0.5, 0.5]\); while, at the four corners, the surface is relatively smooth and flat. Moreover, because the surface is symmetric along both \( u \) direction and \( v \) direction, we should also expect symmetric properties for knots vectors along these two directions.

The optimized knots vectors selected by De Boor’s method are \( u = v = (0, 0.0, 0.138, 0.213, 0.285, 0.357, 0.429, 0.500, 0.571, 0.643, 0.715, 0.787, 0.862, 1, 1, 1, 1) \). For our proposed method, we

\[\begin{array}{|c|c|c|}
\hline
\text{Initial RMSE} & \text{Final RMSE} \\
\hline
\text{Proposed method} & 0.5818 & 0.1035 \\
\text{De Boor’s method} & 0.9499 & 0.1252 \\
\hline
\end{array}\]

Fig. 12 The sinc surface. “+” symbols stand for initial noisy measurement data.
use the same set up as that in Sec. 4.3. Figure 13 shows the initial fitting performance of both methods. From Fig. 13, two phenomena could be observed: first, due to the dimension deficiency, our proposed method does not approximate the true surface well initially; second, the approximation performance of the initial fitting of De Boor’s method is not uniform, i.e., in some areas, it underfits the true surface, while in some areas, it overfits the true surface. This is due to the limitation of De Boor’s method, which chooses knots vectors mainly based on sampling schemes rather than the approximation performance.

The final knots vectors determined by our method after 20 iterations are \( u = (0, 0, 0, 0.25, 0.5, 0.75, 1, 1, 1, 1) \) and \( v = (0, 0, 0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1, 1, 1, 1) \). Figure 14 shows the final approximated surface of our method and De Boor’s method. Compared with the initial fitting, both methods improve significantly. However, the final fitted surface by De Boor’s method still overfits the true surface at the four corners.

The quantitative comparisons based on RMSE on a 101 \( \times \) 101 uniformly distributed grid in the range of \([0, 1]\) are summarized in Table 3. The final RMSE results of the two methods show that our proposed method is 38.2% better than De Boor’s method. Compared with the smooth surface in Sec. 4.3, the proposed method demonstrates more advantages on the sinc surface than De Boor’s method. This is because adaptive knots vectors distribution is more critical for nonuniform surfaces and hence our method shows more improvements.

### 5 Conclusion

In this paper, we proposed a sequential B-spline surface construction procedure. It contains mainly two steps: Whenever the new data points are available, we first refine the knots vectors to reduce the bias of the surface approximation; then, we incorporate new data points to reduce the variance of the surface approximation. Compared with the existing works on B-spline surface approximation, our method has the advantages in the following aspects. First, the approximation performance improves. Our method not only reduces the variance term but also the bias term of the approximation error in each updating step. Second, knots locations are adaptively and automatically determined. Since we refine the knots vectors in each updating step to reduce the upper bound of the approximation error, the knots are distributed adaptively to the underlying structure of the true surface. Moreover, we refine the knots vectors gradually as the information of the surface is getting better, misspecification of knots vectors due to insufficient data are avoided. Third, asymptotical convergence is guaranteed. The simulation study presented in Sec. 4 also illustrates the effectiveness of our method under the finite sample case.

There are still some open issues not explored in this paper. In the current paper, all the discussions are based on one assumption: newly sampled data are given or sampled independently from the current surface approximation. This kind of sampling scheme guarantees the asymptotical convergence as long as the conditions in Theorem 2 are satisfied. However, in practice, when the measurement data are finite, this scheme might cause overfitting in some areas. Since the randomly and independently sampled data usually spread out, while the newly added knots might concentrate among rough areas of the surface, the insufficient data points compared with dense knots could cause overfitting in these areas. To address this problem, one possible solution would be to use the locations of newly added knots to guide the new data sampling. The principle of controlled sampling scheme is to balance the knots distribution with the sample distribution such that their ratios are uniform across the whole surface. The controlled sampling scheme could improve the approximation performance. However, since the locations of newly added knots are determined according to the current surface approximation, then the newly sampled data under this scheme would be also dependent with the current approximation. The asymptotical convergence of this scheme is yet to be proved. Another issue is the control parameter \( M \) in the knots refinement algorithm. Since \( M \) affects both the asymptotical convergence of our proposed method and its finite approximation performance, how to adaptively adjust the value of \( M \) would be worth further investigation. We will continue the investigation along this line and provide reports in the future.

### Acknowledgment

The authors appreciate the editors and reviewers for this paper for their very constructive comments and suggestions. This work was financially supported by the National Science Foundation, under Grants 0856222 and 0545600.

### References