

An Analysis and Comparison of Parameterization-Based Computation of Differential Quantities for Discrete Surfaces

Duo Wang, Bryan Clark, Xiangmin Jiao^{*},

Dept. of Applied Mathematics and Statistics, Stony Brook University, New York 11794.

Abstract

Normals and curvatures are fundamental for geometric modeling and computer-aided design, but their accurate computations on discrete surfaces are challenging. Two types of methods, namely height-function based and parameterization based polynomial fittings, are well founded mathematically and can be proven to deliver convergent results under reasonable assumptions. However, the numerical behaviors of these methods can differ drastically in practice, and no systematic analysis and comparison have been reported previously for these methods. In this paper, we describe a unified framework for these methods based on weighted least squares approximations, and on top of this framework we compare a number of methods in terms of numerical accuracy and stability as well as runtime efficiency and robustness through both theoretical analysis and numerical experiments. Our analysis shows that the choice of parameterization and numerical solver for the least squares problem can have significant impact on the accuracy and stability of polynomial fittings. In addition, we show that the methods based on local orthogonal projection with a safeguard against folding delivers the best combination of simplicity, accuracy, efficiency, and robustness.

Key words: Differential geometry, surface meshes, discrete operators, normals, curvatures

1 Introduction

Computing normals and curvatures is a fundamental problem for many geometric and numerical computations, including feature detection, shape retrieval, shape registration or matching, surface fairing, surface mesh adaptation or remeshing, front

^{*} Corresponding author. Phone: (+1) 631 6324408. Fax: (+1) 631 6328490.
Email address: jiao@ams.sunysb.edu (Xiangmin Jiao).

tracking and moving meshes. Many methods have been introduced for computing normals and curvatures, such as (Taubin, 1995; Meek and Walton, 2000; Meyer et al., 2002; Xu, 2004; Rusinkiewicz, 2004; Cazals and Pouget, 2005; Grinspun et al., 2006; Jiao and Zha, 2008). In recent years, there have been significant interests in the accuracy and stability of these methods, driven by the needs of geometric or physical modeling. Among the existing methods, some do not guarantee the convergence of the estimations except under some special conditions. For example, Meyer et al. (2002) proposed a cotangent formula for estimating mean curvatures, which is closely related to the formula for Dirichlet energy of Pinkall and Polthier (1993). It was shown that the cotangent formula does not produce converging pointwise mean-curvature estimations except for some special cases, as noted in (Borrelli et al., 2003; Xu, 2004; Hildebrandt et al., 2006; Wardetzky, 2007). As another example, Langer et al. (2007) proposed a tangent-weighted formula for estimating mean-curvature vectors, whose convergence relies on special symmetric patterns of a mesh. Cohen-Steiner and Morvan (2003) proposed a method for curvature computation based on the theory of normal cycles, but their error analysis is limited to the case of restricted Delaunay triangulations. On the other hand, a few methods based on polynomial fittings have been proven to deliver converging results (see e.g., Meek and Walton, 2000; Xu, 2004; Cazals and Pouget, 2005; Jiao and Zha, 2008). However, these methods can have drastically different numerical behaviors, and sometimes they can deliver very poor results in practice.

Accuracy and stability of polynomial fittings are subtle numerical issues. These issues are complicated due to the interactions among different aspects of the methods, including point selection, parameterization, numerical solvers, as well as their interactions with classical differential geometry formulas. Therefore, a systematic analysis and comparison of these polynomial-fittings based methods are decidedly nontrivial but are important to advance our understanding of this fundamental problem. The main objective of this paper is to present a systematic analysis and comparison of polynomial-fittings based methods including those based on parameterization or height functions. Our analysis is based on an extension of that in (Jiao and Zha, 2008) for local height functions. We emphasize that our objective is not to compare all the existing methods for normal and curvature estimations. Readers are referred to (Petitjean, 2002) and (Gatzke and Grimm, 2006) for more comprehensive surveys and comparisons. Although our focus is on polynomial fittings only, some of the numerical issues that we address also apply to some other methods, such as those in (Goldfeather and Interrante, 2004) and (Rusinkiewicz, 2004).

The main contributions of the paper are as follows. First, we present a complete set of formulas for computing differential operators for continuous parametric surfaces. Second, we extend the computational framework and associated accuracy and stability analysis in (Jiao and Zha, 2008) for height functions to parametric surfaces. Third, we present theoretical and experimental comparison for a number of polynomial-fitting based methods. Our analysis shows that the choices of parameterization and numerical solver for the polynomial fittings can have signif-

icant impact on the accuracy and stability of the resulting estimations. In addition, we show that the methods based on local orthogonal projection with a safeguard against folding delivers the best combination of simplicity, accuracy, efficiency, and robustness.

The remainder of this paper is organized as follows. Section 2 analyzes the classical formulas for continuous parametric surfaces and presents some alternative formulas that are more amenable to numerical computation for their numerical stability. Section 3 presents a unified framework for polynomial fittings based on local parameterizations and weighted least squares approximations. Section 4 analyzes a few methods within this framework and compares them in terms of point selection, smoothness of parameterization, and robustness of numerical solver. Section 5 presents some numerical experiments to verify our theoretical analysis and compare the different algorithms in terms of accuracy and runtime efficiency. Section 6 concludes the paper with a discussion.

2 Formulas for Continuous Surfaces

Reliable computations of differential quantities on discrete surfaces critically depend on their counterparts for continuous surfaces. The latter is a subject in classical differential geometry and has been studied extensively, but the numerical behaviors of classical formulas have not been carefully scrutinized until recently. In (Jiao and Zha, 2008), the stability of the classical formulas for height functions was analyzed, and some new formulas were proposed. In this section, we extend the analysis to the formulas for parametric surfaces and propose some alternative formulas that are more amenable to numerical computations.

2.1 Formulas for Parametric Surfaces

Given a smooth surface Γ in the global xyz coordinate system, let $\mathbf{x} = \mathbf{f}(\mathbf{u})$ be a parameterization of a neighborhood around a vertex on Γ , where $\mathbf{u} = [u, v]^T$ (we consider all vectors as column vectors for consistency with the modern convention in numerical analysis.). If the surface is smooth, the coordinate function \mathbf{f} defines a smooth surface composed of points $\mathbf{x}(\mathbf{u}) \in \mathbb{R}^3$. The Jacobian matrix of $\mathbf{x}(\mathbf{u})$ with respect to \mathbf{u} is then $\mathbf{J} = [\mathbf{x}_u, \mathbf{x}_v]$. The vectors \mathbf{x}_u and \mathbf{x}_v form a basis of the tangent space of the surface. Let $d\mathbf{u}$ denote $[du, dv]^T$. The *first fundamental form* of the surface is the quadratic form

$$I(d\mathbf{u}) = d\mathbf{u}^T \mathbf{G} d\mathbf{u}, \text{ where } \mathbf{G} = \mathbf{J}^T \mathbf{J}.$$

\mathbf{G} is known as the *first fundamental matrix*. Its determinant is $g = \det(\mathbf{G}) = \|\mathbf{x}_u \times \mathbf{x}_v\|^2$. Let ℓ denote \sqrt{g} , i.e., $\ell = \|\mathbf{x}_u \times \mathbf{x}_v\|$, which we refer to as the “*area*”

element.” The unit normal to the surface is then

$$\hat{\mathbf{n}} = \frac{\mathbf{x}_u \times \mathbf{x}_v}{\ell}. \quad (1)$$

The *second fundamental form* in the basis $\{\mathbf{x}_u, \mathbf{x}_v\}$ is given by the quadratic form

$$II(du) = du^T \mathbf{B} du$$

where

$$\mathbf{B} = - \begin{bmatrix} \hat{\mathbf{n}}_u^T \mathbf{x}_u & \hat{\mathbf{n}}_u^T \mathbf{x}_v \\ \hat{\mathbf{n}}_v^T \mathbf{x}_u & \hat{\mathbf{n}}_v^T \mathbf{x}_v \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{n}}^T \mathbf{x}_{uu} & \hat{\mathbf{n}}^T \mathbf{x}_{uv} \\ \hat{\mathbf{n}}^T \mathbf{x}_{uv} & \hat{\mathbf{n}}^T \mathbf{x}_{vv} \end{bmatrix}, \quad (2)$$

and is known as the *second fundamental matrix*.

The well-known *Weingarten equations* read $[\hat{\mathbf{n}}_u \mid \hat{\mathbf{n}}_v] = -[\mathbf{x}_u \mid \mathbf{x}_v] \mathbf{W}$, where \mathbf{W} is the *Weingarten matrix* (a.k.a. the *shape operator*) with basis $\{\mathbf{x}_u, \mathbf{x}_v\}$. By left-multiplying \mathbf{J}^T on both sides, we have $\mathbf{B} = \mathbf{G}\mathbf{W}$, and therefore,

$$\mathbf{W} = \mathbf{G}^{-1} \mathbf{B}. \quad (3)$$

The *mean curvature* is equal to half of the trace of \mathbf{W} . The *Gaussian curvature* is equal to the determinant of \mathbf{W} . Let κ_1 and κ_2 denote the eigenvalues of \mathbf{W} , which are the *principal curvatures*. Then, $\kappa_H = (\kappa_1 + \kappa_2)/2$ and $\kappa_G = \kappa_1 \kappa_2$. Let $\hat{\mathbf{d}}_1$ and $\hat{\mathbf{d}}_2$ be the eigenvectors of \mathbf{W} . Then $\hat{\mathbf{e}}_1 = \mathbf{J}\hat{\mathbf{d}}_1/\|\mathbf{J}\hat{\mathbf{d}}_1\|$ and $\hat{\mathbf{e}}_2 = \mathbf{J}\hat{\mathbf{d}}_2/\|\mathbf{J}\hat{\mathbf{d}}_2\|$ are the *principal directions*. The principal curvatures and principal directions are sometimes used to construct a 3×3 matrix

$$\mathbf{C} = \kappa_1 \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_1^T + \kappa_2 \hat{\mathbf{e}}_2 \hat{\mathbf{e}}_2^T. \quad (4)$$

We refer to \mathbf{C} as the *principal curvature tensor* or simply the *curvature tensor* for brevity.

The Weingarten matrix in (3) is classical in differential geometry, but it is not well-suited for numerical computations. The reason is that the matrix is in general not symmetric, so its eigenvectors are not orthogonal to each other. In addition, the eigenvalues of \mathbf{W} are not necessarily real in the presence of round-off errors. For robust numerical computations, we derive a symmetric shape operator as follows. Let $\mathbf{J} = \mathbf{Q}\mathbf{R}$ denote the QR factorization of \mathbf{J} , where \mathbf{Q} is 3×2 with orthonormal column vectors (i.e., $\mathbf{Q}^T \mathbf{Q} = \mathbf{I}$) and \mathbf{R} is a 2×2 upper triangular matrix. The QR factorization can be constructed using Gram-Schmidt orthogonalization (see e.g., Golub and Van Loan, 1996). Let $\hat{\mathbf{q}}_1$ and $\hat{\mathbf{q}}_2$ denote the column vectors of \mathbf{Q} . The shape operator in the orthonormal basis $\{\hat{\mathbf{q}}_1, \hat{\mathbf{q}}_2\}$ is the symmetric matrix

$$\tilde{\mathbf{W}} = \mathbf{R}^{-T} \mathbf{B} \mathbf{R}^{-1}, \quad (5)$$

and the curvature tensor is then

$$\mathbf{C} = \mathbf{Q} \tilde{\mathbf{W}} \mathbf{Q}^T = \mathbf{J}^{+T} \mathbf{B} \mathbf{J}^+, \quad (6)$$

where $\mathbf{J}^+ = \mathbf{R}^{-1}\mathbf{Q}^T$ is the pseudo-inverse of \mathbf{J} . Note that the curvature tensor (6) is equivalent to the *embedded Weingarten map* in (Peters and Reif, 2008, p. 20). Eq. (6) also has the same form as Eq. (13) in (Jiao and Zha, 2008), but it is more general in that it applies to parametric surfaces instead of just local height functions (see subsection 2.2).

After obtaining the symmetric shape operator $\tilde{\mathbf{W}}$, its eigenvalues are guaranteed to be real and its eigenvectors are guaranteed to be orthonormal. More specifically, let w_{ij} denote the entries of $\tilde{\mathbf{W}}$. The principal curvatures are then

$$\kappa_{1,2} = \frac{1}{2} \left(w_{11} + w_{22} \pm \sqrt{(w_{11} - w_{22})^2 + 4w_{12}^2} \right). \quad (7)$$

If $\kappa_1 = \kappa_2$, we choose the principal directions to be $\hat{\mathbf{q}}_1$ and $\hat{\mathbf{q}}_2$, respectively. Otherwise, the principal direction corresponding to κ_i is

$$\hat{\mathbf{e}}_i = \frac{(w_{11} - \kappa_{3-i})\hat{\mathbf{u}}_1 + w_{12}\hat{\mathbf{u}}_2}{\|(w_{11} - \kappa_{3-i})\hat{\mathbf{u}}_1 + w_{12}\hat{\mathbf{u}}_2\|} = \frac{w_{12}\hat{\mathbf{u}}_1 - (w_{11} - \kappa_i)\hat{\mathbf{u}}_2}{\|w_{12}\hat{\mathbf{u}}_1 - (w_{11} - \kappa_i)\hat{\mathbf{u}}_2\|} \quad (8)$$

for $i = 1, 2$. For better stability, we use the first equality if $|w_{11} - \kappa_{3-i}| > |w_{11} - \kappa_i|$ and use the second equality otherwise.

2.2 Formulas for Height Functions

The preceding formulas apply to any smooth parameterization of a neighborhood of any point on a surface. A special parameterization is associated with the so-called ‘‘local height function,’’ obtained by the orthogonal projection of the neighborhood onto a plane that is nearly tangential to the surface. Specifically, given a smooth surface in the global xyz coordinate system, it can be transformed into a local uvw coordinate system by translation and rotation. Assume both coordinate frames are orthonormal right-hand systems. Let the origin of the local frame be at point $[x_0, y_0, z_0]^T$. Let $\hat{\mathbf{t}}_1$ and $\hat{\mathbf{t}}_2$ be the unit vectors in the global coordinate system along the positive directions of the u and v axes, respectively. Then, $\hat{\mathbf{m}} = \hat{\mathbf{t}}_1 \times \hat{\mathbf{t}}_2$ is the unit vector along the positive w direction. Let \mathbf{U} denote the orthogonal matrix composed of column vectors $\hat{\mathbf{t}}_1$, $\hat{\mathbf{t}}_2$ and $\hat{\mathbf{m}}$, i.e., $\mathbf{U} = [\hat{\mathbf{t}}_1, \hat{\mathbf{t}}_2, \hat{\mathbf{m}}]$. Any point \mathbf{x} on the surface is then transformed to a point $\mathbf{p}(\mathbf{x}) = [u, v, f(\mathbf{u})]^T = \mathbf{U}^T(\mathbf{x} - \mathbf{x}_0)$. Conversely, $\mathbf{x} = \mathbf{U}\mathbf{p} + \mathbf{x}_0$. The function $f(\mathbf{u}) : \mathbb{R}^2 \rightarrow \mathbb{R}$ (more precisely, from a subset of \mathbb{R}^2 to \mathbb{R}) is a *height function* near x_0 . In the uvw coordinate system, the first two components of the coordinate function are u and v , respectively, so the differential quantities depend on only the derivatives of f with respect to u and v .

Let $\nabla f = [f_u, f_v]^T$ denote the gradient of f with respect to \mathbf{u} and $\mathbf{H} = \begin{bmatrix} f_{uu} & f_{uv} \\ f_{vu} & f_{vv} \end{bmatrix}$ the Hessian of f , where $f_{uv} = f_{vu}$. For this particular form, the Jacobian matrix

and the first and second fundamental matrices can be written explicitly in terms of ∇f and \mathbf{H} , resulting in some simple closed-form formulas. Table 1 summarizes the formulas of first- and second-order differential quantities of a parametric surface or local height function. We separate the table into three parts by double lines. Most of the first two parts are well known; see e.g. (do Carmo, 1976). The formulas for mean and Gaussian curvatures of local height functions were derived in (Jiao and Zha, 2008). To the best of our knowledge, the third part of the table has not appeared in the literature previously. Note that in (Jiao and Zha, 2008) a symmetric shape operator was derived for local height functions using the singular value decomposition (SVD) of the Jacobian matrix, but unfortunately the Jacobian matrix for a parametric surface is too complex to derive an explicit SVD, so we express the symmetric shape operator using QR factorization instead.

2.3 Analysis of Stability

The formulas for the classical Weingarten matrix (3) and the symmetric shape operator (5) both involve an inverse operation. Symbolically, the operators are valid if and only if \mathbf{J} has full rank. However, in the presence of errors in \mathbf{J} and \mathbf{B} (such as roundoff or approximation errors), these errors can propagate into \mathbf{W} or $\tilde{\mathbf{W}}$, and in turn lead to errors in the principal curvatures and principal directions.

Let $\kappa_{\mathbf{J}}$ denote the condition number of the Jacobian matrix in 2-norm, i.e., the ratio between the largest and smallest singular values of \mathbf{J} . From perturbation theory we know that the errors in \mathbf{W} and $\tilde{\mathbf{W}}$ depend on the errors in \mathbf{J} and \mathbf{B} as well as the condition numbers of \mathbf{J} . Let $\epsilon_{\mathbf{J}}$ and $\epsilon_{\mathbf{B}}$ denote the norms of the errors in \mathbf{J} and \mathbf{B} . The error in \mathbf{W} and $\tilde{\mathbf{W}}$ are then both $O(\kappa_{\mathbf{J}}^4 \epsilon_{\mathbf{J}} + \kappa_{\mathbf{J}}^2 \epsilon_{\mathbf{B}})$. Therefore, it is important that $\kappa_{\mathbf{J}}$ is not too large (i.e., \mathbf{J} is far from being rank deficiency).

While the errors in the classical and symmetric shape operators have similar scales, the eigenvalues and eigenvectors of these shape operators can behave drastically differently. In terms of the eigenvalues, the eigenvalues of a symmetric matrix are always well conditioned, but the eigenvalues of a non-symmetric matrix may be ill conditioned, depending on the angles between the eigenvectors. The skewness of the Jacobian matrix can therefore severely jeopardize the stability of the principal curvatures for the classical shape operator. In terms of the principal directions, the eigenvectors of a symmetric shape operators are always orthogonal to each other, but those for the classical shape operator are in general not. The loss of orthogonality of the eigenvectors can translate into large angular errors of the principal directions. These errors in the principal curvature and principal directions from the classical shape operator can further pollute the principal curvature tensor (4). Eq. (6) overcomes this pollution error and at the same time provides a more efficient way for obtaining the curvature tensor.

Table 1

Summary of formulas for continuous parametric surfaces and height functions.

	local parameterization	local height function
description	$\mathbf{x} = \mathbf{f}(u, v)$	$\mathbf{x} = \mathbf{U} \begin{bmatrix} u \\ v \\ f(u, v) \end{bmatrix} + \mathbf{x}_0$
Jacobian matrix	$\mathbf{J} = \begin{bmatrix} \mathbf{x}_u & \mathbf{x}_v \end{bmatrix} = \underbrace{\begin{bmatrix} \hat{\mathbf{q}}_1 & \hat{\mathbf{q}}_2 \end{bmatrix}}_{\mathbf{Q}} \mathbf{R}$	$\mathbf{U} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ f_u & f_v \end{bmatrix}$
1st fundamental matrix	$\mathbf{G} = \mathbf{J}^T \mathbf{J}$	$\begin{bmatrix} 1 + f_u^2 & f_u f_v \\ f_u f_v & 1 + f_v^2 \end{bmatrix}$
area element	$\ell = \sqrt{\det(\mathbf{G})} = \ \mathbf{x}_u \times \mathbf{x}_v\ _2$	$1 + f_u^2 + f_v^2$
surface normal	$\hat{\mathbf{n}} = \frac{\mathbf{x}_u \times \mathbf{x}_v}{\ell}$	$\frac{1}{\ell} \mathbf{U} \begin{bmatrix} -f_u \\ -f_v \\ 1 \end{bmatrix}$
2nd fundamental matrix	$\mathbf{B} = \begin{bmatrix} \hat{\mathbf{n}}^T \mathbf{x}_{uu} & \hat{\mathbf{n}}^T \mathbf{x}_{uv} \\ \hat{\mathbf{n}}^T \mathbf{x}_{uv} & \hat{\mathbf{n}}^T \mathbf{x}_{vv} \end{bmatrix}$	$\underbrace{\begin{bmatrix} f_{uu} & f_{uv} \\ f_{vu} & f_{vv} \end{bmatrix}}_{\mathbf{H}} / \ell$
Weingarten matrix	$\mathbf{W} = \mathbf{G}^{-1} \mathbf{B}$	$\frac{1}{\ell} (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{H}$
mean curvature	$\kappa_H = \frac{1}{2} \text{tr}(\mathbf{W})$	$\frac{\text{tr}(\mathbf{H})}{2\ell} - \frac{(\nabla f)^T \mathbf{H} (\nabla f)}{2\ell^3}$
Gaussian curvature	$\kappa_G = \det(\mathbf{W}) = \det(\mathbf{B}) / \ell^2$	$\det(\mathbf{H}) / \ell^4$
symmetric shape operator	$\tilde{\mathbf{W}} = \mathbf{R}^{-T} \mathbf{B} \mathbf{R}^{-1} = [w_{ij}]$ in basis $\{\hat{\mathbf{q}}_1, \hat{\mathbf{q}}_2\}$	
principal curvature tensor	$\mathbf{C} = \mathbf{Q} \tilde{\mathbf{W}} \mathbf{Q}^T = \mathbf{J}^{+T} \mathbf{B} \mathbf{J}^+$	
principal curvatures	$\kappa_{1,2} = \frac{1}{2} \left((w_{11} + w_{22}) \pm \sqrt{(w_{11} - w_{22})^2 + 4w_{12}^2} \right)$	
principal directions	$\hat{\mathbf{e}}_i = \frac{(w_{11} - \kappa_{3-i}) \hat{\mathbf{q}}_1 + w_{12} \hat{\mathbf{q}}_2}{\ (w_{11} - \kappa_{3-i}) \hat{\mathbf{q}}_1 + w_{12} \hat{\mathbf{q}}_2\ } = \frac{w_{12} \hat{\mathbf{q}}_1 - (w_{11} - \kappa_i) \hat{\mathbf{q}}_2}{\ w_{12} \hat{\mathbf{q}}_1 - (w_{11} - \kappa_i) \hat{\mathbf{q}}_2\ }$	

3 A Computational Framework for Discrete Surfaces

To apply the formulas for continuous surfaces to discrete surfaces, we must construct a parameterization and estimate the first and second derivatives of the coordinate functions with respect to the parameterization. Different methods may use different parameterizations and/or numerical solvers for estimating the derivatives. To compare and analyze different methods, it is important to consider them in a

common framework. In this section, we assume that a neighborhood around a point has been selected and a parameterization is obtained (we will address these issues in the next section). Hereafter, we focus on the description and analysis of a common framework that unifies the methods under consideration, and present a thorough analysis of this framework.

3.1 Weighted Least Squares Polynomial Fitting

In approximation theory, the Taylor series expansion is a powerful tool in deriving numerical approximations. Let \mathbf{u} denote $[u, v]^T$ and $f(\mathbf{u})$ denote a smooth bivariate function, which may be the local height function under orthogonal projection, or the x , y , or z component of the coordinate function for a parametric surface. Let c_{jk} be a shorthand for $\frac{\partial^{j+k}}{\partial u^j \partial v^k} f(\mathbf{0})$. Given a positive integer d , if $f(\mathbf{u})$ has $d + 1$ continuous derivatives, it can be approximated to $(d + 1)$ st order accuracy about the origin $\mathbf{u}_0 = [0, 0]^T$ by

$$f(\mathbf{u}) = \underbrace{\sum_{p=0}^d \sum_{j,k \geq 0} c_{jk} \frac{u^j v^k}{j!k!}}_{\text{Taylor polynomial}} + \underbrace{\sum_{j,k \geq 0} \frac{\partial^{j+k}}{\partial u^j \partial v^k} f(\tilde{u}, \tilde{v}) \frac{\tilde{u}^j \tilde{v}^k}{j!k!}}_{\text{remainder}}, \quad (9)$$

where $0 \leq \tilde{u} \leq u$ and $0 \leq \tilde{v} \leq v$. We emphasize that this equality assumes that f is continuously differentiable up to $d + 1$, and we refer to this as the *regularity assumption*. The derivatives of the Taylor polynomial are the same as f at \mathbf{u}_0 up to degree d , and hence the problem of estimating the derivatives reduces to the estimation of the coefficients c_{jk} of the Taylor polynomial. Specifically, given a set of data points, say $[u_i, v_i, f_i]^T$ for $i = 1, \dots, m - 1$, sampled from a neighborhood near a point $\mathbf{u}_0 = [u_0, v_0, f_0]^T$ on a smooth surface. Plugging in each given point into (9), we obtain an approximate equation

$$\sum_{p=0}^d \sum_{j,k \geq 0} c_{jk} \frac{u_i^j v_i^k}{j!k!} \approx f_i, \quad (10)$$

which has $n = (d + 1)(d + 2)/2$ unknowns (i.e., c_{jk} for $0 \leq j + k \leq d$, $j \geq 0$ and $k \geq 0$), resulting in an $m \times n$ rectangular linear system. We refer to d as the *degree of fitting*. Note that one could enforce the fit to pass through the point \mathbf{u}_0 by setting $c_{00} = 0$ and removing the equation corresponding to \mathbf{u}_0 , reducing to an $(m - 1) \times (n - 1)$ rectangular linear system, but there is typically little or no benefit for doing it.

The above method for estimating the Taylor polynomial is known as *polynomial fitting* or *local polynomial fitting*, because the fitting is in a local neighborhood

around point \mathbf{u}_0 . Let us denote the rectangular linear system obtained from (10) as

$$\mathbf{V}\mathbf{c} \approx \mathbf{f}, \quad (11)$$

where \mathbf{c} is an n -vector composed of c_{jk} , and \mathbf{V} is a *generalized Vandermonde matrix*. For a local height function, \mathbf{f} is an m -vector composed of f_i ; for a parametric surface, \mathbf{f} is an $m \times 3$ matrix, of which each column corresponds to a component of the coordinate function.

Numerically, (11) can be solved using the framework of *weighted linear least squares* (Golub and Van Loan, 1996, p. 265), i.e., to minimize a weighted norm (or semi-norm),

$$\min_{\mathbf{c}} \|\mathbf{V}\mathbf{c} - \mathbf{f}\|_{\Omega} = \min_{\mathbf{c}} \|\Omega(\mathbf{V}\mathbf{c} - \mathbf{f})\|_2, \quad (12)$$

where Ω is a *weighting matrix*. Typically, Ω is an $m \times m$ diagonal matrix, whose i th diagonal entry ω_i assigns a priority to the i th point $[u_i, v_i]^T$ by scaling the i th row of \mathbf{V} . This formulation is equivalent to the linear least squares problem

$$\tilde{\mathbf{V}}\mathbf{c} \approx \mathbf{b}, \text{ where } \tilde{\mathbf{V}} = \Omega\mathbf{V} \text{ and } \mathbf{b} = \Omega\mathbf{f}. \quad (13)$$

In general, $\tilde{\mathbf{V}}$ is $m \times n$ and $m \geq n$. However, this linear system may be rank deficient (i.e., the column vectors of $\tilde{\mathbf{V}}$ may not be linear dependent) or ill-conditioned (i.e., the singular values of $\tilde{\mathbf{V}}$ may have very different scales) due to a variety of reasons, including poorly scaling, insufficient number of points, or degenerate arrangements of points (Lancaster and Salkauskas, 1986). The scaling of \mathbf{A} can be improved substantially by introducing a *scaling matrix* \mathbf{S} and change the problem as

$$\min_{\mathbf{d}} \|\mathbf{A}\mathbf{d} - \mathbf{b}\|_2, \text{ where } \mathbf{A} = \tilde{\mathbf{V}}\mathbf{S} \text{ and } \mathbf{d} = \mathbf{S}^{-1}\mathbf{c}. \quad (14)$$

Here \mathbf{S} is typically a diagonal matrix. Let $\tilde{\mathbf{v}}_i$ denote the i th column of $\tilde{\mathbf{V}}$. The i th diagonal entry of \mathbf{S} is typically chosen to be either $\|\tilde{\mathbf{v}}_i\|_{\infty}$ (e.g., Cazals and Pouget, 2005; Xu, 2007) or $\|\tilde{\mathbf{v}}_i\|_2$ (Jiao and Zha, 2008), where the latter approximately minimizes the condition number of $\tilde{\mathbf{V}}\mathbf{S}$ (Golub and Van Loan, 1996, p. 265). However, the problem may still be ill-conditioned after rescaling, and we address this issue in Section 4.3.

3.2 Analysis of Accuracy and Stability

The weighted least-squares formulation is a well-known linear algebra technique. However, in the context of derivative estimation for parametric surfaces, its numerical analysis must be customized and does not seem to exist in the literature. We present an analysis in this subsection, focusing on two aspects: 1) the accuracy and stability of the least squares problem (14), and 2) the propagation of the errors from (14) to differential quantities.

3.2.1 Errors in Least Squares Formulation

By the perturbation theory, the error in the solution to Eq. (14) depends on a number of factors, including the input errors in \mathbf{A} and \mathbf{b} , the condition number of \mathbf{A} , the angle of \mathbf{b} with respect to column space of \mathbf{A} , as well as the orientation of \mathbf{b} within the column space of \mathbf{A} (Trefethen and Bau, 1997, Lecture 18). The entries in \mathbf{A} depend only on the parameterization, which we address in subsection 3.2.2. Here, we focus on the errors in \mathbf{b} and the condition number of \mathbf{A} .

The error in \mathbf{b} depends on both the “noise” in the input points and the residual in (9). Note that an important assumption behind (9) is the *regularity assumption*. If this assumption is violated, then the derivatives may be unbounded, and the errors in the remainder may be arbitrarily large. It is therefore very important that the coordinate functions are smooth with respect to the parameterization (i.e., with bounded derivatives).

We now consider the condition number of $\mathbf{A}\mathbf{d} \approx \mathbf{b}$. Assume the least squares problem is well conditioned and the numerical solver is stable. Let h denote the average edge length of the mesh, and assume the maximum diameter of the neighborhood is $O(h)$. We bound the errors in the approximations to the partial derivatives in terms of h .

Proposition 1 *Given a set of points $[u_i, v_i, \tilde{f}_i]$ that interpolate a smooth height function f or approximate f with an error of $O(h^{d+1})$. Assume the point distribution and the weighing matrix are independent of the mesh resolution, and the condition number of the scaled matrix $\mathbf{A} = \tilde{\mathbf{A}}\mathbf{S}$ in (14) is bounded. The degree- d weighted least squares fitting approximates c_{jk} to $O(h^{d-j-k+1})$.*

The proof of the proposition follows that for Theorem 4 in (Jiao and Zha, 2008). We omit the proof here for brevity. Note that in practice the matrix \mathbf{A} may be rank deficient or ill-conditioned, which poses challenges to the numerical solvers, as we will discuss in Section 4.3.

3.2.2 Error Propagation to Differential Quantities

The analysis above considers the errors in the approximations to c_{jk} , i.e. the partial derivatives of the coordinate functions with respect to the parameters u and v . From the coefficients c_{jk} , we compute the normal and curvatures using the formulas in Table 1. The errors in c_{jk} therefore can propagate into the computed normals and curvatures. Following the same proof as Theorem 5 in (Jiao and Zha, 2008), we obtain the following proposition.

Proposition 2 *Assume the position, gradient, and Hessian of coordinate functions that approximated to $O(h^{d+1})$, $O(h^d)$ and $O(h^{d-1})$, respectively, and assume the condition number of the Jacobian matrix is bounded. a) The angle between the*

computed and exact normals is $O(h^d)$; b) the components of the shape operator and curvature tensor are approximated to $O(h^{d-1})$; c) the Gaussian and mean curvatures are approximated to $O(h^{d-1})$.

In the proposition, it is important that the condition number of the Jacobian matrix is bounded. This requirement is a direct consequence of the stability analysis in Section 2.3. If the condition number of the Jacobian matrix is unbounded, the errors in c_{jk} can be magnified by an arbitrarily large factor. Note that this proposition makes no claim about the principal directions, as they are inherently unstable if the maximum and minimum curvatures are roughly equal. If the magnitudes of the principal curvatures are well separated, then the principal directions would also have similar convergence rates as the curvatures. When using our symmetric shape operator, the computed principal directions are guaranteed to be orthonormal.

3.2.3 Summary of Requirements for Accuracy and Stability

The analysis above indicates that polynomial fittings produce converging estimations of differential quantities under certain conditions. We summarize the conditions as follows, grouped into three categories. First of all, the input points must satisfy the following:

- 1. Decreasing neighborhood size:** The size of the neighborhood (in terms of the distances between points) should decrease asymptotically for finer meshes (i.e., should be $O(h)$).
- 2. Accurate input points:** For degree- d polynomial fitting, the coordinate functions should be at least $(d + 1)$ st order accurate (i.e., $O(h^{d+1})$).

These requirements are necessary for asymptotically bounding the errors of the input to polynomial fittings. These conditions are universal for any fitting method, and they may or may not be satisfied by certain applications. Under the above conditions, the parameterization must satisfy the following requirements:

- 3. Smooth coordinate functions:** The partial derivatives of the coordinate functions must be bounded with respect to the parameters in the given neighborhood.
- 4. Well conditioned Jacobian:** The Jacobian matrix must be far from rank deficiency (i.e., must be well conditioned).

These two conditions are related to, but do not necessarily imply, each other. Finally, additional requirements must be imposed on the numerical solver of the linear least squares system (14):

- 5. Robust solver:** The least squares solver must be stable and at the same time be able to resolve ill-conditioned systems.

Under the above conditions, convergence is guaranteed theoretically for the computed differential quantities, and small errors can be expected in practice for sufficiently fine meshes. On the other hand, if any of the above conditions is violated, convergence is not guaranteed and the errors can be large, although one may still observe convergence and relatively small errors in practice. The above analysis provides us guidelines for choosing the neighborhood, the parameterizations, and numerical solvers. It also provides us a platform for comparing different methods, as we report in the next two sections.

4 Analysis of Parameterization-based Approximations

The framework that we just described is very general. However, not all realizations of the framework can perform equally well in practice in terms of accuracy, stability, flexibility, and efficiency. In this section, we compare a few methods using the terminology of this framework. Because there potentially can be a large number of variants of different methods, we consider separately the issues of point selection, smoothness of parameterization, and robustness of numerical solver, and use some specific methods in the literature as examples during our discussions.

4.1 Point Selections Strategies

For polynomial fittings on meshes, it is typical to select the points based on mesh connectivity, and sometimes coupled with some geometry-based filtration. It is common to use a k -ring neighborhood for some integer k , such as 1, 2, or 3 (Xu, 2004; Cazals and Pouget, 2005; Gatzke and Grimm, 2006). The *1-ring neighbor faces* of a vertex v is the faces incident on v , and the *1-ring neighbor vertices* are the vertices of these faces. For an integer $k \geq 1$, the $(k + 1)$ -ring neighborhood of a vertex is the union of the 1-ring neighbors of its k -ring neighbor vertices.

When k is constrained to integers, the numbers of vertices in k -ring neighbors grow rapidly as k increases. For finer control, Jiao and Zha (2008) proposed to use half-ring increments by defining the *1.5-ring neighbor faces* to be the faces that share an edge with a 1-ring neighbor face, and the *1.5-ring neighbor vertices* to be the vertices of these faces. For an integer $k \geq 1$, the $(k + 1.5)$ -ring neighborhood is the union of the 1.5-ring neighbors of the k -ring neighbor vertices. Figure 1 illustrates the neighborhood definitions up to 2.5 rings. Table 2 shows the typical numbers of vertices of a $(d + 1)/2$ -ring for d up to 6, which have approximately twice as many points as the number of unknowns of degree- d fittings. In our later discussions, we allow k to have half increments when referring to k -ring neighbors. Note that under some pathological situations (such as near the boundary of an open surface), a $(d + 1)/2$ -ring may have insufficient number of points for degree- d fittings. Jiao

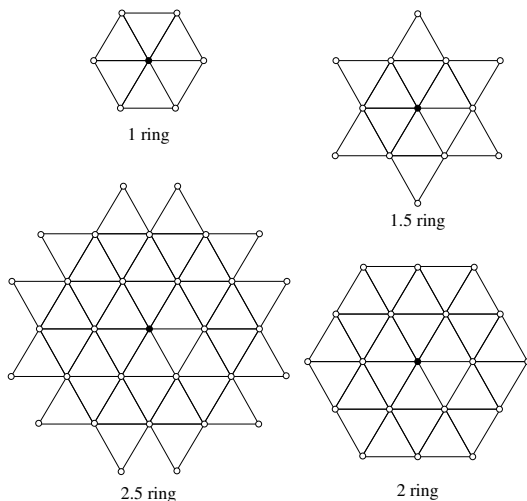


Fig. 1. Schematics of 1-, 1.5-, 2-, and 2.5-ring neighborhood. Each diagram shows the neighborhood of the center (black) vertex.

Table 2

Numbers of coefficients in d th degree fittings versus numbers of points in typical $\frac{d+1}{2}$ rings.

degree (d)	1	2	3	4	5	6
#coeffs.	3	6	10	15	21	28
#points in $\frac{d+1}{2}$ ring	7	13	19	31	37	55

and Zha (2008) proposed to adaptively enlarge the neighborhood size by half-ring increments if the number of vertices is fewer than 1.5 times of the required number of points. We follow such an adaptive strategy in this paper when appropriate.

A k -ring neighborhood depends only on mesh connectivity. For highly irregular meshes it could contain some points that are far away from the point of interest. One could address this problem by choosing the vertices based on distances rather than mesh connectivity (such as using the k nearest neighbors), but it is less efficient and prone to the well-known problem of “short circuiting” (i.e., to choose points that are close in Euclidean distance but are far away in geodesic distance). Under the weighted least squares framework, we can easily filter out any vertex within a k -ring neighbor by simply setting its corresponding weight in the weighting matrix Ω to zero or to a very small number. Typically, the weight can be inversely proportional to some power of the distance from the point to the vertex under consideration and sometime can also depend on the vertex normals for better robustness (Jiao and Zha, 2008). Therefore, k -ring neighbors with weighted least squares provides a simple and flexible approach, so we do not consider other point-select strategies here.

4.2 Alternatives of Parameterizations

Given a local neighborhood near a vertex, different methods may utilize different parameterizations. Parameterization has received significant attention in recent years in geometric modeling and computer graphics (e.g. Floater, 2003; Gotsman et al., 2003; Lévy et al., 2002). See (Floater and Hormann, 2005) for a survey of recent work on parameterization. It is important to note that polynomial fittings require only a local parameterization around a vertex, instead of a global parameterization of the whole surface mesh. This localization simplifies the problems both theoretically and computationally and allows specialized parameterization strategies. It is a natural but fundamental question how well these parameterizations are suited for our problem at hand. In this section, we evaluate three representative parameterizations in increasing generality.

4.2.1 Xu's 1-Ring Parameterization

To support quadratic fittings, Xu (2004; 2007) proposed a simple but specialized procedure to parameterize the 1-ring neighborhood of a vertex. At high level, this procedure flattens the neighborhood while preserving both the ratios between the interior angles of the triangles at the vertex and the lengths of the edges incident on the vertex. Such a parameterization is not unique, subject to translation and rotation. Xu eliminated the additional degrees of freedom by making the vertex the origin of the uv plane and choosing one of the edges as the u direction. We refer readers to (Xu, 2007) for detail.

Xu's 1-ring parameterization is very simple and efficient. It is approximately isometric at the vertex, so it is likely to be smooth. However, this construction has some limitations, among which the most notable is its limitation to 1-ring neighborhood. A 1-ring neighborhood sometimes does not even have enough points for quadratic fittings, not to mention cubic or higher-degree fittings. This limitation has serious consequences, as we will demonstrate in numerical experiments. In addition, the algorithm in (Xu, 2004, 2007) assumes a 1-ring neighborhood is topologically a disk so that the sum of the interior angles at a vertex is equal to 2π after flattening. The procedure would have to be modified for boundary vertices if it were used for open surfaces (more precisely, for 2-manifolds with boundary).

4.2.2 Local Orthogonal Projection with Safeguard

Another simple and efficient procedure for constructing the parameterization is to project the neighborhood orthogonal onto a plane. We refer to this method as *local orthogonal projection* (LOP). To avoid rank deficient (or ill-conditioned) Jacobian matrices, the plane should be approximately tangential to the surface. The LOP is often used in the construction of a local height function, as we describe in Section

2.2 and as done in (Cazals and Pouget, 2005) and (Jiao and Zha, 2008). Therefore, LOP enjoys the flexibility of utilizing the formulas for either local parameterization or local height function in Table 1.

The LOP is simple and efficient, but unlike Xu’s parameterization it is more general and can be applied to k -rings for $k \geq 1$. However, if the surface is highly curved and the mesh is too coarse, the projection of the k -ring neighborhood onto the plane may not be one-to-one, which in turn can lead to a violation of the regularity assumption of polynomial fittings.

The proneness to folding is the single most important issue with LOP. In the weighted least squares framework, this problem can be addressed by utilizing the weighting matrix to filter out vertices whose normals form too large an angle with the normal to the uv plane, as suggested in (Jiao and Zha, 2008). Specifically, let $\hat{\mathbf{m}}_i$ denote a rough estimation of the unit normal at the i th vertex, and let $\hat{\mathbf{m}}_0$ denote the normal to the uv plane, typically equal to the approximate vertex normal at the vertex in consideration. Note that the $\hat{\mathbf{m}}_i$ are used only for the construction of projection plane and the weights, so some simple averaging of face normals suffices. Jiao and Zha (2008) chose the weight for the i th vertex to be

$$\omega_i = \gamma_i / \left(\sqrt{\|\mathbf{u}_i\|^2 + \epsilon} \right)^{d/2}, \quad (15)$$

where ϵ is a small number to prevent division by zero. The key safeguard in (15) is γ_i , which is set to $\hat{\mathbf{m}}_i^T \hat{\mathbf{m}}_0$ if the angle between $\hat{\mathbf{m}}_i$ and $\hat{\mathbf{m}}_0$ is small but set to 0 if the angle is too large. For a typical mesh, γ_i is approximately equal to 1 and plays no role. For a coarse mesh with rapidly varying normals where a vertex may “wrap around,” γ_i would become zero and the vertex is filtered out. This safeguard is simple and efficient, and we employ it for LOP in our tests.

4.2.3 Conformal Parameterizations of k -Ring

Besides employing a safeguard, the potential problem of mesh folding of LOP can also be resolved (or alleviated) by using a more sophisticated parameterization strategy. Such an approach was advocated by some authors, such as Gatzke and Grimm (2006). A number of methods have been developed for planar parameterization of surfaces in recent years. A few have been implemented in CGAL (www.cgal.org), including barycentric mapping (Tutte, 1963), mean value coordinates (Floater, 2003), discrete authalic parameterization (Desbrun et al., 2002), and least squares conformal maps (LSCM) (Lévy et al., 2002); we refer readers to (Saboret et al., 2007) for a complete list.

The planar parameterization methods in general can be categorized as either fixed boundary or free boundary. For fixed-boundary methods, such as barycentric mapping and mean value coordinates, a surface is mapped onto a convex shape (such as

Table 3
High-level comparison of different parameterization schemes.

	advantages	disadvantages
Xu	simple and efficient	limited to 1-ring neighborhood
LOP	simple, efficient, flexible	prone to folding if without safeguard
LSCM	flexible	expensive; no guarantee against folding

a circle or a convex polygon). Such mappings are typically guaranteed to be one-to-one but are not necessarily smooth. Furthermore, because a k -ring neighborhood is not necessarily convex even for $k = 1$, mapping it to a convex shape can lead to arbitrarily large distortions and in turn large errors for polynomial fittings. Among the free-boundary methods, the most appealing types are those based on conformal mappings. For smooth compact surfaces, a conformal mapping is a type of harmonic mapping that preserves angles, so it is smooth by construction. For discrete surfaces, exact conformal mapping in general does not exist, and an approximation is constructed by solving a linear or nonlinear system of equations. Unlike free-boundary methods, fixed-boundary methods cannot guarantee the mapping to be one-to-one and can even lead to flipped triangles (Saboret et al., 2007), resulting in a violation of the regularity assumption for polynomial fittings. Therefore, none of these general parameterization methods is ideal for polynomial fittings.

After extensive experimentation, we selected LSCM of Lévy et al. (2002) as a representative for the general parameterization methods for our comparisons, because it is a free-boundary method based on conformal mappings, and in our tests it delivered better results than the others available in CGAL. Although LSCM does not guarantee against mesh folding, it is less likely to cause folding than a naive LOP without any safeguard. However, LSCM is more expensive than LOP for its requirement of solving a linear system. Its effectiveness compared to safeguarded LOP is unclear and can only be examined through numerical experimentation, which we report in Section 5.

4.2.4 Summary of Different Parameterizations

Before concluding this subsection, we summarize a high-level comparison of the advantages and disadvantages of different methods in Table 3. Note that each of the methods has its own disadvantages. Xu’s parameterization and LSCM suffer from inflexibility and inefficiency, respectively. LOP seems to be promising to deliver the best simplicity, efficiency, and flexibility. Its potential problem of mesh folding can be resolved in a simple manner with the weighted least squares formulation. In Section 5, we will study the accuracy and efficiency of different methods experimentally and verify this conclusion.

4.3 Alternatives of Numerical Solvers

In polynomial fittings the most difficult aspect is the solution of the least squares system (14), because this system can be rank deficient (i.e., under-determined) or even worse be nearly rank deficient (i.e., highly ill-conditioned). Such ill-conditioned problems can occur even if the number of points is greater than the number of unknowns. Although there exist general techniques for solving ill-conditioned least squares problems, such as SVD, an effective solution should take advantage of the special properties of the problem at hand. In this subsection, we compare two techniques, SVD, and a customization of QR factorization.

4.3.1 Singular Value Decomposition

In numerical linear algebra, singular value decomposition (SVD) is the standard technique for solving rank-deficient least squares problems; see for example (Golub and Van Loan, 1996, Chapter 5). Given a linear least squares problem $\mathbf{Ax} \approx \mathbf{b}$, where $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$, let $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ denote the SVD of \mathbf{A} , where $\mathbf{U} \in \mathbb{R}^{m \times m}$ is composed of left singular vectors \mathbf{u}_i of \mathbf{A} , $\mathbf{\Sigma} = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_n)$ with $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n$, and $\mathbf{V} \in \mathbb{R}^{n \times n}$ is composed of the right singular vectors \mathbf{v}_i of \mathbf{A} . A general solution to a rank-deficient problem $\mathbf{Ax} \approx \mathbf{b}$ is

$$\mathbf{x} = \sum_{\sigma_i > \epsilon} \mathbf{v}_i \mathbf{u}_i^T \mathbf{b} / \sigma_i,$$

where ϵ is small number close 0 (such as 10^{-8}). For an ill-conditioned problem, we can set ϵ to a small factor of σ_1 (such as $10^{-4}\sigma_1$), which effectively would solve a modified problem $\tilde{\mathbf{A}}\mathbf{x} \approx \mathbf{b}$, where $\tilde{\mathbf{A}} = \sum_{\sigma_i \geq \epsilon} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$. The condition number of $\tilde{\mathbf{A}}$, namely the ratio between its largest and smallest singular values, is bounded by $1/\epsilon$. Given that the noise in the input points is small, limiting the condition number of $\tilde{\mathbf{A}}$ effectively limits the sensitivity of \mathbf{x} with respect to the noise in \mathbf{b} .

In the context of polynomial fittings, SVD has been used to address the potential rank deficiency in (Xu, 2007) and (Cazals and Pouget, 2005, 2008). Xu (2007) solved (14) by constructing the normal equation $\mathbf{A}^T \mathbf{A} \mathbf{x} = \mathbf{A}^T \mathbf{b}$ and then solve it using the SVD of $\mathbf{A}^T \mathbf{A}$, which is equivalent to the eigenvalue decomposition of $\mathbf{A}^T \mathbf{A}$, as $\mathbf{A}^T \mathbf{A}$ is symmetric positive semi-definite. Let $\tilde{\sigma}_i$ denote the eigenvalues of $\mathbf{A}^T \mathbf{A}$ and $\tilde{\mathbf{u}}_i$ denote the eigenvectors of $\mathbf{A}^T \mathbf{A}$. Xu solves the system as $\mathbf{x} = \sum_{\tilde{\sigma}_i \geq \epsilon} \tilde{\mathbf{u}}_i \tilde{\mathbf{u}}_i^T \mathbf{A}^T \mathbf{b} / \tilde{\sigma}_i$, where ϵ was chosen to be 10^{-8} . In contrast, Cazals and Pouget (2005) used SVD of \mathbf{A} to solve (14) directly, but no detail was given. In either case, the use of SVD does not take into account the geometric meaning of polynomial fittings, and it does not give higher priorities to lower derivatives. In addition, SVD is far more expensive compared to techniques based on QR factorizations, as we describe next.

4.3.2 QR Factorization with Safeguard

Instead of using a standard technique, Jiao and Zha (2008) proposed a customized QR factorization with a safeguard for polynomial fittings. The idea is based on the observation that given the QR factorization $\mathbf{A} = \mathbf{Q}\mathbf{R}$, the QR factorization of the first k leading columns of \mathbf{A} (i.e., $\mathbf{A}_{:,1:k}$ using MATLAB-like notation) are the k leading columns \mathbf{Q} (i.e., $\mathbf{Q}_{:,1:k}$) and the $k \times k$ leading submatrix of \mathbf{R} (i.e., $\mathbf{R}_{1:k,1:k}$). If \mathbf{A} has a large condition number, one can remove the columns of \mathbf{A} from the right to obtain a better conditioned problem. This effectively reduces to a lower-degree fitting, so Proposition 1 is still applicable to bound the errors. Furthermore, because the condition number (in 2-norm) of \mathbf{A} is the same as that of \mathbf{R} , the condition number can be estimated efficiently. See (Jiao and Zha, 2008) for more detail. Compared to SVD, this QR factorization based approach is about four to five times faster than using SVD. Note that QR factorization with partial pivoting is another alternative for solving rank deficient least squares problem (Golub and Van Loan, 1996), but like SVD such an approach is less appropriate because it does not give higher priorities to lower derivatives.

5 Numerical Experimentation

In this section, we report some numerical experiments to verify our preceding analysis. The experimental results can be affected by a number of factors, such as input errors, surface topology, mesh connectivity, parameterization methods, degrees of fittings, and numerical solvers. For the input meshes, we consider three aspects: closed versus open surfaces, noise-free versus noisy input points, and well-shaped (or regular) versus poor-shaped (or irregular) meshes. For the algorithms, we evaluate four different methods that we described earlier, including (1) Xu’s 1-ring parameterization as detailed in (Xu, 2007), (2) local height functions (Jiao and Zha, 2008), and parameterizations based on (3) local orthogonal projection or (4) least-squares conformal maps (Lévy et al., 2002). When appropriate, we use different degrees of polynomial fittings (degrees 2, 3, or 4). We denote these different methods by Xu- d , LHF- d , LOP- d , LSCM- d , respectively, where d is the degrees of polynomial fittings.

The combinations of the above options lead to tens of cases. For each case we compute the normals, and mean and Gaussian curvatures, principal curvatures, and principal directions. For convergence study, we use four meshes of different resolutions for each case and compute the average convergence rate as

$$\text{convergence rate} = \frac{1}{3} \log_2 \left(\frac{\text{error of level 1}}{\text{error of level 4}} \right).$$

The convergence rate is showed at the right end of the curves. Let v denote the total number of vertices, and let $\hat{\mathbf{n}}_i$ and $\tilde{\mathbf{n}}_i$ denote the exact and computed unit

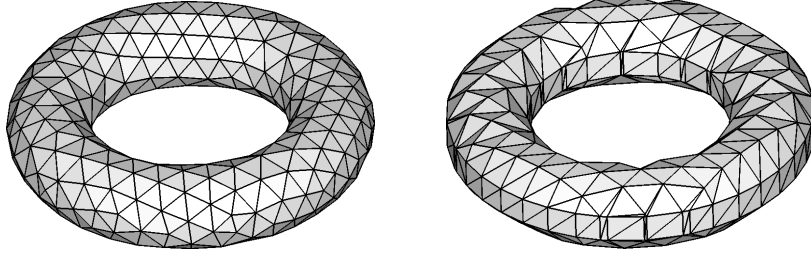


Fig. 2. Sample meshes of torus generated by commercial mesh generation software (left) and by isosurface algorithm (right).

vertex normals at the i th vertex. We measure the relative L_2 errors in normals as $\sqrt{\sum_1^v \|\tilde{\mathbf{n}}_i - \hat{\mathbf{n}}_i\|_2^2 / v}$. For comprehensiveness, we sometimes also consider the L_∞ error of normals, evaluated as $\max_i \|\tilde{\mathbf{n}}_i - \hat{\mathbf{n}}_i\|_2$. Let κ_i and $\tilde{\kappa}_i$ denote the exact and computed curvatures at the i th vertex. We measure the relative errors of curvatures in L_2 norm as $\|\tilde{\kappa} - \kappa\|_2 / \|\kappa\|_2 = \sqrt{\sum_{i=1}^v (\tilde{\kappa}_i - \kappa_i)^2} / \sqrt{\sum_{i=1}^v \kappa_i^2}$, and measure the L_∞ error as $\max_i |\tilde{\kappa}_i - \kappa_i| / |\kappa_i|$. Altogether, we obtain thousands of data points. In consideration of paper length, we report only a representative subset of results. All of our computations use double-precision floating point arithmetic.

5.1 Experiments for Noise-Free Closed Surfaces

We first present results on noise-free closed surfaces. We chose a torus with inner radius 0.7 and outer radius 1.3. A torus is representative for smooth surfaces as it contains parabolic, elliptic and hyperbolic points. In practice, a mesh can be well shaped (such as those obtained from CAD software for finite element analysis) or poor shaped (such as those from marching cubes for visualization purposes). To capture both types of meshes, we generated four triangular meshes using a commercial mesh generation software GAMBIT of Fluent Inc. (now part of ANSYS, Inc.) and four others using the isosurface function in MATLAB. Figure 2 shows a coarse mesh of either type. To eliminate potential input errors, we projected the vertices onto the torus so that the input points are accurate up to machine precision.

We first compare quadratic fittings using the four methods described earlier, denoted by Xu-2, LHF-2, LOP-2, and LSCM-2, respectively. For Xu-2, 1-ring neighbors were used, and for others 1.5-rings were used. Figure 3 shows the L_2 error and L_∞ errors in the computed normals, and Figure 4 shows the corresponding results for mean curvature. Note that except for Xu-2, the other methods converged at about quadratic rates for normals in terms of both L_2 error and L_∞ errors, and converged at about linear rates for curvatures in L_∞ error and nearly quadratic rates in L_2 error. The super-convergence in L_2 error is likely due to statistical cancellation of truncation errors. For Xu-2, the behavior was inconsistent: for well-shaped meshes it performed better than others in L_2 errors but substantially worse in L_∞

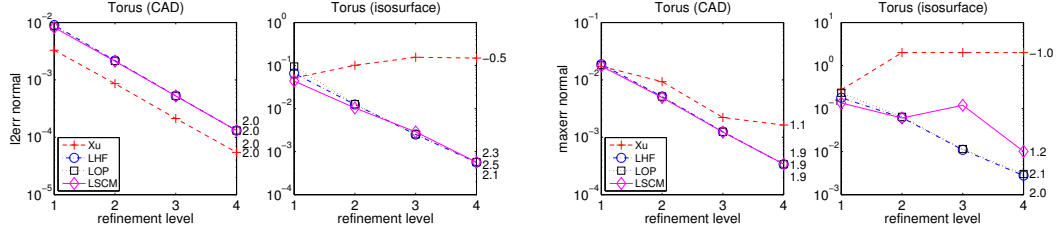


Fig. 3. Comparison of L_2 (left) and L_∞ (right) errors in computed normals using degree-2 fittings for noise-free torus meshes.

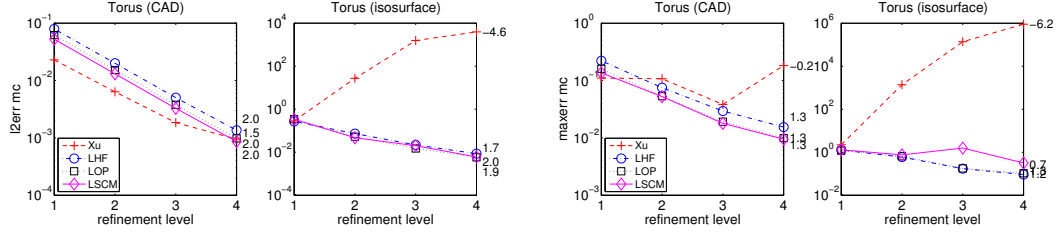


Fig. 4. Comparison of L_2 (left) and L_∞ (right) errors in computed mean curvatures using degree-2 fittings for noise-free torus meshes.

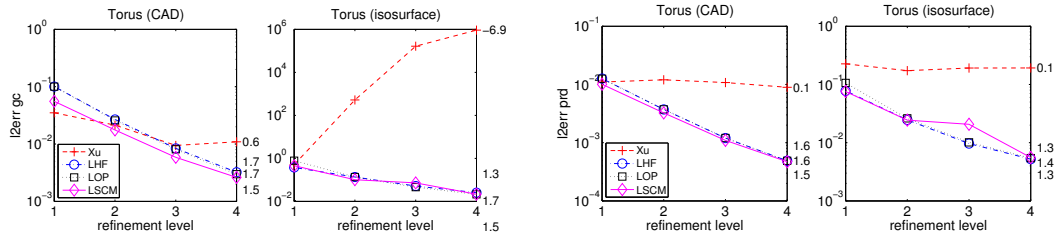


Fig. 5. Comparison of L_2 errors in computed Gaussian curvature and principal directions using degree-2 fittings for noise-free torus meshes.

errors, and for poor-shaped meshes it failed to converge. This inconsistent behavior is not surprising, because the 1-ring neighbors used in Xu-2 are more compact than 1.5-ring and in turn could sometimes deliver more accurate results, but they sometimes have too few points and lead to rank-deficient or ill-conditioned systems. Even though the SVD employed in Xu (2007) could produce numerical solutions in such cases, accuracy is not guaranteed. For completeness, we also report the L_2 errors in the computed Gaussian curvature and principal directions in Figure 5, whose behaviors were qualitatively similar to mean curvature.

For noise-free surfaces from CAD models or analytic functions, it is possible and sometimes desirable to obtain higher order estimations. Except for Xu’s method, the other three methods are capable of supporting higher-degree fittings. We consider only LHF and LSCM for high-degree fittings and evaluate degree-3 and 4 fittings using 2 and 2.5-rings, respectively, while adaptively increasing the ring size at half increments if the numbers of points are insufficient. Since LHF and LOP deliver nearly identical results (which is evident from the results of quadratic fittings), the primary difference between LHF and LSCM is the parameterization being used.

Figure 6 shows the L_2 and L_∞ errors in the computed normals. Figure 18 shows the L_2 errors in the computed mean and Gaussian curvatures. Note that LSCM-3 delivered slightly better accuracy than LHF-3 in most cases, but LHF-4 substantially out-performed LSCM-4 in almost all cases. In all cases, the convergence rates of LHF-4 were nearly an order higher than the theoretical prediction due to statistical error cancellations. In contrast, LSCM-4 converged at a rate nearly an order lower than the theoretical prediction for well-shaped meshes and failed to converge for poor-shaped meshes. This behavior of LSCM indicates that parameterizations can affect the accuracy of high-degree fittings significantly, and LSCM seems to be sufficiently smooth for cubic fitting but insufficient for higher-degree fittings.

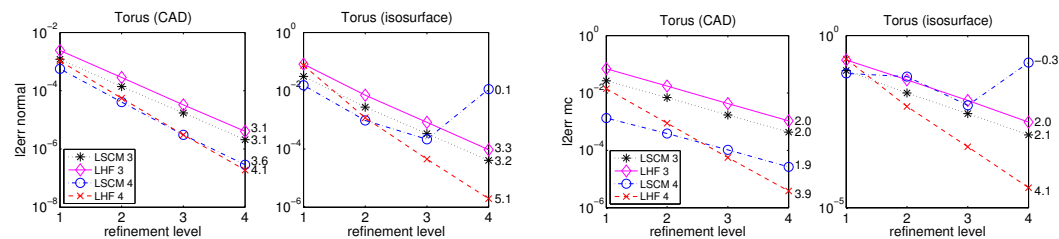


Fig. 6. Comparison of L_2 errors of in computed normals (left) and mean curvatures (right) using degree-3 and 4 fittings for noise-free torus meshes.

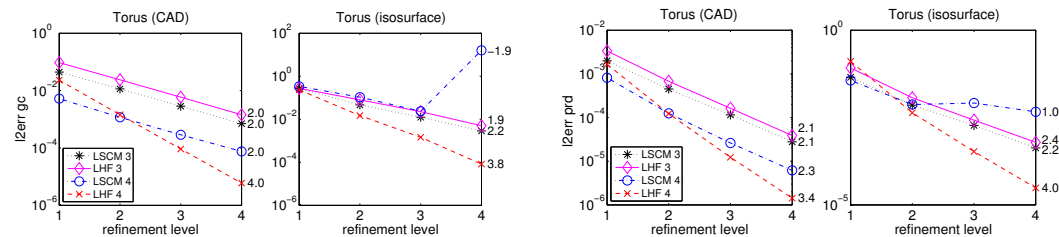


Fig. 7. Comparison of L_2 errors in computed Gaussian curvatures (left) and principal directions (right) using degree-3 and 4 fittings for noise-free torus meshes.

5.2 Experiments for Noise-Free Open Surfaces

In geometric modeling, surfaces often have boundaries and/or sharp features. It is therefore sometimes necessary to compute differential quantities using one-sided stencils. The boundaries (or sharp features) can adversely affect the computations of the differential quantities for two reasons: first, the neighborhood of a boundary vertex typically has fewer points, which may lead to ill-posed or ill-conditioned equations. Second, the stencil of boundary vertices are asymmetric, preventing statistical error cancellation associated with symmetry. Some existing methods for normal and curvature computations actually require the surface to be closed and rely on symmetry for convergence. Although polynomial fittings in general do not require closed surfaces, the accuracies of different methods may vary substantially near boundaries. To assess accuracies, we use a surface defined by the following

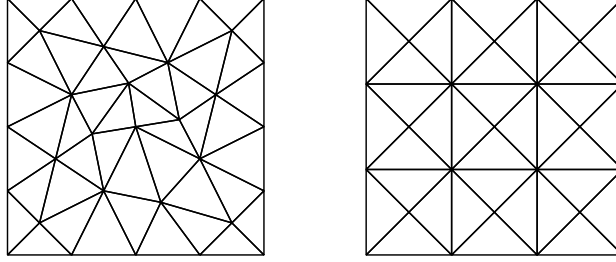


Fig. 8. Test meshes for open surfaces.

function adopted from (Xu, 2004):

$$z = f(x, y) = \exp\left(-\frac{81}{16} \left((x - 0.5)^2 + (y - 0.5)^2\right)\right),$$

where $(x, y) \in [0, 1] \times [0, 1]$. We use two types of meshes, as shown in Figure 8, which we refer to as irregular and 4-8 meshes, respectively. These meshes are both well-shaped compared to the meshes from isosurfacing algorithms, but the variations of vertex valences can pose some challenges to the algorithms.

Figure 9 shows the L_2 and L_∞ errors in the computed normals for Xu-2, LHF-2, LOP-2, and LSCM-2. Because Xu’s parameterization is limited to 1-ring neighborhoods that are topological disks, we excluded border vertices when computing errors for Xu-2. However, boundary vertices are included for all the other methods. Figure 10 shows the L_2 errors of the computed mean and Gaussian curvatures. Except for Xu-2, the other three methods delivered very similar results, but the convergence rates in L_2 errors of mean curvatures were lower than those for the torus, probably due to the loss of statistical error cancellations along boundaries. For Xu-2, it is worth noting that the curvatures failed to converge for both types of meshes due to insufficient number of points in the stencil.

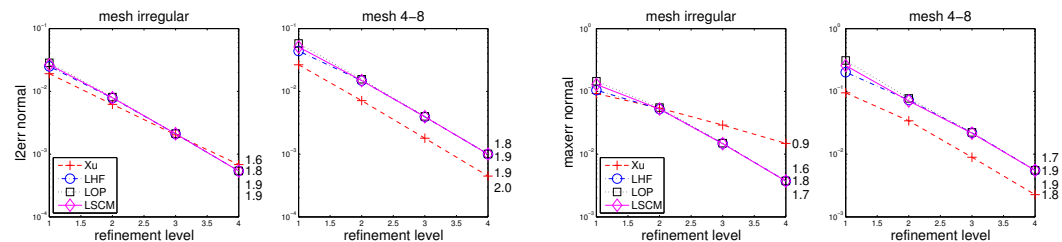


Fig. 9. Comparison of L_2 (left) and L_∞ (right) errors in computed normals using degree-2 fittings for noise-free open surface meshes.

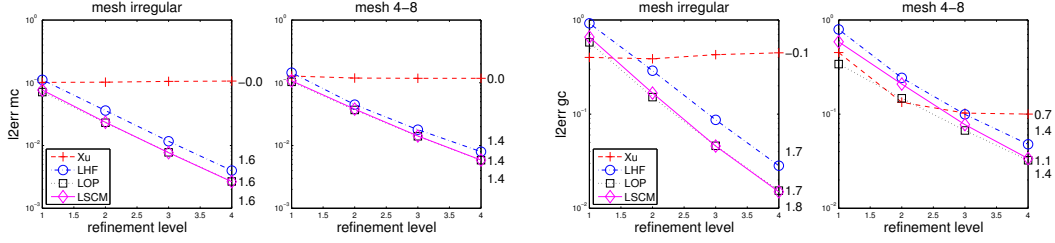


Fig. 10. Comparison of L_2 errors in computed mean (left) and Gaussian (right) curvatures using degree-2 fittings for noise-free open surface meshes.

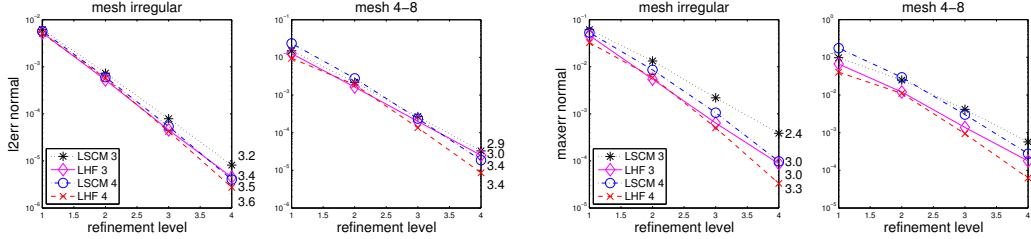


Fig. 11. Comparison of L_2 (left) and L_∞ (right) errors in computed normals using degree-3 and 4 fittings for noise-free open surface meshes.

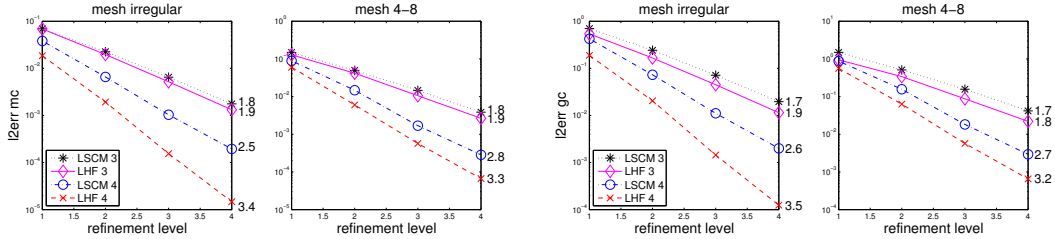


Fig. 12. Comparison of L_2 errors in computed mean (left) and Gaussian (right) curvatures using degree-3 and 4 fittings for noise-free open surface meshes.

Figures 11 and 12 compare the errors for degree-3 and 4 fittings, namely LHF-3, LHF-4, LSCM-3, and LSCM-4. LHF consistently delivered better accuracy than LSCM for both degree-3 and 4 fittings, and LSCM-4 converged substantially slower than LHF-4, which confirms our observation that LSCM is unsuitable for high-degree fittings. From these tests, we conclude that LHF and similarly LOP are more flexible and robust than Xu's method, and they deliver higher accuracy than LSCM for high-degree fittings.

In the proceeding results, we included both the interior and boundary vertices, except for Xu's method. To study the effect of border vertices, Figure 13 shows the L_2 errors and their convergence rates in the computed normals using LHF and LSCM for boundary vertices and their 1-ring neighbors Figure 14 shows the corresponding results for the mean and Gaussian curvatures. It can be seen that these convergence rates are somewhat lower than those for the interior vertices but are very close to the theoretical analysis. These results indicate that these fitting meth-

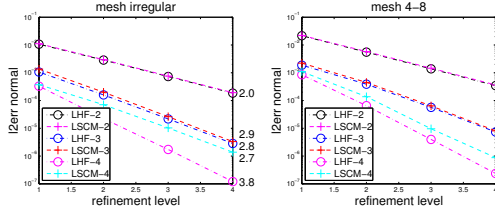


Fig. 13. L_2 errors in computed normals for boundary vertices for noise-free open surface meshes.

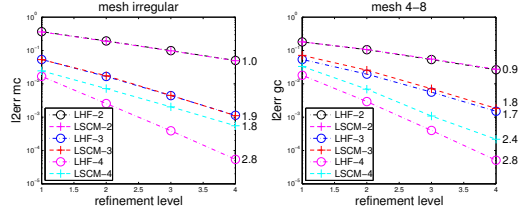


Fig. 14. L_2 errors in computed mean (left) and Gaussian (right) curvatures for boundary vertices for noise-free open surface meshes.

Table 4

Input position errors of torus meshes extracted using isosurface function of MATLAB.

#volume cells	#surface verts.	average edge length	error		convergence rate	
			L_2	L_∞	L_2	L_∞
16^3	320	0.11	3.60e-3	9.30e-3	—	—
32^3	1760	0.048	1.12e-3	3.47e-3	1.69	1.42
64^3	7320	0.023	2.59e-4	9.03e-4	2.11	1.94
128^3	31136	0.011	6.71e-5	2.25e-4	1.95	2.00
256^3	122464	0.0057	1.65e-5	5.70e-5	2.03	1.98

ods do not rely on symmetry for convergence, although they can benefit from symmetry and statistical error cancellation in practice.

5.3 Experiments for Noisy Surface Meshes

In many practical situations, the input points are not precisely on an analytical surface but contain noise or input errors. An example is the meshes extracted from isosurfaces, for which the vertex positions contain inherent errors. Another example is meshes obtained from solutions of numerical computations. We now assess the methods for these situations.

To study the effect of numerical errors, we use the meshes obtained from the isosurface function in MATLAB as in Section 5.1, but unlike in Section 5.1 we do not project the vertices onto the true torus. Table 4 shows the errors in the vertex positions for the surface mesh extracted using the isosurface function of MATLAB. We compute the position error for each vertex based on its distance to the torus, and report both L_2 and L_∞ errors for five different mesh resolutions. From Table 4, it seems that the isosurface function in MATLAB is second-order accurate, so convergence of curvature is not guaranteed theoretically.

For improved noise resistance, we increased the neighborhood size by half a ring for LHF, LOP, and LSCM. Figure 15 shows the L_2 and L_∞ errors of the computed normals for degree-2 fittings, and Figure 16 shows the L_2 errors in the computed mean and Gaussian curvatures, respectively. It can be seen that Xu-2 had the largest errors, nearly two orders of magnitude larger than the other methods. This again is primarily because Xu’s method uses only 1-ring neighbors. For the other three methods, the L_2 errors of the normal directions converged at about second order, while the L_∞ errors converged slower than second order. The L_2 errors of the curvatures converged at about first order. These rates are higher than the theoretical predictions for noisy data, probably due to statistical error cancellation. We note that when a first-order isosurface algorithm is used, we observed no convergence and sometimes even divergence for the curvatures.

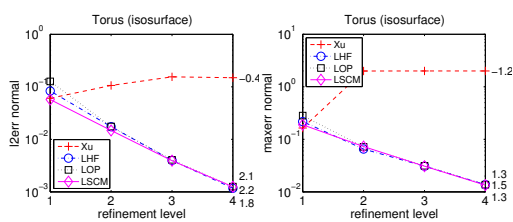


Fig. 15. Comparison of L_2 (left) and L_∞ (right) errors in computed normals using degree-2 fittings for noisy torus.

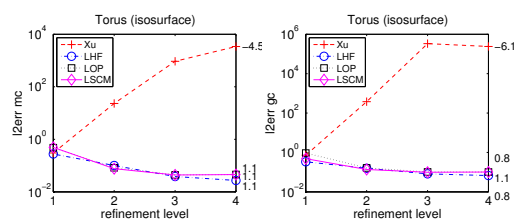


Fig. 16. Comparison of L_2 errors in computed mean (left) and Gaussian curvatures using degree-2 fittings for noisy torus.

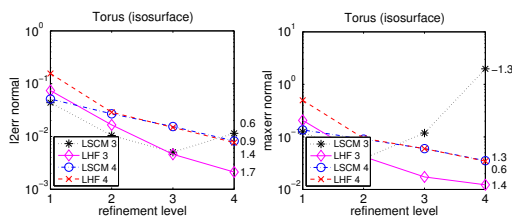


Fig. 17. Comparison of L_2 (left) and L_∞ (right) errors in computed normals using degree-3 and 4 fittings for noisy torus.

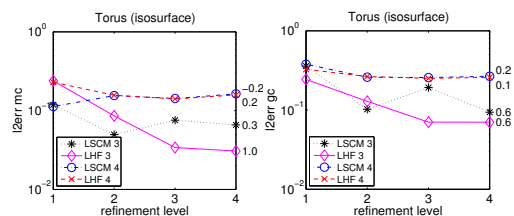


Fig. 18. Comparison of L_2 errors in computed mean (left) and Gaussian curvatures using degree-3 and 4 fittings for noisy torus.

For completeness, we also present the results of cubic and quartic fittings in Figures 17 and 18 for the computed normals and curvatures, respectively. The results of these higher-degree fittings were worse than those for quadratic fittings, because a degree- d fitting requires the input errors to be order $d + 1$ or higher for optimal convergence.

The input noise in the preceding test was due to numerical errors in isosurfacing, which are beyond our control. To verify our analysis, we report experiments with controlled errors, where the noise is added by perturbing the surface. We use the open surface meshes in this test, and for each vertex we perturb it by adding a noise of αh^k along the normal direction, where α is a random number between 0 and 0.1

with Gaussian distribution, h is the average edge length, and k is chosen to be 2 or 3. For brevity, we report results only for quadratic fittings.

Figure 19 shows the L_2 and L_∞ errors of computed normals for $O(h^3)$ perturbation, and Figure 20 shows the corresponding results for $O(h^2)$ perturbations. Figure 21 shows the L_2 errors of the computed mean and Gaussian curvatures for $O(h^3)$ perturbation, and Figure 22 shows the corresponding results for $O(h^2)$ perturbations. We excluded boundary vertices when computing errors for Xu-2 but included boundary vertices for the other methods. Note that for the normals, each method performed nearly identically for the two different orders of perturbations. Xu-2 again delivered slightly better results than the others for the 4-8 mesh due to its use of a smaller neighborhood. However, for curvatures Xu-2 failed to converge again due to insufficient number of points, and the other methods delivered higher than linear convergence for $O(h^3)$ perturbations and generally lower than linear convergence for $O(h^2)$ perturbations, consistent with the theoretical analysis.

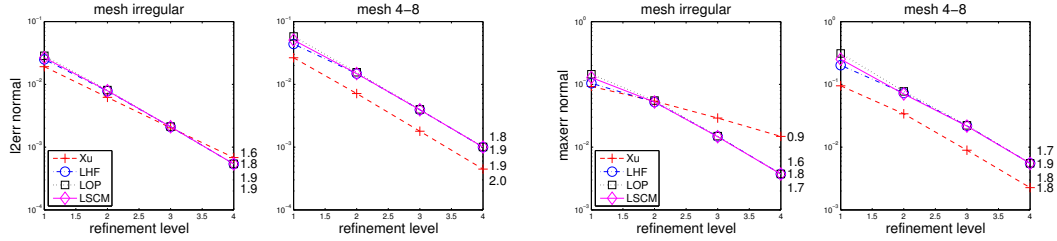


Fig. 19. Comparison of L_2 (left) and L_∞ (right) errors of computed normals using degree-2 fittings for noisy open surface with third-order perturbation.

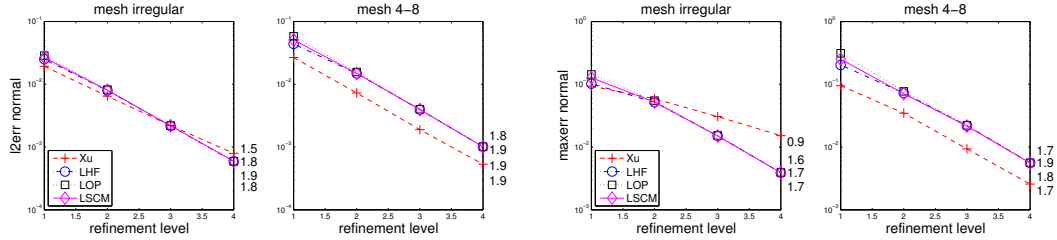


Fig. 20. Comparison of L_2 (left) and L_∞ (right) errors of computed normals using degree-2 fittings for noisy open surface with second-order perturbation.

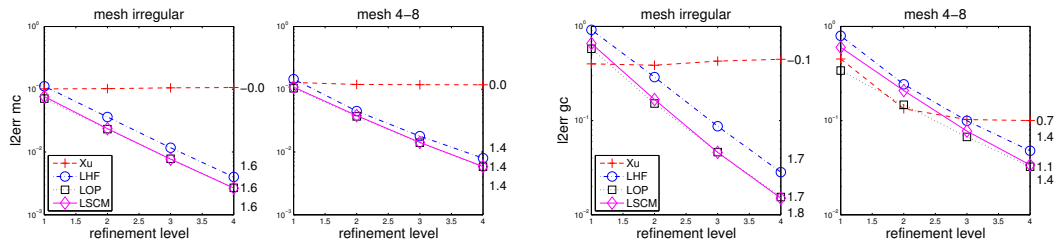


Fig. 21. Comparison of L_2 errors of computed mean (left) and Gaussian (right) curvatures using degree-2 fittings for noisy open surface with third-order perturbation.

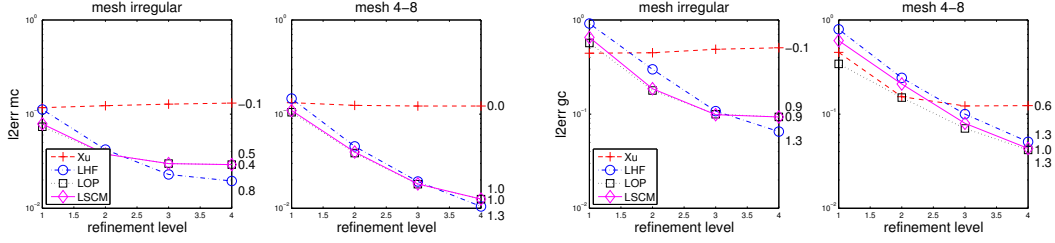


Fig. 22. Comparison of L_2 errors of computed mean (left) and Gaussian (right) curvatures using degree-2 fittings for noisy open surface with second-order perturbation.

Table 5

Comparison of execution times in seconds of different methods for torus mesh.

Mesh	#verts.	#tris.	Xu-2	LOP-2	LHF-2	LHF-3	LHF-4	LSCM-2	LSCM-3	LSCM-4
1	1338	2676	0.035	0.011	0.0061	0.013	0.028	0.20	0.36	0.66
2	5246	10492	0.12	0.039	0.023	0.050	0.11	0.81	1.43	2.65
3	21156	42312	0.51	0.16	0.096	0.20	0.44	3.32	5.74	10.58
4	85208	170416	2.06	0.65	0.38	0.83	1.81	13.66	23.21	43.16

5.4 Comparison of Efficiency

A key practical consideration in choosing a proper method is runtime efficiency. Table 5 reports the execution times of the different methods for the well-shaped torus mesh. We implemented the methods of Xu, LOP, and LHF in MATLAB and then converted the MATLAB code into C using Agility MCS (www.agilityds.com). For Xu’s method, we used the singular value decomposition procedures in LAPACK (Anderson et al., 1999). For least-squares conformal mapping, we used the C++ implementation of the parameterization algorithm in CGAL and used the same C code as for LOP for the other numerical computations. All the codes were compiled using gcc 4.2.4, with optimization enabled. We performed the tests on a Linux computer with a 3GHz Intel Duo Core Pentium 4 processor and 2GB of RAM.

As can be seen in the table, LSCM was roughly an order of magnitude more expensive than the others. This is partially due to the overhead of the data structures in CGAL and the high cost for solving linear system for the conformal parameterization. With a better optimized procedure for conformal parameterizations, LSCM may become more competitive in cost. Xu’s method was the second most expensive, even though it uses only 1-ring neighbors. This is primarily because of its use of SVD, which is substantially more expensive than QR factorization. The most efficient algorithm was LHF. The effective performance of LHF-2 is about 450 thousand triangles per second, and the cost approximately doubles when the degree of fitting is increased by one. For comparison, we also tested `Jet_fitting_3` of Cazals and Pouget (2008) on the same computer, and the effective performance of

Jet_fitting_3 for second-order fitting was about 28 thousand triangles per second, consistent with their results reported in (Cazals and Pouget, 2008). Our algorithm and implementation is more than 16 times faster than Jet_fitting_3, probably because of their use of SVD as well as the the overhead of the data structures in CGAL. LOP slightly under-performs LHF, because the linear system for LOP involves three right-hand-side vectors (instead of one in LHF) and also the formulas for parametric surfaces are more complex than those for height functions. From these experimental results, it seems that LHF offers the best combination of efficiency, accuracy, simplicity, and robustness.

6 Conclusions

In this paper, we analyzed the numerical properties of parameterization-based computation of first and second-order differential quantities for continuous and discrete surfaces, and also compared a number of different methods for surface meshes in terms of accuracy, stability and efficiency. We studied, both theoretically and experimentally, the impact of a number of key factors, including input errors, parameterization methods, numerical solvers, as well as the formulas for continuous functions. From our study, we draw the following conclusions. First, guaranteed convergence of curvature computation requires the input points be third or higher-order accurate. This high-order accuracy requirement is fortunately satisfied by meshes generated from CAD models and solutions of high-order numerical simulations. Our second and related conclusion is that meaningful (and even converging) curvature estimations can be obtained from noisy input (with lower than third-order accurate input) for purposes such as visualization or shape analysis when using stable least squares approximations. In practice, these estimations are fairly close to the true results, although they are not guaranteed to converge to the true solutions during mesh refinement. Thirdly, we observed that approximate conformal parameterizations are smooth enough for quadratic and cubic fittings but are insufficient for high-degree fittings. In contrast, local height function or local orthogonal projection are applicable in higher degree fittings and can deliver high-order convergence rates. Finally, it was evident that the choices of numerical solvers can have significant impacts on the accuracy and efficiency on least squares polynomial fitting. In particular, QR factorization with safeguard turned out to be more efficient (by roughly a factor of five) and more accurate (especially for high order methods) compared to SVD. We believe these conclusions will be valuable in helping readers setting proper expectations and choosing proper methods for computing differential quantities for surface meshes. As a future research direction, we will investigate the use of local height functions or local orthogonal projection in geometric flows for geometric modeling and physics-based simulations.

Acknowledgments

This work was supported by the National Science Foundation under award number DMS-0713736. We thank the support of the Department of Applied Mathematics and Statistics of Stony Brook University. We also thank the anonymous reviewers for their helpful comments.

References

- Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D., 1999. LAPACK User's Guide, 3rd Edition. SIAM.
- Borrelli, V., Cazals, F., Morvan, J.-M., 2003. On the angular defect of triangulations and the pointwise approximation of curvatures. *Comput. Aided Geom. Des.* 20 (6), 319–341.
- Cazals, F., Pouget, M., 2005. Estimating differential quantities using polynomial fitting of osculating jets. *Comput. Aid. Geom. Des.* 22, 121–146.
- Cazals, F., Pouget, M., 2008. *Jet_fitting_3*: A generic C++ package for estimating the differential properties on sampled surfaces via polynomial fitting. *ACM Trans. Math. Softw.* 35 (3), 1–20.
- Cohen-Steiner, D., Morvan, J.-M., 2003. Restricted Delaunay triangulations and normal cycle. In: *Proc. of 19th Annual Symposium on Computational Geometry*. pp. 312–321.
- Desbrun, M., Meyer, M., Alliez, P., 2002. Intrinsic parameterizations of surface meshes. In: *Proceedings of Eurographics 2002 Conference*. pp. 209–218.
- do Carmo, M. P., 1976. *Differential Geometry of Curves and Surfaces*. Prentice-Hall.
- Floater, M. S., 2003. Mean value coordinates. *Comput. Aid. Geom. Des.* 20, 19–27.
- Floater, M. S., Hormann, K., 2005. Surface parameterization: a tutorial and survey. In: *Dodgson, N. A., Floater, M. S., Sabin, M. A. (Eds.), Advances in Multiresolution for Geometric Modelling*. Springer Verlag, pp. 157–186.
- Gatzke, T., Grimm, C., 2006. Estimating curvature on triangular meshes. *Int. J. Shape Modeling* 12, 1–29.
- Goldfeather, J., Interrante, V., 2004. A novel cubic-order algorithm for approximating principal direction vectors. *ACM Trans. Graph.* 23, 45–63.
- Golub, G. H., Van Loan, C. F., 1996. *Matrix Computation*, 3rd Edition. Johns Hopkins.
- Gotsman, C., Gu, X., Sheffer, A., 2003. Fundamentals of spherical parameterization for 3d meshes. *SIGGRAPH* 22, 358–363.
- Grinspun, E., Gingold, Y., Reisman, J., Zorin, D., 2006. Computing discrete shape operators on general meshes. *Eurographics (Computer Graphics Forum)* 25, 547–556.

- Hildebrandt, K., Polthier, K., Wardetzky, M., 2006. On the convergence of metric and geometric properties of polyhedral surfaces. *Geometria Dedicata* 123, 89–112.
- Jiao, X., Zha, H., 2008. Consistent computation of first- and second-order differential quantities for surface meshes. In: *ACM Solid and Physical Modeling Symposium*. ACM, pp. 159–170.
- Lancaster, P., Salkauskas, K., 1986. *Curve and Surface Fitting: An Introduction*. Academic Press.
- Langer, T., Belyaev, A. G., Seidel, H.-P., 2007. Exact and interpolatory quadratures for curvature tensor estimation. *Comput. Aid. Geom. Des.* 24, 443–463.
- Lévy, B., Petitjean, S., Rry, N., Maillot, J., 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics* 21, 362–371.
- Meek, D. S., Walton, D. J., 2000. On surface normal and Gaussian curvature approximations given data sampled from a smooth surface. *Comput. Aid. Geom. Des.* 17, 521–543.
- Meyer, M., Desbrun, M., Schröder, P., Barr, A., 2002. Discrete differential geometry operators for triangulated 2-manifolds. In: Hege, H.-C., Polthier, K. (Eds.), *Visualization and Mathematics*. Vol. 3. Springer, pp. 34–57.
- Peters, J., Reif, U., 2008. *Subdivision Surfaces*. Springer.
- Petitjean, S., 2002. A survey of methods for recovering quadrics in triangle meshes. *ACM Comput. Surveys* 34, 211–262.
- Pinkall, U., Polthier, K., 1993. Computing discrete minimal surfaces and their conjugates. *Exper. Math.* 2, 15–36.
- Rusinkiewicz, S., 2004. Estimating curvatures and their derivatives on triangle meshes. In: *Proc. of 2nd International Symposium on 3D Data Processing, Visualization and Transmission*. pp. 486–493.
- Saboret, L., Alliez, P., Lévy, B., 2007. Planar parameterization of triangulated surface meshes. In: Board, C. E. (Ed.), *CGAL User and Reference Manual*, 3rd Edition. Online at <http://www.cgal.org>.
- Taubin, G., 1995. Estimating the tensor of curvature of a surface from a polyhedral approximation. In: *Proc. of Int. Conf. on Computer Vision*. pp. 902–907.
- Trefethen, L. N., Bau, D., 1997. *Numerical Linear Algebra*. SIAM.
- Tutte, W. T., 1963. How to draw a graph. *Proceedings London Mathematical Society* 13, 743–768.
- Wardetzky, M., 2007. Convergence of the cotan formula - an overview. In: Bobenko, A. I., Sullivan, J. M., Schröder, P., Ziegler, G. (Eds.), *Discrete Differential Geometry*. Birkhäuser Basel, pp. 275–286.
- Xu, G., 2004. Convergence of discrete Laplace-Beltrami operators over surfaces. *Comput. Math. Appl.* 48, 347–360.
- Xu, G., 2007. Consistent approximation of some geometric differential operators. Tech. rep., Institute of Computational Mathematics, Chinese Academy of Sciences, research Report No. ICM-07-02.