Eurographics Symposium on Geometry Processing (2006)
Konrad Polthier, Alla Sheffer (Editors)

Error Bounds and Optimal Neighborhoods for MLS Approximation

Yaron Lipman Daniel Cohen-Or David Levin

Tel-Aviv University

Abstract

In recent years, the moving least-square (MLS) method has been extensively studied for approximation and reconstruction of surfaces. The MLS method involves local weighted least-squares polynomial approximations, using a fast decaying weight function. The local approximating polynomial may be used for approximating the underlying function or its derivatives. In this paper we consider locally supported weight functions, and we address the problem of the optimal choice of the support size. We introduce an error formula for the MLS approximation process which leads us to developing two tools: One is a tight error bound independent of the data. The second is a data dependent approximation to the error function of the MLS approximation. Furthermore, we provide a generalization to the above in the presence of noise. Based on the above bounds, we develop an algorithm to select an optimal support size of the weight function for the MLS procedure. Several applications such as differential quantities estimation and up-sampling of point clouds are presented. We demonstrate by experiments that our approach outperforms the heuristic choice of support size in approximation quality and stability.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Surface approximation, Point clouds, Meshes, Differential quantities estimation

1. Introduction

A fundamental problem in surface processing is the reconstruction of a surface or estimating its differential quantities from scattered (sometimes noisy) point data [HDD*92]. A common approximation approach is fitting local polynomials, explicitly [ABCO*01], or implicitly [OBA*03], to approximate the surface locally. This approach can be realized by the moving least-squares (MLS) method, where, for each point x, a polynomial is fitted, in the least-squares sense, using neighboring points x_i. This technique works well assuming that the surface is smooth enough [Lev98, Wen01]. The local polynomial fitting enables up and down sampling of the surface [ABCO*01], estimating differential quantities such as normal or curvature data [CP05], and performing other surface processing operations [PKKG03].

In recent years, the MLS technique has gained much popularity, and the method is now well studied. However, proper choice of neighboring points x_i to be used in the approximation still remains an important open problem. Apparently, there is a large degree of freedom in choosing the points participating in the approximation since the number of data points is usually very large, while the degree of polynomial is usually very small. Naturally, one would like to make use of these large degrees of freedom to achieve the "best" approximating polynomial. Several researchers [ABCO*01, PGK02, PKKG03] have used different heuristic approaches, such as using a neighborhood proportional to the local sampling density measured via the radius of the ball containing the K nearest neighbors, or using Voronoi triangulation [FR01] to choose the neighboring points. In this paper, we compare a heuristic method in the spirit of these approaches with a new approach based on error analysis.

Since the problem of choosing the points to be used in the approximation is closely related to multivariate interpolation, it is known that the choice of the points depends on the geometry of the points, and not only their number, as in the sampling density based approaches. However, the geometric configuration of points which admits a stable interpolation/approximation problem is a hard problem in the field of multivariate polynomial approximation [Bos91, SX95, GS00]. Loosely speaking, a 'stable' points'

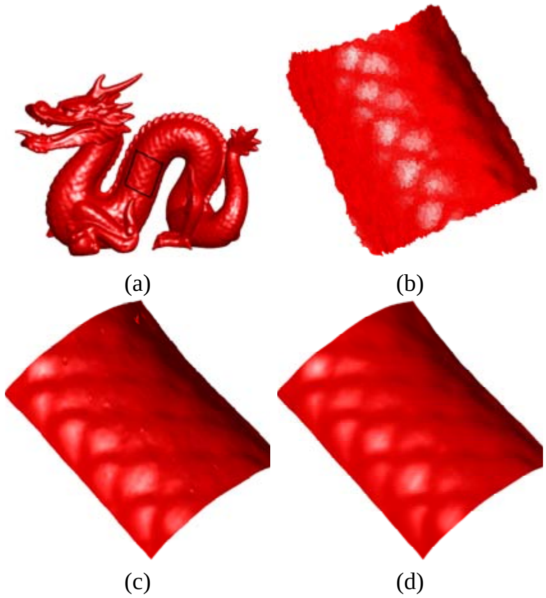


Figure 1: *Resampling of a noisy surface. A patch of the dragon model (a) with additional white noise (b) is resampled with two methods: Using the local density (heuristic) to determine the local neighborhood yields some noticeable artifacts (c). The proposed method, assuming the maximal value of the noise is known, faithfully reconstruct the surface.*

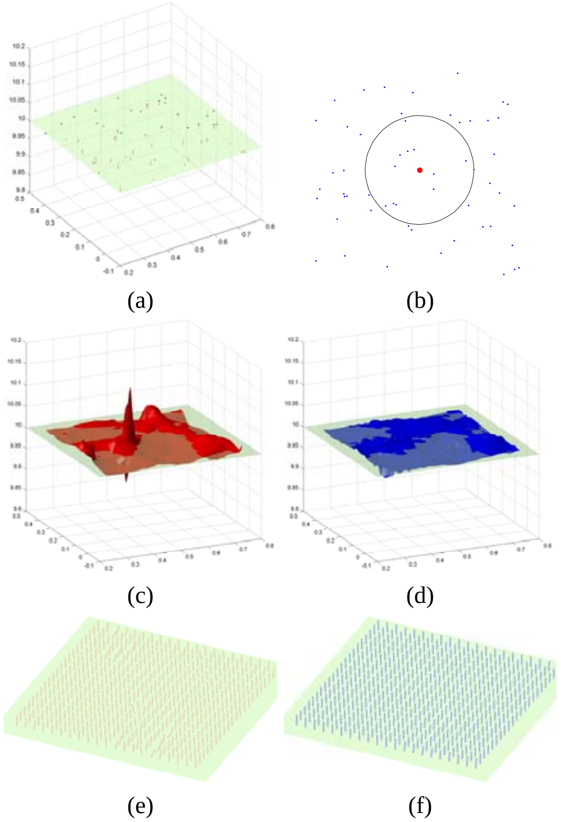


Figure 2: *An example of the amplifying effect of noise in the data. In (a) the 'true surface', i.e., a plane, and the sampled points with white noise errors of maximal magnitude* 0.015*. In (b), the sample data (blue) and a red point where an approximation is sought. The black circle indicates the points used by the heuristic method. In (c) the surface reconstructed by this heuristic, note that the high peak is created at the place indicated by the red point in (b). In (d) the proposed algorithm for neighborhood was used. In (e) surface (plane) normals were estimated using the heuristic algorithm and in (f) the normals were estimated using the new proposed algorithm. We used approximation by quadratics, i.e.,* $N = 2$*.*

configuration is such that is 'far' from a degenerate point configuration, where a degenerate point configuration lies on an algebraic curve of the dimension of the interpolation space.

Preliminary example. Consider the points $X_h = \{h(\cos j2\pi/6, \sin j2\pi/6), j = 0,1,..,5\}$. X forms a degenerate point configuration for bi-variate quadratic interpolation. Although this point set seems nicely distributed around $(0,0)$, quadratic interpolation cannot be used for approximation at $(0,0)$, no matter how small we take h. This can be seen by taking, for example, the values associated with $x_i \in X_h$ to be zeros and noting that both $f_1(x) \equiv 0$ and $f_2(x) = (h^2 - x_1^2 - x_2^2)/h^3$ solve the interpolation problem, but $|f_1(0) - f_2(0)| \sim 1/h$. Another phenomenon is shown in Figure 1 where a noisy part (b) of the dragon model (a) was re-sampled using a density based heuristic (c), which caused undesired artifacts, and in (d) the new proposed method was used. In Figure 2, another example of this kind is shown. As elaborated in Section 3.5, this configuration of points causes amplification of the noise level in the data by a factor of ~ 10 when using MLS to approximate the surface value at the red point in (b), and a factor of ~ 100 when this point configuration is used to approximate $\partial_x f$ at that point. The resulting approximations, i.e., point evaluation (c), and normal approximation (e), are useless at this point. Using the new approach presented here yields bounded errors which don't exceed much the initial noise level in the data, see (d) and (f).

Recognizing that there are no simple nor intuitive rules which distinguish 'bad' point sets from 'good' point sets for approximation at a given point, we take a different route to decide which neighbors should be used in the approximation process: We introduce an error formula which provides means to understand and to evaluate the approximation quality of MLS approximation. From it, we derive two tools. The first, is a tight bound independent of the data, assuming only that the local corresponding derivatives of the function are bounded. The second is a data dependent approximation to the error function of the MLS interpolant. We examine the practical usage of these tools and compare them to a carefully chosen heuristic method.

Based on the above bounds, we develop an algorithm to select an optimal radial neighborhood for the MLS procedure. Loosely speaking, since the underlying surface from which the sampled points are taken is unknown, the optimality of the chosen neighborhoods is in the sense of the approximation error having the lowest error bound.

MLS approximations are used in a variety of cases, but the problem is always reduced to the functional case, by defining some parameter domain [Lev03, ABCO*01]. Thus, for the error analysis, it is enough to consider the functional case.

We develop the various theoretical error terms and bounds in Section 3, and based on these results, in Section 4, we introduce an algorithm for selecting the optimal neighborhood. In Section 5 we derive a heuristic rule which we use for comparison. In the following section, we briefly describe the MLS technique, and define the terms and notation to be used in the paper. In Section 6 we present some numerical experiments, and in Section 7 we conclude.

2. Background

Surfaces or 2-manifolds embedded in $\mathbb{R}^3$ are most commonly represented by a set of spacial points, with neighboring relations (meshes) or without (point clouds). A common way to estimate the value of the surface in a new point, or to estimate differential quantities of the surface at any point, is by fitting a local polynomial and extracting it's value or derivatives.

In order to reduce the problem to the functional case a local parameter plane is constructed, and the local polynomial is defined over this parameter space. Eventually, one ends up with the problem of fitting a polynomial $p \in \Pi$, where Π is some polynomial subspace, given data points $(x_i, f(x_i)) \in \Omega \times \mathbb{R}$, $i = 1,..,I$, where Ω is a domain in $\mathbb{R}^d$. The goal is to approximate a functional L^x at a point $x \in \Omega$, where L^x can be a function evaluation at x or a derivative evaluation, e.g., $L^x(f) = f(x)$ or $L^x(f) = (\partial_{xy} f)(x)$. A common way to do it is by fitting the polynomial locally in the least-squares sense:

$$\min \left\{ \sum_{i=1}^{I} \left(f(x_i) - p(x_i) \right)^2 w(\|x_i - x\|) \quad , \quad p \in \Pi \right\}, \tag{1}$$

where $w(r)$ is a radial weight function. When the minimizer polynomial p is achieved, the approximation functional L^x is applied to it to form the approximation

$$L^x(f) \approx L^x(p). \tag{2}$$

As showed in [Lev98],

$$L^x(p) = \sum_{i=1}^{I} a_i f(x_i),$$

where a_i are the solution to the constrained quadratic minimization problem

$$\left\{ \min \sum_{i=1}^{I} w(\|x - x_i\|)^{-1} |a_i|^2 \; s.t. \quad \sum_{i=1}^{I} a_i p(x_i) = p(x), \; \forall p \in \Pi \right\}.$$

The weight function w is usually chosen to ensure fast decay of the magnitude of the a_i for points distant from the evaluation point x. The decay rate is heuristically chosen to be as fast as possible while keeping enough points in the significant weights area to keep the problem well-posed. Furthermore, a smooth weight function implies smooth approximation. In this paper we have chosen to use the weight function of finite support [Lev98], $w(r) = w_h(r)$, where

$$w_h(r) = e^{-\frac{r^2}{(h-r)^2}} \chi_{[0,h)}(r). \tag{3}$$

The main objective of this paper is to present an algorithm for choosing the support size h which best assures a minimal approximation error using the procedure (1)-(2). This is accomplished in two independent ways: First by minimizing a novel, tight, local error bound formula. This procedure also supplies a bound on the error which is achieved in the approximation process, given that a bound on local corresponding derivatives of f is known. Second, a novel approximation of the error in the MLS approximation is constructed, and the best support size h is chosen as before. The latter generally performs better than the former.

3. Error analysis

3.1. Settings

The settings of the problem consists of a data set $(x_i, f(x_i))$, $X = \{x_i\}_{i=1}^{I} \subset \Omega \subset \mathbb{R}^d$, sampled from a smooth function $f \in C^k(\Omega)$, and another point x where an approximation $L^x f = D^\alpha f(x)$, where $D^\alpha = \partial_{x^{(1)}}^{\alpha_1} \cdot \ldots \cdot \partial_{x^{(d)}}^{\alpha_d}$, is sought. Denote by N the degree of the polynomials used as the approximation space. $J = \binom{N+d}{d}$ is the dimension of the space Π_N of d-variate polynomial of degree N. Also define $p_1, p_2, ..., p_J \in \Pi$ to be the standard basis of Π_N shifted to x, that is, $\{(\cdot - x)^\alpha\}_{|\alpha| \le N}$, where we use the multi-index notation $\alpha = (\alpha_1, ..., \alpha_d)$, $\alpha! = \alpha_1! \cdot \ldots \cdot \alpha_d!$, $|\alpha| = \alpha_1 + \ldots + \alpha_d$, and for $x = (x^{(1)}, ..., x^{(d)})$ $x^\alpha = (x^{(1)})^{\alpha_1} \ldots (x^{(d)})^{\alpha_d}$. We also define the generalized Vandermonde matrix E by $E_{i,\beta} = p_\beta(x_i)$, $i = 1,..,I$, $|\beta| \le N$.

Denote the subset $X_h = X \cap B_h(x)$, where $B_h(x)$ denotes a ball of radius h with center x, and let $I = |X_h|$, the number of data points in $B_h(x)$. Then, for a fixed h, we define the approximation $D^\alpha f(x) \approx D^\alpha p(x)$, where $p \in \Pi_N$, is defined by (1), and $w = w_h$ defined by (3).

Let us introduce some test functions. The first one is taken from [Fra82],

$$\begin{aligned} F_1 \;=\; & \frac{30}{4} e^{-\frac{(9x+5)^2+(9y+5)^2}{16}} + \frac{30}{4} e^{-\frac{(9x+11)^2}{196} - \frac{9y+11}{20}} \\ + \; & \frac{10}{2} e^{-\frac{(9x-5)^2+(9y+3)^2}{16}} - \frac{10}{5} e^{-\frac{(9x+1)^2+(9y-5)^2}{4}} . \end{aligned} \tag{4}$$

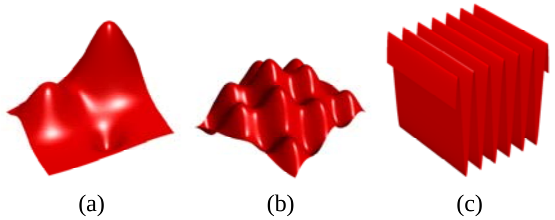


Figure 3: *The test function used in the paper. (a) is the graph of F_1, (b) of F_2 and (c) of F_3.*

$$\begin{aligned} F_2 &= 0.3\cos(8x)\sin(6y)+e^{-x^2-y^2}. && (5)\\ F_3 &= \cos(20x). && (6)\end{aligned}$$

In Figure 3, we plotted the graphs of these test functions. These functions were selected since they seem to represent well several smooth surface types: F_1 is a standard test function and has been used in numerous papers. F_2 has interesting 'details', and F_3 is an anisotropic surface with very high derivatives.

3.2. Pointwise error in the MLS approximation

In this section we lay out the formula for the error in the MLS approximation which forms the basis for all latter developments in the paper:

Theorem 3.1 Denote by p the fitted polynomial defined by (1) to the data $(X_h, f(X_h)) \subset \Omega \times I\!R$, sampled from a smooth function $f \in C^{N+1}(\Omega)$, then for $x \in \Omega$,

$$R(x) = D^\alpha p(x) - D^\alpha f(x) = \qquad (7)$$
$$\frac{\alpha!}{(N+1)!}\sum_{i,\nu} D^\nu f(\eta_i(x_i-x)+x)(x_i-x)^\nu \frac{\det\left(E^tWE_{\alpha\leftarrow e_i}\right)}{\det\left(E^tWE\right)}$$

where $\sum_{i,\nu}$ stands for $\sum_{|\nu|=N+1}\sum_{i=1}^I$, $0 \le \eta_i \le 1$, and E is the Vandermonde matrix $E_{i,\beta} = p_\beta(x_i)$. $E_{\alpha\leftarrow e_i}$ denotes the matrix E where the α column is replaced by the standard basis vector $e_i = \delta_{j,i}$. The weight matrix W is defined by $W = diag(w_h(\|x_1-x\|),\ldots,w_h(\|x_I-x\|))$.

Proof. The proof relies on the polynomial reproduction property of the Least-Squares method and is based upon local Taylor expansions as approximations of f.

First, w.l.o.g, we may assume $x=0$. Denote by $p_1,\ldots,p_J$ the standard basis of the multivariate polynomials of degree $\le N$, that is $p_\alpha(x) = x^\alpha$, $|\alpha| \le N$. Next, writing $p = \sum_\beta c_\beta p_\beta$, leads to

$$D^\alpha p(0) = \sum_\beta c_\beta D^\alpha p_\beta(0) = \sum_\beta c_\beta \alpha! \delta_{\alpha,\beta} = \alpha! c_\alpha.$$

The multivariate Vandermonde matrix E is ordered by the multi-index β, i.e., $E_{i,\beta} = p_\beta(x_i)$, as we do also for the vector $c = \{c_\beta\}_{|\beta|\le N}$ of the unknown coefficients. The fitted polynomial p is then defined as the solution in the least-squares sense. That is, c satisfies the normal equations:

$$E^tWEc = E^tWF. \qquad (8)$$

Using Taylor expansion,

$$f(x_i) = \sum_{|\nu|\le N}\frac{D^\nu f(0)}{\nu!}x_i^\nu + \frac{1}{(N+1)!}\sum_{|\nu|=N+1} D^\nu f(\eta_i x_i)x_i^\nu,$$

where $1 \le \eta_i \le \nu$. Hence, the vector F can be written as

$$F = \sum_{|\nu|\le N}\frac{D^\nu f(0)}{\nu!}E_\nu + \frac{1}{(N+1)!}\sum_{|\nu|=N+1} Q_\nu E_\nu,$$

where $Q_\nu = diag(D^\nu f(\eta_1 x_1),\ldots,D^\nu f(\eta_I x_I))$, and E_ν denotes the ν column vector of matrix E. Then, For the solution of (8) we have,

$$c_\nu = \frac{D^\nu f(0)}{\nu!} + \frac{1}{(N+1)!}\left(\sum_{|\nu|=N+1}(E^tWE)^{-1}E^tWQ_\nu E_\nu\right)_\nu.$$

Next, $E^tWQ_\nu E_\nu = \sum_{i=1}^I (E^tW)_i D^\nu f(\eta_i x_i) p_\nu(x_i)$, with $p_\nu(x_i) = x_i^\nu$, and by Cramer's rule and the linearity of the determinant $\left((E^tWE)^{-1}E^tWQ_\nu E_\nu\right)_\nu =$

$$\sum_{i=1}^I D^\nu f(\eta_i x_i)x_i^\nu \frac{\det(E^tWE_{\nu\leftarrow e_i})}{\det(E^tWE)}.$$

Finally we get for $\nu = \alpha$, $\alpha! c_\alpha - D^\alpha f(0) =$

$$\frac{\alpha!}{(N+1)!}\sum_{|\nu|=N+1}\sum_{i=1}^I D^\nu f(\eta_i x_i)x_i^\nu \frac{\det(E^tWE_{\nu\leftarrow e_i})}{\det(E^tWE)},$$

where $|\nu| = N+1$ and $i = 1,..,n$.

Corollary 3.1 Denote by p the fitted polynomial defined by (1) to the data $(X_h, f(X_h)) \in \Omega \subset I\!R^d \times I\!R$, sampled from a smooth function $f \in C^{N+1}(\Omega)$, then the following is a *tight* error bound,

$$|R(x)| \le \frac{\alpha! C}{(N+1)!}\sum_{i,\nu}|x_i - x|^\nu \left|\frac{\det\left(E^tW(E)_{\alpha\leftarrow e_i}\right)}{\det\left(E^tWE\right)}\right|,$$

where C is the bound: $\max_{|\nu|=N+1, x\in\Omega}|D^\nu f(x)| \le C$.

3.3. Data Independent Bound

For a given support size h, an error bound for the polynomial fitting procedure based on the data X_h can be calculated via the tight bound in Corollary (3.1). We define the bounding function B_α by

$$B_\alpha = B_\alpha(x, X_h) = \frac{\alpha!}{(N+1)!}\sum_{\beta,i}|x_i - x|^\beta \frac{|\det(E^tWE_{\alpha\leftarrow e_i})|}{|\det(E^tWE)|}, \qquad (9)$$

where, as before, $\sum_{i,\beta}$ is a short notation for $\sum_{|\nu|=N+1}\sum_{i=1}^I$.

From the computational point of view, in order to compute (9), we note that $\frac{\det(E^tWE_{\alpha\leftarrow e_i})}{\det(E^tWE)}$ is the α coordinate of the solution to the linear system:

$$E^t W E c = (E^t W)_i,$$

where $(E^t W)_i$ denotes the i-th column of matrix $E^t W$. Therefore, in the calculation of (9), one should calculate the solution V to $E^T W E V = E^t W$, and then set

$$V_{\alpha,i} = \frac{\det(E^t W E_{\alpha \leftarrow e_i})}{\det(E^t W E)}. \tag{10}$$

Then formula (9) reduces to,

$$B_\alpha(x, X_h) = \frac{\alpha!}{(N+1)!} \sum_{\beta,i} |x_i - x|^\beta |V_{\alpha,i}|. \tag{11}$$

3.4. Data dependent error approximation

In this section we construct a data dependent approximation to the error function in the MLS approximation for $f \in C^{N+2}(\Omega)$. This error approximation uses the known values at the points X_h in order to better approximate the error in the approximation (7). In particular we note that $D^\nu f(\eta_i(x_i - x) + x) = D^\nu f(x) + O(h)$, for $|\nu| = N+1$, where h is the support size used. Therefore, Eq. (7) can be written as:

$$R(x) = \frac{\alpha!}{(N+1)!} \sum_{i,\nu} D^\nu f(x)(x_i - x)^\nu V_{\alpha,i} + O(h^{N+2-|\alpha|}).$$

The idea is to improve the error estimate by approximating the unknown values $f_\nu = D^\nu f(x)$, $|\nu| = N+1$. Such approximations can be derived by using the error formula at points x_k near x: We have

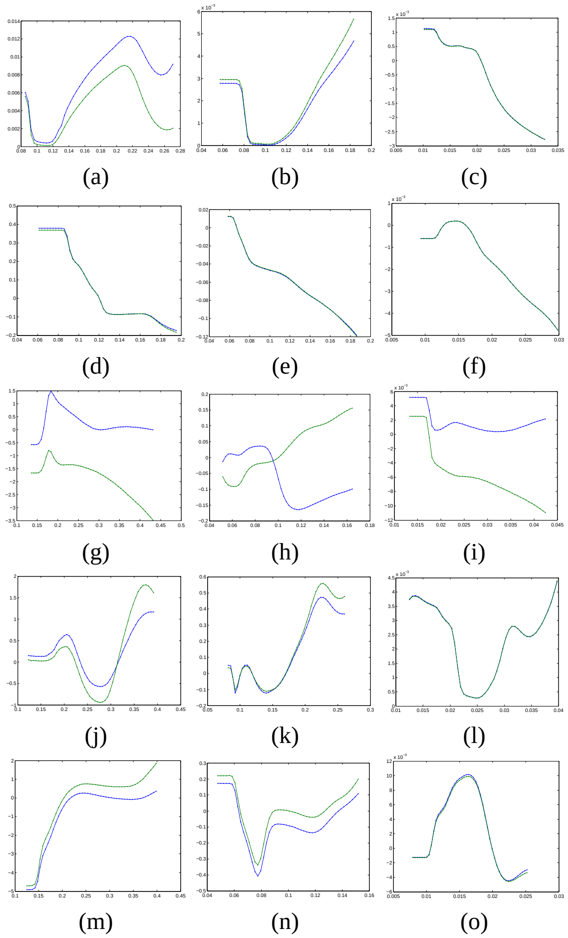
$$p(x_k) - f(x_k) = R(x_k) = \frac{\alpha!}{(N+1)!} \sum_\nu f_\nu \left(\sum_i (x_i - x_k)^\nu V^k_{\alpha,i} \right), \tag{12}$$

where $V^k_{\alpha,i}$ are defined similar to $V_{\alpha,i}$ in (10), using the shifted basis $\{(\cdot - x_k)^\alpha\}_{|\alpha| \leq N}$. The points $\{x_i\}$ are taken from a ball of radius $h = 3h_J$ centered at x, where h_J denotes the radius of the ball which contains the J nearest points to x. The points $\{x_k\}$ are taken as the $2J$ nearest points to x. The system (12), of $2J$ equations and $N+2$ unknowns f_ν is solved in the least-squares sense.

Plugging the resulting estimated values f_ν into the error bound (7) we get an approximation of the error term:

$$\tilde{R}_\alpha = \tilde{R}_\alpha(x, X_h) = \frac{\alpha!}{(N+1)!} \sum_{i,\nu} f_\nu (x_i - x)^\nu V_{\alpha,i}.$$

This error approximation incorporates the given data values, and as shown in Section 6, in practice it approximate the actual error better than the tight bound described in Section 3.3. In Figure 4, we demonstrate the high similarity of the approximated error function $\tilde{R}_\alpha$ to the true error function R. We used in this example the test functions introduced in Eq. (4)-(6).



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(m) (n) (o)

Figure 4: *Comparison between the true error R graphs (blue), and the approximated error $\tilde{R}$ graphs (green) when using quadratics to approximate the test functions using a uniformly distributed random point set. In each graph the x-axis stands for the support size h. In (a), the function F_1 has been used to create the data set, where the density of the points used was 0.25k points per unit square. In (b) the density is 1k points per unit square, and in (c) 25k points per unit square. In (d)-(f) function F_2 has been used. In (g)-(i) function F_3, and since all it's third derivatives vanish at the point of evaluation (origin) , the approximation is bad. Using third degree polynomial in (j)-(l) alleviates the problem. Also perturbing the evaluation point by 0.05 in the x coordinate (m)-(o) alleviates the problem.*

A drawback of this approach that it is not a bound, but merely an approximation of the error function, and it depends on the quality of the approximation of the coefficients f_ν. In the presence of very high derivatives and low sampling density it can perform worse than the data-independent bound. Another drawback appears at points where all the derivatives of order $N+1$ vanish. Then, the approximation of the error function may be damaged, see Figure 4. If we have a prior knowledge about such a point, the problem can be avoided by perturbing the interest point a little, or using higher degree polynomial N' (assuming the $N'+1$ derivatives do not all vanish at that point), in Figure 4, where all the third derivatives of F_3 vanish at the origin (the point of interest in that example) the approximation of the error function is quite inaccurate, however using third degree polynomial or moving the point of interest a little alleviates the problem. The reason of this phenomenon lies in the fact that the sign of the coefficients $D^{\nu} f(\eta_i(x_i - x) + x)$ are likely to change in the vicinity of x, hence the error function is highly dependent on the values of η_i.

3.5. Noisy Data

In this section we extend our previous error bounds and approximations to optimally handle errors (noise) in the sampled data. We assume that errors ε_i, where $|\varepsilon_i| \le \varepsilon$ are introduced into the data, that is, $f(x_i) = f^*(x_i) + \varepsilon_i$, where f^* stands for the 'true' sampled function.

It then follows,as in Theorem 3.1, that

$$R^*(x) = R(x) + \alpha! \sum_{i=1}^{I} \varepsilon_i V_{\alpha,i},$$

where $R^*(x) = D^{\alpha} p(x) - D^{\alpha} f^*(x)$ and $R(x)$ is given in Eq. (7). Hence,

$$|R^*(x)| \le |R(x)| + \alpha! \varepsilon \sum_{i=1}^{I} |V_{\alpha,i}|. \qquad (13)$$

Note that this bound is again tight since no assumption can be made on the signs of ε_i nor their magnitude, except that $|\varepsilon_i| \le \varepsilon$.

In Figure 2, using the points inside the black circle (b), which are chosen by the heuristic method, in the MLS approximation leads to $\sum_{i=1}^{I} |V_{\alpha,i}| \approx 15$ for $\alpha = (0,0)$ and ≈ 99 for $\alpha = (1,0)$. Hence, we can suspect that the noise level in the data might be amplified by these factors when approximating the value or the partial derivative ∂_x at the red point, respectively. Indeed, the *true* error in the function evaluation is $\sim 9\varepsilon$ and the error in the derivative approximation is $\sim 106\varepsilon$. Minimizing the bound (13) imply choosing a bigger support size in this case, which results in $\sim 0.17\varepsilon$ and $\sim 3.3\varepsilon$ error in approximation of the value and derivatives respectively. In section 3.6 we discuss another aspect of the error amplification phenomenon.

Next, we integrate the sampling error term into the former error terms. First the data dependent approximation,

$$|R^*(x)| \preceq |\tilde{R}(x)| + \alpha! \varepsilon \sum_{i=1}^{I} |V_{\alpha,i}|,$$

where $\preceq$ stands for $\le$ up to a term of magnitude $O(h)$. Therefore, we denote our approximated error in the approximation:

$$\tilde{R}_{\alpha,\varepsilon}(x, X_h) = |\tilde{R}_{\alpha}(x, X_h)| + \alpha! \varepsilon \sum_{i=1}^{I} |V_{\alpha,i}|. \qquad (14)$$

For the data-independent bound:

$$|R^*(x)| \le C|B_{\alpha}(x, X_h)| + \alpha! \varepsilon \sum_{i=1}^{I} |V_{\alpha,i}|.$$

Since C is unknown, this bound is better presented if we consider relative error, i.e., $f(x_i) = f^*(x_i)(1 + \varepsilon_i)$. In this case by similar consideration as before, the tight error bound in the presence of noise in the data becomes:

$$|R^*(x)| \le C \left(|B_{\alpha}(x, X_h)| + \alpha! \varepsilon \sum_{i=1}^{I} |V_{\alpha,i}| \right), \qquad (15)$$

where C bounds the relevant derivatives and the function values. In practice we considered the term in the parentheses as the function to be minimized in the presence of noise in the data:

$$B_{\alpha,\varepsilon}(x, X_h) = |B_{\alpha}(x, X_h)| + \alpha! \varepsilon \sum_{i=1}^{I} |V_{\alpha,i}|. \qquad (16)$$

3.6. A Confidence Measure

By Corollary 3.1 and Eq. (15) we have that

$$|D^{\alpha} p(x) - D^{\alpha} f(x)| \le C B_{\alpha,\varepsilon}(x, X_h),$$

where C bounds certain derivatives of the unknown function. Therefore, if we assume that the unknown function is sufficiently smooth with bounded derivatives, $B_{\alpha,\varepsilon}(x, X_h)$ furnishes a tool which justifies an approximation result. It can be understood as a *confidence measure* of the ability of a given set of points X_h to approximate $D^{\alpha} f(x)$. As an example of this application, assume we want to approximate local curvatures on a mesh. A common way to do it is fitting a local polynomial at each vertex using it's 1 or 2-ring neighborhood, extracting it's derivatives and using some standard classical differential geometry formula. As an easy example, suppose we want to approximate ∂_{xx} at the vertices of a sphere mesh. We use a sphere since we know its derivatives are bounded and are the same everywhere on the sphere, w.r.t the local frame. We define the parameter domain to be the plane perpendicular to the weighted average of the adjacent face's normals. Figure 5 shows a coloring of the sphere mesh, using the two parts of the tight error bound factor (16): In (a), the bound of the error factor caused in the approximation, $|B_{\alpha}(x, X_h)|$, and in (b) the bound of the error factor related to the noise in the data: $\alpha! \sum_{i=1}^{I} |V_{\alpha,i}|$. Note that the latter means that if one of the 1-ring neighborhoods contains noise in the direction of the normal of the parameter plane, the errors in the approximation might be multiplied by this factor. In this case we see that even 'nice' 1-rings of the sphere might cause ~ 100 times bigger errors than the noise level of the data. In (c) it is shown that irregular triangulation may yield much higher errors.

4. Optimal neighborhoods

In this section we present an algorithm, which finds the optimal support size ,h_{opt}, which should be used in the ap-

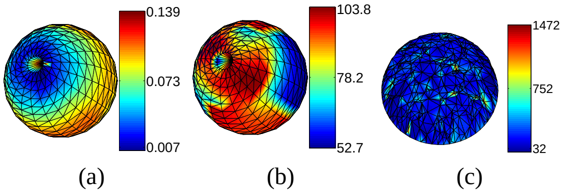


Figure 5: *Color maps of the confidence measures for approximating ∂_{xx} using quadratic polynomial fitted in the least-squares sense to the 1-ring neighborhoods on a sphere mesh. In (a) the (tight) error bound factor B_α caused in the approximation, for non-noisy data. In (b) the factors which multiply the noise ε in the data. Note that small noise level in the data may still cause ~ 100 times bigger errors in the approximation. (c) is the same as (b) for an irregular sphere mesh.*

proximation procedure (1)-(2) to ensure minimal error. We base our algorithm on the error analysis presented in former sections, i.e., equations (14) and (16). More specifically, for a given X, α, N and a point where the approximation is sought, x, we look for the optimal h, which we denote by h_{opt}, which minimizes the bound or the approximation of the error $|D^\alpha f(x) - D^\alpha p(x)|$.

4.1. Finding an Optimal Support Size

We use the same algorithm for both bounds (14) and (16). We look for the support size h which minimizes the bound function $EB_\alpha(x, X, h)$, where for brevity we will use the symbol EB_α for both bounds.

We fix an upper and lower bounds H_{min}, H_{max} for h ,e.g., H_{min} could be set to h_J (which is defined in Section 3.4) and H_{max} to some large support size radius, we used for example $4h_J$. A rough step size Δ is set, e.g., we used $|H_{max} - H_{min}|/50$, and the algorithm traverse $h = H_{min}, H_{min} + \Delta, \ldots, H_{max}$ where for each h the algorithm calculates the error bound function EB_α, for the data points X_h. Next, after extracting the minimizing support size radius $h^{(0)}_{opt} = argmin_{h=H_{min}+j\Delta}\{EB_\alpha(x, X, h)\}$, we further improve the approximation to h_{opt} by fitting an interpolating quadratic near $h^{(0)}_{opt}$ and minimize it to define $h^{(1)}_{opt}$. We iterate this procedure until $|h^{(k+1)}_{opt} - h^{(k)}_{opt}| \le tolerance$. In our application we actually minimized EB^2, for faster convergence.

For efficient computation the following considerations are employed. First we move the origin to x, i.e., we use the points $x_i := x_i - x$,$i = 1, .., I$. Second, we rearrange $x_i \in X \cap B_{H_{max}}(0)$ with respect to their distance from 0 (x), where now the sub-index i is with respect to this ordering. If E is the Vandermonde matrix based on the data points $x_1, \ldots, x_I$, then the matrix E' for the data points $x_1, \ldots, x_{I+1}$ can be written as $E'_{i,j} = E_{i,j}$ for $i \le I$ and $E_{I+1,j} = p_j(x_{I+1})$. This implies that when h changes to include a new point x_{I+1} in X_h, we only need to add a single row to the previous Vandermonde matrix E, to construct the Vandermonde matrix for X_h.

Note that if we have calculated the bound for the data points $x_1, \ldots, x_I$, then the quantity $\sum_{|\beta|=N+1} |x_i|^\beta$ for $i = 1, .., I$ should be re-used and the only new calculation that should be performed is $\sum_{|\beta|=N+1} |x_{I+1}|^\beta$. Taking into consideration all the above remarks, when h is changed to include a new point, the most time consuming part of the calculation consists of factorization of $J \times J$ matrix (for example for quadratic interpolation we have $J = 6$), and back-substitution for I vectors (the matrix $E^t W$), this leads to an $O(J^3 + J^2 I)$ complexity for each step of the algorithm. This is multiplied by the number of iterations of the algorithm, which in our implementation is ≤ 100. In the case of using the data dependent bound, there is a preprocess step of solving for the coefficients f_ν as explained in Section 3.4. The computational cost of this step is $O(J^4 + J^3 I)$.

An important consequence of the above procedure is that since V is calculated anyway, we actually get for no significant extra calculations the bound of the error for any α, $|\alpha| \le N$, that is, the bound for every possible derivative (and value) approximation.

It should be noted that to get consistent h values, we take the first global minimum (if there is more than one zero). Another delicate point, is that the parameter value h_{min} of the minimum of $EB_\alpha(x, X, h)$, i.e., $h_{min}(x) = argmin_h\{EB_\alpha(x, X, h)\}$, is a piecewise smooth function of x if the neighborhoods used in the approximation of f_ν are smoothly chosen. This implies that the MLS approximation, based on this h field, is only piecewise smooth.

4.1.1. Integrating with the MLS projection operator

All the previous construction dealt with a function over a parameter space. When dealing with a point cloud there is no natural choice of such space. A popular method for choosing this space in the case of surfaces is the MLS projection operator [Lev03, ABCO*01, PKKG03, AK04]. After choosing the parameter space, in this case a plane, we are back to our original functional settings, with a minor difference: the distance to the neighboring points is measured using their actual position in space and not their projection on the parameter space, i.e., p is defined by minimizing

$$\sum_{i=1}^{I} (f(x_i) - p(x_i))^2 \eta_h(\|(x_i, f(x_i)) - (x, z)\|),$$

where (x, z) is chosen by the first step of the MLS projection. This small change can be easily incorporated in our system, one just have to redefine the way distances are measured. We have integrated that into our system and noticed two interesting results: For the test function F_1, F_2 the results were similar to the algorithm which measured the distance on the parameter space (results are demonstrated in Section 6). However, In the case of data taken from F_3, since the function is rapidly oscillating, the new distance measure is likely to prefer points from other periods and not from close parameter values, and the approximation quality deteriorates.

density	L^x	f	N	ε	$\overline{H}$	$\sigma(H)$
.25k	$f(x)$	F_1	2	0	2.24	0.74
.25k	$f(x)$	F_2	2	0	2.15	0.73
1k	$f(x)$	F_2	2	0	2.1	0.69
1k	$f(x)$	F_1	2	0	2.3	0.75
1k	$f(x)$	F_1	3	0	1.88	0.57
1k	$\partial_x f(x)$	F_1	3	0	2.28	0.74
4k	$\partial_{xy} f(x)$	F_2	3	0	2.06	0.74
.5k	$f(x)$	F_3	2	10^{-2}	2.41	0.77
.5k	$f(x)$	F_3	2	10^{-4}	2.35	0.77
.5k	$\partial_x f(x)$	F_2	2	10^{-5}	2.25	0.8

Table 1: *Experiments results used to derive the heuristic support size rule.*

5. Heuristics

In this section we present the derivation of the heuristic method for choosing a support size h, which we later use for comparison with the optimal choice. The method is derived by experiments, in the following way: We consider the *true* error of our MLS approximation procedure as a function of the support size used h. We let h vary from it's minimal value, i.e., the radius h_J of the ball containing the J nearest neighbors up to 4 times this radius. It is observed that the minimum can be predicted as a certain constant times h_J. The heuristic is based on finding the right constant, and we do it by extensive simulation. We define the random variable

$$H = \frac{h_{best}}{h_J},$$

where h_{best} stands for the true optimal support size, i.e., the support size which minimizes the true error function in the interval $[h_J, 4 \times h_J]$. In table 1, we specify several measurements of H, in particular we consider different test functions (4)-(6), point densities, noisy and non-noisy data, quadratic and cubic polynomial and different functionals. The density specify the number of data points per unit square. H was sampled in a grid inside the domain $[-1,1] \times [-1,1]$. The mean $\overline{H}$ and standard deviation $\sigma(H)$ are computed. In general the results of the different scenarios are similar: the $\overline{H}$ is approximately 2.2 and $\sigma(H)$ is generally around 0.75. Therefore, our heuristic choice of support size to be used in the MLS approximation is $h = 2.2h_J$.

A similar heuristic for choosing the support size h may be obtained by using the error bound B_α, as follows: For a given distribution of data points near the origin we find the optimal h minimizing B_α, and compute h_{opt}/h_J. Averaging these ratios over many randomly chosen distributions of data points, we obtain for example, an average ratio ~ 1.9 for quadratic polynomial approximation of the function value, $N = 2$, $\alpha = (0,0)$, and ratio ~ 2.4 for $N = 2$, $\alpha(1,0)$ with noise level of 10^{-5}. Another option is to compute a different rule for each noise level ε, but we didn't pursue this direction.

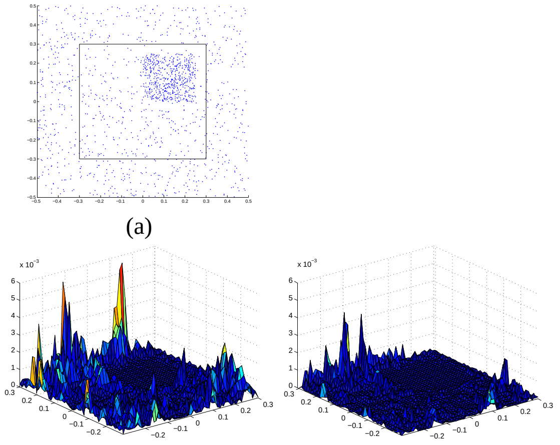


Figure 6: *An example of data set with irregular density. We used quadratic polynomials in the MLS approximation, and resampled in the drawn rectangle (a). In (b) the error resulted using the heuristic support size h. In (c) the optimal h has been used. The error ratio* E_1 *is* 0.37.

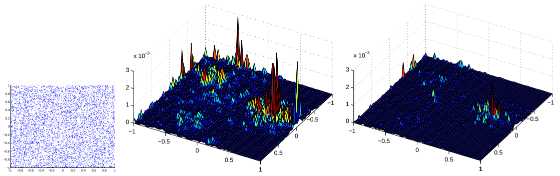


Figure 8: *An example of uniformly distributed data set sampled from* F_1. *We used cubic MLS approximation, and resampled in the drawn region (a). In (b) we display the error resulting using the heuristic support size. In (c) the error using the optimal support size. Note the errors at the boundaries. The error ratio* E_1 *is* ≈ 0.22.

6. Numerical experiments

In this section we present numerical experiments performed with the algorithms described in the Section 4. We compare the algorithm for choosing h by minimizing the error bound (16), or the approximation of the error function (14), to the heuristic approach described in Section 5. In general the method based on the data dependent approximation works best, and the method based on the tight bound works slightly better than heuristic method.

We have tested our algorithm in the following three main scenarios: 1. Uniform distributed points. 2. Data with noise. 3. Irregular distributed points (change in density). In our experiments we use quotient of the 1-norms as a measure of error: $E_1 := \|E_{opt}\|_1 / \|E_{heu}\|_1$, and $E_1' := \|E_{bnd}\|_1 / \|E_{heu}\|_1$. We denote by E_{opt} the error of the MLS algorithm using the support size h determined by data dependent error approximation $\tilde{R}$, by E_{bnd} we denote the error of the algorithm when

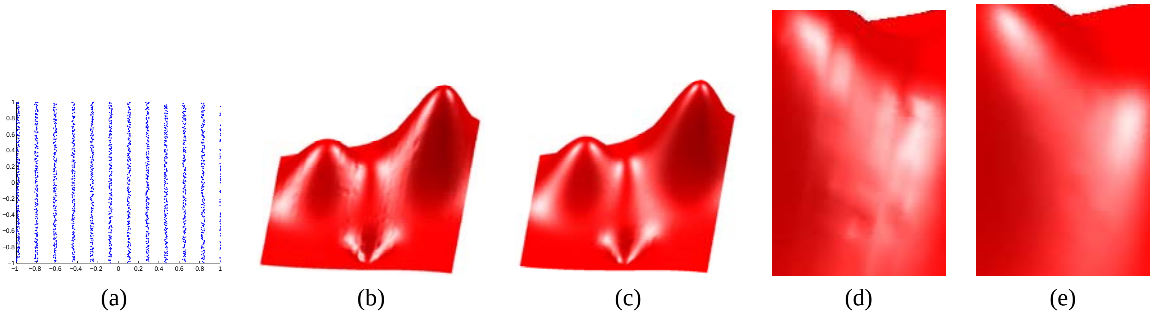


Figure 7: *An example of data set sampled from F_1 with irregular density (a). We used quadratic MLS approximation, and resampled in the drawn region (a). In (b) (zoom-in in (d)) the reconstructed surface using the heuristic support size. In (c) (zoom-in in (e)) optimal h has been used. The resulting error ratio E_1 is ≈ 0.34.*

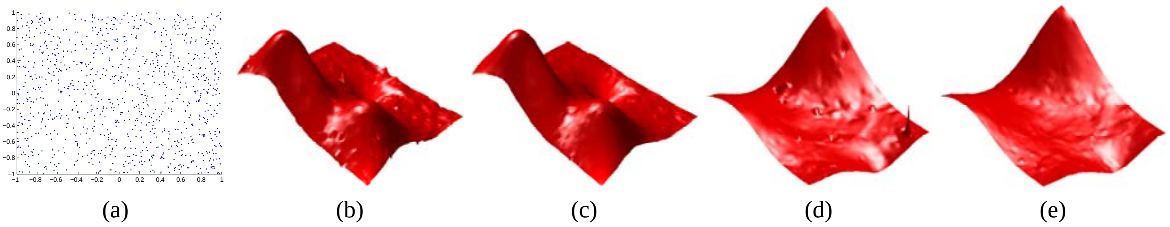


Figure 9: *A very noisy point cloud data (a), for quadratic MLS surface reconstruction. The point cloud was sampled with white noise $\varepsilon = 10^{-1}$ from F_1. In (b) the reconstructed surface (zoom-in in (d)) using the heuristic method, and in (c) (zoom-in in (e)) using the data dependent method.*

using the data-independent bound B_α, and by E_{heu} we denote the error when applying the heuristic approach of choosing the h.

f	density	E_1	E_1'
F_1	0.25k	0.46	0.86
F_1	1k	0.32	0.87
F_1	25k	0.16	0.82
F_2	0.25k	0.53	0.84
F_2	1k	0.38	0.86
F_2	25k	0.19	0.87
F_3	0.25k	0.74	0.82
F_3	1k	0.52	0.89
F_3	25k	0.26	0.89

Table 2: *Experiments with uniform distributed data and no noise.*

Table 2 shows a comparison of the two error analysis based methods to the heuristics, in the case of no noise and uniform distributed points. The approximated functional is point evaluation, i.e., $L^x(f) = f(x)$, and the degree of the polynomial space is $N = 2$. In each experiment a new uniformly distributed data points where taken and the 1000 query points where randomized. The density specify, as before, the number of data points per unit square. Note that method based on the data dependent approximation performs the best, and improves as the density increases. The method based on the data independent bound is working slightly better than the heuristic. In Figure 8, we show the error graphs when resampling using cubic ($N = 3$) MLS for uniformly distributed data (a), by the heuristic method (b), or by the data dependent method (c). Note that the errors near the boundaries are lower with the latter method.

Table 3 exhibits a similar comparison between the methods for noisy data, approximating various derivative functionals. Figure 9, shows a resampling of F_1 contaminated with high noise level ($\varepsilon = 0.1$) (a), by the heuristic method (b), zoom-in in (d). Similarly (c),(e) exhibits the above cases when using the method based on the data dependent approximation.

Figures 6, 7, exhibit experiments using data set with irregular density. Figure 6 examine the resampling algorithm in region (a) in the vicinity of a high density region. (c) shows the error graph of the heuristic method. (d) is the error graph of the method based on the data dependent approximation. Similarly, Figure 7, shows another configuration of irregular point density (a), which may present itself at scanned point clouds. The reconstructed surfaces are drawn in (b) and (c) and zoom-in at (d),(e).

In Figures 1, 10, we show reconstruction results with scanned data. We compare the heuristic approach to the data-dependent error approximation approach. In Figure 1 we added noise to the data, while in Figure 10 we used the original raw data, and assume error at maximal size $\varepsilon = 10^{-5}$.

f	density	N	L^x	ε	E_1	E_1'
F_1	1k	3	∂_{xy}	10^{-2}	0.32	0.31
F_2	11k	3	∂_x	10^{-4}	0.62	0.71
F_1	1k	3	∂_y	10^{-5}	0.61	2.53
F_2	1k	2	$f(x)$	10^{-1}	0.63	0.6

Table 3: *Experiments with noise.*

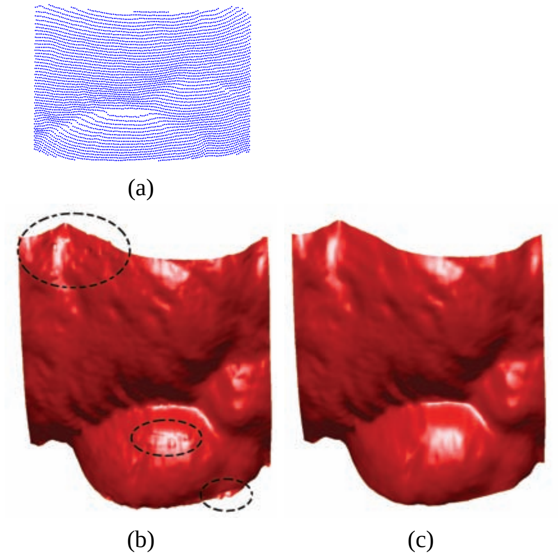


Figure 10: *Resampling part of a raw point cloud of the Bunny taken from the Stanford 3D Scanning Repository (a). The result using a quadratic MLS approximation with the heuristic approach to determine the local support size is shown in (b). The artifacts appearing in (b) are removed by using the approach based on the data dependent error approximation, as shown in (c) (using $\varepsilon = 10^{-5}$).*

7. Conclusions

In this paper we consider the problem of evaluating the approximation quality of the MLS method, and we derive an algorithm which finds the best support size to be used in the approximation. Two methods based on a novel error formula in the MLS approximation were considered: One, based on a conservative tight bound, and second, based on a data dependent approximation to the error function in the MLS approximation. In the process, we have carefully chosen a heuristic, based on the observation that the ratio of the optimal support size h_{best} and the radius of the ball containing the J nearest neighbors h_J, can be fairly well predicted by the constant ~ 2.2.

Comparing our error analysis based methods to the heuristic shows that the method based on the tight bound performs slightly better than the heuristic in presence of very small and very large noise levels. The method based on the data dependent approximation works generally better than the other methods, and achieves the best approximation and stability properties.

References

[ABCO*01] Alexa M., Behr J., Cohen-Or D., Fleishman S., Levin D., Silva C. T.: Point set surfaces. In *VIS '01: Proceedings of the conference on Visualization '01* (Washington, DC, USA, 2001), IEEE Computer Society, pp. 21–28.

[AK04] Amenta N., Kil Y. J.: The domain of a point set surfaces. *Eurographics Symposium on Point-based Graphics 1*, 1 (June 2004), 139–147.

[Bos91] Bos L.: On certain configurations of points in $\mathbb{R}^n$ which are unisolvent for polynomial interpolation. *Journal of Approximation Theory 64*, 3 (1991), 271–280.

[CP05] Cazals F., Pouget M.: Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design 22*, 2 (2005), 121–146.

[FR01] Floater M. S., Reimers M.: Meshless parameterization and surface reconstruction. *Comput. Aided Geom. Des. 18*, 2 (2001), 77–92.

[Fra82] Franke R.: Scattered data interpolation: tests of some methods. *Math Comp 38* (1982), 181–200.

[GS00] Gasca M., Sauer T.: Polynomial interpolation in several variables. *Adv. Comput. Math. 12*, 4 (2000), 377–410.

[HDD*92] Hoppe H., DeRose T., Duchamp T., McDonald J., Stuetzle W.: Surface reconstruction from unorganized points. *Computer Graphics 26*, 2 (1992), 71–78.

[Lev98] Levin D.: The approximation power of moving least-squares. *Mathematics of Computation 67*, 224 (1998), 1517–1531.

[Lev03] Levin D.: Mesh-independent surface interpolation. *Geometric Modeling for Scientific Visualization* (2003).

[OBA*03] Ohtake Y., Belyaev A., Alexa M., Turk G., Seidel H.-P.: Multi-level partition of unity implicits. *ACM Trans. Graph. 22*, 3 (2003), 463–470.

[PGK02] Pauly M., Gross M., Kobbelt L. P.: Efficient simplification of point-sampled surfaces. In *VIS '02: Proceedings of the conference on Visualization '02* (Washington, DC, USA, 2002), IEEE Computer Society, pp. 163–170.

[PKKG03] Pauly M., Keiser R., Kobbelt L. P., Gross M.: Shape modeling with point-sampled geometry. *ACM Trans. Graph. 22*, 3 (2003), 641–650.

[SX95] Sauer T., Xu Y.: On multivariate lagrange interpolation. *Math. Comput. 64*, 211 (1995), 1147–1170.

[Wen01] Wendland H.: Local polynomial reproduction and moving least squares approximation. *IMA Journal of Numerical Analysis 21* (2001), 285–300.