Algorithm XXX: SHEPPACK: Modified Shepard Algorithm for Interpolation of Scattered Multivariate Data

WILLIAM I. THACKER
Winthrop University
JINGWEI ZHANG, LAYNE T. WATSON, JEFFREY B. BIRCH, MANJULA A. IYER
Virginia Polytechnic Institute and State University
and
MICHAEL W. BERRY
University of Tennessee

Scattered data interpolation problems arise in many applications. Shepard's method for constructing a global interpolant by blending local interpolants using local-support weight functions usually creates reasonable approximations. SHEPPACK is a Fortran 95 package containing five versions of the modified Shepard algorithm: quadratic (Fortran 95 translations of Algorithms 660, 661, and 798), cubic (Fortran 95 translation of Algorithm 791), and linear variations of the original Shepard algorithm. An option to the linear Shepard code is a statistically robust fit, intended to be used when the data is known to contain outliers. SHEPPACK also includes a hybrid robust piecewise linear estimation algorithm RIPPLE (residual initiated polynomial-time piecewise linear estimation) intended for data from piecewise linear functions in arbitrary dimension m. The main goal of SHEPPACK is to provide users with a single consistent package containing most existing polynomial variations of Shepard's algorithm. The algorithms target data of different dimensions. The linear Shepard algorithm, robust linear Shepard algorithm, and RIPPLE are the only algorithms in the package that are applicable to arbitrary dimensional data.

Categories and Subject Descriptors: G.1.1 [**Numerical Analysis**]: Interpolation — *Spline and piecewise polynomial interpolation*; G.1.2 [**Numerical Analysis**]: Approximation — *Least squares approximation; Linear approximation*; G.3 [**Mathematics of Computing**]: Probability

This work was supported in part by National Science Foundation grants DMI-0422719 and DMI-0355391, and Department of Energy grant DE-FG02-06ER25720.
Authors' addresses: W. I. Thacker, Computer Science Department, Winthrop University, Rock Hill, SC 29733; e-mail: thackerw@winthrop.edu; J. Zhang, L. T. Watson, M. A. Iyer, Departments of Computer Science and Mathematics, Virginia Polytechnic Institute & State University, Blacksburg, VA 24061; e-mail: {jwzhang, ltw, manjula}@cs.vt.edu; J. B. Birch, Department of Statistics, Virginia Polytechnic Institute & State University, Blacksburg, VA 24061; e-mail: jbbirch@vt.edu; M. W. Berry, Department of Computer Science, University of Tennessee, Knoxville, TN 379963450; e-mail: berry@cs.utk.edu.

and Statistics — *Robust regression*; G.4 [**Mathematics of Computing**]: Mathematical Software — *Algorithm design and analysis*

General Terms: Algorithms, Design, Documentation

Additional Key Words and Phrases: Shepard's algorithm, M-estimation, RIPPLE

1. INTRODUCTION

Interpolation problems arise in many areas where there is a need to construct a continuous surface from irregularly spaced data points. These areas include cartography, geophysics, data mining, engineering, meteorology, landscape ecology, computer graphics, and scientific visualization. The problem is to find a surface that approximates a function defined in m-dimensional Euclidean space E^m, from a finite set of data points.

Currently, there are a number of solutions to the scattered data interpolation problem. The choice of the interpolation technique depends on the distribution of points in the data set, application domain, approximating function, or the method that is prevalent in the discipline.

Shepard's interpolation method, based on a weighted average of values at the data points, usually creates good approximations. There are several variations of the original Shepard algorithm based on quadratic, cubic, linear, and trigonometric polynomials. Quadratic and cubic Shepard's method variations require more coefficients, and hence more data, than the linear Shepard method requires. Thus, in higher dimensions, it is more practical to use the linear Shepard method. Even though the performance of the linear Shepard method is comparable to other Shepard's techniques, there are situations where statistically robust least squares fits are necessary. Applications of such robust local approximations are well known in image processing.

SHEPPACK contains five different Fortran 95 implementations of the modified Shepard algorithm. QSHEP2D, QSHEP3D, and QSHEPMD are quadratic Shepard methods. QSHEP2D [Renka, 1988a] and QSHEP3D [Renka, 1988b] are direct translations of the original ACM algorithms, written in FORTRAN 77, developed by Renka for two-dimensional and three-dimensional data interpolation, respectively. QSHEP5D (developed by Berry [1999] in C++ as an upgrade to QSHEP3D for five-dimensional interpolation) has been translated to Fortran 95 in QSHEPMD. CSHEP2D, a cubic Shepard method, is also a direct translation of the original ACM algorithm, written in FORTRAN 77, developed by Renka [1999a]. However, since the original codes were written using single precision, the tolerance for detecting an ill-conditioned system was changed from the arbitrary value of 0.01 to the square root of machine epsilon for the current processor. This increases the portability of the codes. LSHEP is a linear Shepard interpolation method for arbitrary dimensional data. Note that the new code LSHEP is the only one of the five that is applicable to data of dimension $m > 5$. The code LSHEP includes, as an option, a

statistically robust algorithm. SHEPPACK also includes a new hybrid robust piecewise linear estimation algorithm called RIPPLE (residual initiated polynomial-time piecewise linear estimation) intended for data from piecewise linear functions in arbitrary dimension m.

The remaining sections of this paper are organized as follows. Section 2 mentions some common interpolation techniques. Section 3 presents Shepard's original algorithm and its modified polynomial variations. Section 4 presents a statistically robust linear Shepard algorithm and the motivation for a hybrid robust estimation algorithm. Section 5 presents the new hybrid robust algorithm RIPPLE for piecewise linear data. Section 6 describes the organization and usage of SHEPPACK, and Section 7 presents performance results for all the algorithms contained in SHEPPACK.

2. INTERPOLATION TECHNIQUES

Shepard [1968] proposed an interpolation method that created a surface based on a weighted average of values at data points. The weight function was an inverse distance function of the data points. All methods of this type can be viewed as generalizations of Shepard's method. It was later found that this form of a weight function accorded too much influence to data points that were far away from the point of approximation. Franke and Neilson [1980] developed a modification in which the weight function was designed to have local support and localized the overall approximation. This method is called the local modified Shepard method. Several variations of the modified Shepard algorithm have been developed.

Coons [1967] proposed a method for describing free form curved surfaces of a very general kind. Following this work, NURBS (nonuniform rational B-splines) were proposed by Versprille [1975]. They are a generalization of tensor product B-splines, and have several useful properties that have contributed to their popularity and wide commercial use. As in the case of ordinary B-splines, the sum of rational basis functions of NURBS is equal to unity. By varying the knots, NURBS can satisfy a variety of smoothness requirements. Using NURBS, it is possible to represent free form curves and surfaces as well as analytic surfaces (such as conics and quadrics). A large variety of shapes can be designed by changing the coordinates of the control points and the weights associated with them. See Piegl [1989a, 1989b] for a more detailed discussion of the properties of NURBS.

A DACE (design and analysis of computer experiments) model is an interpolating model based on Bayesian statistics. It uses the prior distribution mechanism in Bayesian statistics through which one applies past experiences, knowledge, and intuition when solving a problem. In DACE models, the unknown function is typically expressed as the sum of a known function and a Gaussian random function. The known function is usually a constant, which is estimated based on observed response values. The Gaussian random function is characterized by a covariance matrix that depends on a correlation function selected by the user. See Guinta [1997] for a good overview of DACE. A more detailed discussion of the fundamental

statistical and mathematical concepts can be found in Sacks et al. [1989], Koehler and Owen [1996], Osio and Amon [1996], and Booker et al. [1995].

The MARS (multivariate adaptive regression splines) technique adaptively selects a set of spline basis functions for approximating the response function through a forward/backward iterative approach. The algorithm partitions the input space into regions, each with its own regression equation. It then constructs a relation between the predictor variables and dependent variables (the spline approximation) from a set of basis functions that are entirely based on the regression data. In general, the technique is popular because it does not assume any particular type of relationship between predictor and dependent variables, and thus is widely applicable. See Friedman [1991] for a complete description of MARS.

Another popular approximation method uses radial basis functions (RBFs). The significance and usefulness of approximation by RBFs follows from the theory developed in Schoenberg [1938]. The paper by Micchelli [1986] presents fundamental theory for approximation by RBFs. This type of approximation works well with scattered data because the interpolant only depends on the distances from the interpolation points. A popular choice for an interpolant function is a polyharmonic spline, which is derived from a strictly positive definite function. More details about the theory behind the choice of the interpolant function can be found in Sibson and Stone [1991], who obtained results using the thin-plate spline radial basic functions together with linear polynomials.

For the special case of piecewise linear approximation in one dimension, recent work includes the algorithm L2WPMA of Demetriou [2007]. The code L2WPMA (least squares weighted piecewise linear approximation, Algorithm 863) calculates a piecewise monotonic approximation to n univariate data points contaminated by random errors. The continuous piecewise linear interpolant consists of k (a positive integer provided by the user) monotonic linear splines, alternately monotonically increasing and monotonically decreasing.

An excellent source for modern approximation theory is Cheney and Light [1999], which describes many different approximation methods.

3. SHEPARD ALGORITHMS

This section describes the original Shepard algorithm, and also presents the quadratic, cubic, and linear variations of the modified Shepard algorithm.

3.1 Original Shepard Algorithm

Local methods are attractive for very large data sets because the interpolation or approximation at any point can be achieved by considering only a local subset of the data. Many local methods can be characterized as weighted sums of local approximations $P_k(x)$, where the weights $W_k(x)$ form a partition of unity. In order for the overall method to be local, it is necessary that the weight functions have local support, that is, be nonzero over a bounded region, or at a limited number of the data points.

The original global inverse distance weighted interpolation method is due to Shepard [1968]. All methods of this type may be viewed as generalizations of Shepard's method.

Let E^m denote m-dimensional Euclidean space, $x = (x_1, \ldots, x_m) \in E^m$, and for real w let $w_+ = \max\{0, w\}$. The scattered data interpolation problem can be defined as: given a set of irregularly distributed points $x^{(i)} \in E^m$, $i = 1, \ldots, n$, and scalar values f_i associated with each point satisfying $f_i = f\big(x^{(i)}\big)$ for some underlying function $f : E^m \to E$, look for an interpolating function $\tilde{f} \approx f$ such that $\tilde{f}\big(x^{(i)}\big) = f_i$. Assume that every m-simplex formed from the points $x^{(i)}$ is nondegenerate (has a nonempty interior).

Define an approximation to $f(x)$ by

$$\tilde{f}(x) = \frac{\sum_{k=1}^{n} W_k(x) f_k}{\sum_{k=1}^{n} W_k(x)},$$

where the weight functions $W_k(x)$ are defined in the original Shepard algorithm as

$$W_k(x) = \frac{1}{\big\|x - x^{(k)}\big\|_2^2}.$$

However, this form of the weight functions accords too much influence to data points that are far away from the point of approximation and may be unacceptable in some cases.

3.2 Modified Shepard Algorithm

Franke and Nielson [1980] developed a modification that eliminates the deficiencies of the original Shepard's method. They modified the weight function $W_k(x)$ to have local support and hence to localize the overall approximation, and replaced f_k with a suitable local approximation $P_k(x)$. This method is called the *local modified Shepard method* and has the general form

$$\tilde{f}(x) = \frac{\sum_{k=1}^{n} W_k(x) P_k(x)}{\sum_{k=1}^{n} W_k(x)},$$

where $P_k(x)$ is a local approximant to the function $f(x)$ centered at $x^{(k)}$, with the property that $P_k\big(x^{(k)}\big) = f_k$. The choice for the weight functions $W_k(x)$ used by Renka [1988a] was suggested by Franke and Nielson [1980] and is of the form

$$W_k(x) = \left[\frac{\Big(R_w^{(k)} - d_k(x)\Big)_+}{R_w^{(k)} d_k(x)} \right]^2,$$

where $d_k(x) = \left\|x - x^{(k)}\right\|_2$ is the Euclidean distance between the points x and $x^{(k)}$, and the constant $R_w^{(k)} > 0$ is a radius of influence about the point $x^{(k)}$ chosen just large enough to include N_w points. The data around $x^{(k)}$ only influences $\tilde{f}(x)$ values within this radius.

There are several variations of the original Shepard algorithm based on polynomial and trigonometric functions for P_k.

The polynomial function P_k is written as a Taylor series about the point $x^{(k)}$ with constant term $f_k = P_k\big(x^{(k)}\big)$ and coefficients chosen to minimize the weighted sum of squares error

$$\sum_{\substack{i=1 \\ i \neq k}}^{n} \omega_{ik} \left[P_k\big(x^{(i)}\big) - f_i\right]^2,$$

with weights

$$\omega_{ik} = \left[\frac{\left(R_p^{(k)} - d_i\big(x^{(k)}\big)\right)_+}{R_p^{(k)} d_i\big(x^{(k)}\big)}\right]^2,$$

and $R_p^{(k)} > 0$ defining a radius about $x^{(k)}$ within which data is used for the least squares fit. R_w and R_p are taken by Franke and Nielson [1980] as

$$R_w = \frac{D}{2}\sqrt{\frac{N_w}{n}}, \quad R_p = \frac{D}{2}\sqrt{\frac{N_p}{n}},$$

where $D = \max\limits_{i,j} \left\|x^{(i)} - x^{(j)}\right\|_2$ is the maximum distance between any two data points, and N_w and N_p are arbitrary positive integers. The constant values for R_w and R_p are appropriate assuming uniform data density. The recommended values for N_w and N_p for the various Shepard algorithms are given below.

3.3 Quadratic Shepard Algorithm for Two-dimensional Data

QSHEP2D (quadratic Shepard algorithm for two-dimensional data) is Algorithm 660 developed by Renka [1988b]. The subroutine QSHEP2 in SHEPPACK is a direct Fortran 95 translation of Renka's FORTRAN 77 code of the same name.

The parameters N_p and N_w for QSHEP2 were chosen by computing the error using p-norms $\|\cdot\|_p$, $p = 1, \ldots, 24$ for all parameter values $N_p \in \{9, 10, \ldots, 16\}$ and $N_w \in \{1, 2, \ldots, 30\}$ over an assortment of test functions and data sets. The optimal pair of parameter values was found to vary widely with the choice of norm, test function, and data set, but in each case, the error norms vary smoothly with N_p and N_w, generally increasing monotonically and slowly with distance from the optimal pair. The recommended values are $N_p = 13$ and $N_w = 19$.

3.4 Quadratic Shepard Algorithm for Three-dimensional Data

QSHEP3D (quadratic Shepard algorithm for three-dimensional data) is Algorithm 661 developed by Renka [1988c]. The subroutine QSHEP3 in SHEPPACK is a direct Fortran 95 translation of Renka's Fortran 77 code of the same name.

The parameters N_p and N_w for QSHEP3D were chosen by computing the error using p-norms $\|\cdot\|_p$, $p = 1, \ldots, 12$ for all parameter values $N_p \in \{11, 12, \ldots, 18\}$ and $N_w \in \{6, 7, \ldots, 35\}$ over an assortment of test functions and data sets. The choice of optimal values is less clear than in the two-dimensional case due to more rapid variation of some of the error norms with respect to N_p values. The optimal range of parameters was found to be $N_p \in \{13, 14\}$ and $N_w \in \{31, \ldots, 35\}$.

3.5 Quadratic Shepard Algorithm for Higher Dimensional Data

Algorithm 798 developed by Berry [1999], is a quadratic Shepard algorithm for higher dimensional data (up to five dimensions). The Fortran 95 subroutine QSHEPM provided in SHEPPACK is a direct translation of Berry's C++ code QSHEPM. The visualization of results using netcdf file format has not been incorporated in the translation.

Following the recommendation of Renka [1988a], QSHEPM accepts parameters N_p and N_w as follows. For an m-dimensional hypervolume with N known nodes, N_p and N_w must satisfy

$$\frac{m(m+3)}{2} \le N_p \le \min\{50, N-1\}, \qquad 1 \le N_w \le \min\{50, N-1\}.$$

3.6 Cubic Shepard Algorithm for Two-dimensional Data

CSHEP2D (cubic Shepard algorithm for two-dimensional data) is Algorithm 790 developed by Renka [1999a]. The subroutine CSHEP2 in SHEPPACK is a direct Fortran 95 translation of Renka's FORTRAN 77 code of the same name.

The parameters N_p and N_w were chosen using Renka's recommendation [1988a]. The optimal values were found to be $N_p = 17$ and $N_w = 30$.

3.7 Linear Shepard Algorithm for Arbitrary Dimensional Data

The basis function $P_k(x)$ was the constant f_k in the original Shepard algorithm. Later variants used a quadratic polynomial, a cubic polynomial, and a cosine trigonometric polynomial (Renka [1999b]) as basis functions. The primary disadvantage for large data sets is that a considerable amount of preprocessing is needed to determine the closest points and calculate the local approximation. The second order polynomial models have $(m+2)(m+1)/2$ coefficients for m design variables, therefore the number of data points required to estimate $P_k(x)$ is at least $(m+2)(m+1)/2$, which is prohibitive for a problem in which $m \gg 5$ and function values f_k are expensive. If $P_k(x)$ has degree d, the number of coefficients is $\binom{m+d}{d}$, requiring at least that many data points.

This consideration motivates the choice of $P_k(x)$ as linear, which would require only $(m+1)$ function values to be computed in order to construct the local least squares fit. Define $N_p = \min\{n, \lceil 3m/2 \rceil + 1\}$ as the number of points used in the local least squares approximation and

$$R^{(k)} = \min\left\{ r \mid \overline{B(x^{(k)}, r)} \text{ contains at least } N_p \text{ of the points } x^{(i)} \right\},$$

where $\overline{B(x,r)}$ is the closed ball of radius r with center x. Then, the radii R_p and R_w vary with k and are taken to be $R_p^{(k)} = 1.1R^{(k)}$, $R_w^{(k)} = \min\{D/2, R^{(k)}\}$. The choice of N_p derives from the statistics rule of thumb that $3m/2$ data points are required to reasonably estimate m parameters.

The linear Shepard method would choose $P_k(x)$ as

$$P_k(x) = f_k + \sum_{j=1}^{m} a_j^{(k)} \left(x_j - x_j^{(k)} \right).$$

Let $S = \left\{ i_1, i_2, \ldots, i_{N_p - 1} \right\}$ be the set of indices corresponding to the $N_p - 1$ points closest to $x^{(k)}$ (or those with the smaller indices in case of a tie) that determine the local least squares approximation $P_k(x)$. Their weights satisfying $\omega_{i_j k} > 0$ since $R_p^{(k)}$ is slightly larger than $R^{(k)}$.

Define the $(N_p - 1) \times m$ matrix A and $(N_p - 1)$-vector b by

$$A_{j\cdot} = \sqrt{\omega_{i_j k}} \left(x^{(i_j)} - x^{(k)} \right)^t,$$

$$b_j = \sqrt{\omega_{i_j k}} \left(f_{i_j} - f_k \right).$$

The coefficients $a^{(k)}$ of $P_k(x)$ are then the minimum norm solution of the linear least squares problem

$$\min_{a \in E^m} \|Aa - b\|_2,$$

found by the SVD factorization of A via the LAPACK subroutine DGELSS.

In the unlikely event that for some k the matrix A corresponding to the index set S is ill-conditioned, the computation proceeds using the minimum norm solution, which is possible since the least squares problem is solved via SVD factorization. In this event, an error flag is set indicating that at least one ill-conditioned least squares problem was encountered.

The choice of the blending influence radius $R_w^{(k)}$ is determined by the diameter D of the point set and the number of points used for a local linear least squares approximation. Depending on the relative location of $x^{(k)}$ within the point set and the shape of the point set, either of the two expressions in the min may determine the minimum.

The subroutine LSHEP in SHEPPACK implements the linear Shepard algorithm as described above, for arbitrary dimension m. The function LSHEPVAL returns the approximate function value $\tilde{f}(x)$. If the weight function $W_k(x)$ is zero for all local approximations, LSHEPVAL returns the approximate function value defined in the original Shepard algorithm using only the $m + 1$ closest points to x (and again, if the smallest ball $\overline{B(x,r)}$ containing the closest $m + 1$ points also contains more than $m+1$ points $x^{(i)}$, then only the $m+1$ points $x^{(i)}$ with the lowest indices i are used).

4. ROBUSTNESS IN LINEAR SHEPARD ALGORITHM

As described in the previous section, in the linear Shepard method, the coefficients of the linear polynomial $P_k(x)$ are the minimum norm solution to a linear least squares problem. Even though the performance of the linear Shepard method is comparable to other Shepard's techniques [Gantovnik et al. 2004; Iyer et al. 2006], there are situations when statistically robust least squares fits are necessary. Applications of such local approximations are well known in image processing [Besl et al. 1989]. This section develops a robust linear Shepard algorithm, which is an option in the subroutine LSHEP.

4.1 Robust Linear Approximation Using M-estimation

By its inherent nature, least squares approximation allows "bad" data points to exert a disproportionate influence on the fit. A robust approximation is intended to be resistant to such deviant data points. Effectively, data points with larger residuals will have smaller weights and thereby less influence in the fit. M-estimation (maximum likelihood-type estimation) is a statistically robust technique that minimizes the sum $\sum \rho(r_i)$ of residuals r_i contributed by points being used in the fit.

In a robust linear fit using M-estimation, the coefficients of $P_k(x) = a(x - x^{(k)}) + f_k$ are chosen to minimize the weighted least squares error function

$$E(a) = \sum_{j=1}^{N_p-1} \rho\Big(P_k\big(x^{(i_j)}\big) - f_{i_j}\Big),$$

where the function ρ gives the contribution of each residual $r_j = P_k\big(x^{(i_j)}\big) - f_{i_j}$ to the total error $E(a)$. The number N_p here is the same as for the linear Shepard method. The function ρ should be C^1 and have the following properties: $\rho(r) \geq 0$, $\rho(0) = 0$, $\rho(r) = \rho(-r)$, and $\rho(r_s) \geq \rho(r_t)$ for $r_s > r_t > 0$.

Let ψ be the derivative of ρ. Differentiating the function $E(a)$ with respect to the coefficients a, and setting the partial derivatives to zero, produces a system of equations

$$\sum_{j=1}^{N_p-1} \psi\Big(a\Big(x^{(i_j)} - x^{(k)}\Big) + f_k - f_{i_j}\Big)\Big(x^{(i_j)} - x^{(k)}\Big) = 0.$$

Let the weight function be defined as $\omega(r) = \psi(r)/r$, $r \neq 0$ and $\omega(r) = 1$, $r = 0$, and let $\omega_j = \omega(r_{i_j})$. The system of equations can then be written as

$$\sum_{j=1}^{N_p-1} \omega_j\Big(a\Big(x^{(i_j)} - x^{(k)}\Big) + f_k - f_{i_j}\Big)\Big(x^{(i_j)} - x^{(k)}\Big) = 0.$$

Solving this system of equations is a weighted least squares problem, similar to the one in Section 3.7. The only difference is that now ω_j depends on the unknown

solution a, which makes it a nonlinear system. The standard approach to solve this problem, called iteratively reweighted least squares (IRLS), is to compute and fix the weights ω_j for given a, then solve a *linear* system for new a, and iterate until convergence.

When the Huber minimax function [Huber 1981] is used for ψ, the data points with larger residuals (outliers) will be "downweighted" proportional to the magnitude of the residual error. The weights will never be equal to zero, unless the residual is infinite. If the ψ function is redescending with the property that $\psi(r) = 0$ for $|r| > c$ like the bisquare minimax function [Huber 1981], where c is a preselected cutoff value determined, in part, by the scale estimate of the residual error, then the data points with large residual errors are completely ignored. The scale of the residual error is estimated by the rescaled median absolute residual (MAR).

In practice, five iterations of IRLS with the Huber minimax function as ψ are applied first with all initial weights equal one. Five iterations of IRLS with the bisquare function are applied after that to give less weight to extreme outliers. Since IRLS with the bisquare ψ may diverge, $E(a)$, using the bisquare ρ function, for the estimate of a after the five Huber iterations is compared to $E(a)$ with the final a. If the latter is larger, then the estimate of a from the Huber iterations is used for $P_k(x)$. The influence radius $R_w^{(k)}$ is then adjusted such that the closed ball $\overline{B(x, R_w^{(k)})}$ only includes points with weights larger than some threshold, say 0.8. This adjustment of $R_w^{(k)}$ only for robust local approximations is empirically validated.

4.2 Motivation for a Hybrid Statistically Robust Estimation Algorithm

The breakdown bound [Huber 1981, Rousseeuw 1987] for the M-estimator is $1/(p+1)$, where p is the number of parameters to be estimated. The value of the breakdown bound for the M-estimator is low. This means that a large number of data points might be required to obtain robust estimates. Using the M-estimator with the linear Shepard method constructs better approximations than the standard linear Shepard method [Iyer and Watson 2006]. However, the primary advantage of the linear Shepard method is that, in m-dimensional space, it requires $\mathcal{O}(m)$ data points to construct the fit, which is no longer possible when M-estimation is used, since the number of points required by M-estimation is $\mathcal{O}(m^2)$. (If F is the fraction of points that are outliers, $(m+2)F \leq \frac{1}{(m+2)}(m+2) \Rightarrow (m+2)^2 F \leq m+2$, so $\mathcal{O}(m)$ outliers requires $\mathcal{O}(m^2)$ data points.) This computational complexity problem (the requirement for $\mathcal{O}(m^2)$ data points) is solved by using a different robust estimation technique, least median of squares (LMS) [Rousseeuw 1984], which has a breakdown value of $1/2$, independent of the dimension m. Thus, LMS would require $\mathcal{O}(m)$ data points to construct a robust linear Shepard approximation. LMS achieves this optimal breakdown value by constructing all $\binom{n}{p+1}$ possible fits to subsets of $p+1$ points, where again p is the number of parameters being estimated. For some n and p, this is practical, but in general this factorial complexity is computationally untenable. This motivates the use of a hybrid statistically robust method,

somewhere between M- and LMS-estimation, that requires fewer data points than the quadratic Shepard method and yet produces better approximations than the standard linear Shepard method.

5. RIPPLE

RIPPLE [Iyer and Watson 2007] is an acronym for *residual initiated polynomial-time piecewise linear estimation*, a new algorithm described next. As seen in the previous sections, as the space dimension m increases, the amount of data required for quadratic nodal functions $P_k(x)$ becomes computationally prohibitive. In general, the use of polynomials of degree < 2 is not sufficient to locally describe the behavior of highly nonlinear functions. However, in many engineering problems, the individual response functions are piecewise linear and thus can be approximated by using linear local approximations. An example of such a function is the penalty function based fitness function used for the design of composite structures via genetic algorithms [Gantovnik et al. 2004]. The core idea is that away from the breakpoints between the linear pieces, the standard linear Shepard approximation is adequate. Near a piecewise linear breakpoint, a robust linear estimator may be able to choose the "correct" linear piece, resulting in an accurate approximation, $\tilde{f}(x)$.

The standard linear Shepard algorithm uses weighted linear least squares to construct the local approximation. It uses all the data to construct the local fit. As a result, it does not produce good approximations near a piecewise linear function ridge as it "averages" all the facets near the ridge. When M-estimation is used, the linear Shepard method ideally picks the facet that is consistent with a majority of the nearby data. Even though this is usually better than the standard linear Shepard method, there are cases when the required majority of data points may not lie on the same facet as the point of approximation. This produces large errors, which can be reduced if the points to be used in the fit are chosen carefully, as LMS estimation would have done.

Let $S = \{i_1, i_2, \ldots, i_{N_p-1}\}$ be the set of indices as defined in Section 3.7. The most consistent points (not necessarily a majority) in the set S must be used for the local least squares approximation. Suppose that based on some criteria, the points in this set could be classified as 'inliers' and 'outliers' such that each inlier produces a lower absolute residual (vertical distance to the approximation) than all outliers. Intuition suggests that if a point is classified as an inlier, the points close to it have a higher probability of being classified as an inlier. However, if two points that have equal distance from an inlier are on two different facets, then the corresponding function values are "averaged", resulting in a large approximation error. Thus, it is not possible to pick the inliers based solely on the criterion of distance from the point of approximation x. Also, it is not possible to predict how many data points will be used for constructing the local approximation.

Suppose that the best set (set of points producing minimum sum of squared residuals) of the minimum number of points required to construct the fit is chosen. It is now possible to examine every other data point and determine whether it

should be added to the best set based on the residual that is produced. Thus, the problem is now reduced to finding the best minimal set of points. The idea for choosing minimal sets of points required to construct the fit has been borrowed from the RANSAC (random sample consensus) algorithm [Fischler and Bolles 1981], which chooses the minimal sets of points at random. If the data dimension is m, the number of parameters in $P_k(x)$ to be estimated is m, requiring at least $(m+1)$ data points to construct a linear approximation. The total number of data points is $N_p - 1$. The best minimal set lies among $\binom{N_p-1}{m+1}$ possible sets. However, in general this has exponential complexity and is therefore untenable.

The best minimal set of points can be approximated in polynomial time by using a special distance criterion that is described as follows. Define the $(N_p-1)\times(m+3)$ distance matrix D such that $D_{.1} = (i_1, i_2, \ldots, i_{N_p-1})^t$ and $D_{i,j+1}$ is the index of the point that has least Euclidean distance from the point whose index is $D_{i,j}$, for $i = 1$ to $N_p - 1$ and $j = 1$ to $m + 2$, subject to the constraint that every row of D must contain distinct indices. In case $x^{(i_s)}$ and $x^{(i_t)}$ are equidistant from $x^{(D_{i,j})}$, choose the one closer to $x^{(k)}$. If that choice also results in a tie, use index i_s (assuming $i_s < i_t$).

Compute the local approximation $P_k(x)$ using $x^{(k)}$ and different sets of $(m + 1)$ distinct points $x^{(i)}$ that are picked as follows. For each row t, $t = 1$ to $N_p - 1$, in D, pick the point corresponding to index $D_{t,1}$. The remaining m points can be picked in $\binom{m+2}{m}$ ways from row t. Thus, the total number of local approximations $P_k(x)$ computed using $x^{(k)}$ and a set of $(m+1)$ other points will be $(N_p-1)\binom{m+2}{m}$. For each approximation, compute the absolute residuals (vertical distances) for the $(m + 1)$ points used to compute the least squares fit. Record which set of $(m + 1)$ points produces the minimum least squares error. Let $R = \{i_1, i_2, \ldots, i_{m+1}\}$ correspond to the indices of these $m + 1$ points. In case of a tie for this best set, choose the set containing the points closer to $x^{(k)}$ (precisely, sort the distances $\|x^{(i_j)} - x^{(k)}\|$ into an $(m + 1)$ vector in increasing order, then compare the two vectors lexicographically; if these distances are equal, sort and compare lexicographically the sets of indices).

Once the best set of minimal points is obtained, compute the linear interpolant at $x^{(k)}$. Compute the average residual produced by the points $x^{(i)}$, $i \in R$, and record the coefficients of the linear interpolant. Use M-estimation to determine the set $T \supset R$ of points for computing the final local approximation. For solving the nonlinear system using the iteratively reweighted least squares method, use the coefficients and MAR of the linear fit obtained from set R as the initial linear least squares approximation and scale estimate, respectively. Compute the final local approximation $P_k(x) = f_k + \sum_{j=1}^{m} a_j^{(k)} \left(x_j - x_j^{(k)} \right)$ using points $x^{(k)}$ and $x^{(i)}$, $i \in T$.

The algorithm RIPPLE described above is implemented in the subroutine RIPPLE in SHEPPACK, for arbitrary dimension m, and RIPPLE uses the same function LSHEPVAL as LSHEP uses to return the approximate function value $\tilde{f}(x)$. RIPPLE is specifically intended for sparse scattered data from approximately piecewise linear continuous functions. In other contexts RIPPLE may perform erratically compared to, say, LSHEP.

6. USAGE AND ORGANIZATION

The README file distributed with SHEPPACK describes the physical organization of the package into files, and includes the basic instructions for compiling, testing, and running the code. The following describes the organization of key modules in the package. The package consists of the module SHEPPACK, which contains module procedures CSHEP2, LSHEP, QSHEP2, QSHEP3, QSHEPM, and RIPPLE, which construct the approximations, module procedures CS2GRD, CS2HES, CS2VAL, LSHEPVAL, QS2GRD, QS2VAL, QS3GRD, QS3VAL, QSMGRD, and QSMVAL, which evaluate the approximation and its partial derivatives, and PRIVATE subroutines called by the various drivers. In addition, module REAL_PRECISION (a real precision module from HOMPACK90, Algorithm 777) has been included. For completeness, the package also includes all required LAPACK and BLAS routines.

7. PERFORMANCE

All the test results in the papers for QSHEP[23M]D and CSHEP2D have been replicated with the Fortran 95 translations in SHEPPACK, so there is no need to describe those tests again. Instead, the focus here is on the new codes LSHEP and RIPPLE, data sets with outliers, and piecewise linear functions in high dimensions. Five continuous, piecewise smooth test functions $f_i : [0,1]^m \to E$, $(i = 1, \ldots, 5)$, similar to functions occurring in many applications [Besl et al. 1989; Gantovnik et al. 2004], are used to test the performance of the algorithms in SHEPPACK. The test functions f_1, f_2, f_3, f_4, and f_5 for $m = 2$ are shown in Figures 1–5, respectively.

The function f_1 in m dimensions is

$$f_1(x) = \begin{cases} \frac{2}{m} \sum_{i=1}^{m} x_i, & \sum_{i=1}^{m} x_i \le \frac{m}{2}, \\ -\frac{2}{m} \sum_{i=1}^{m} x_i + 2, & \sum_{i=1}^{m} x_i > \frac{m}{2}, \end{cases}$$

which has the maximum $f_1(x^*) = 1$ at $\sum_{i=1}^{m} x_i^* = \frac{m}{2}$.

The function f_2 in m dimensions is

$$f_2(x) = 1 - \frac{2}{m} \sum_{i=1}^{m} |x_i - 0.5|,$$

which has the maximum $f_2(x^*) = 1$ at $x_i^* = 0.5$, for $i = 1, \ldots, m$.

The function f_3 in m dimensions is

$$f_3(x) = 1 - 2 \max_{1 \le i \le m} (|x_i - 0.5|),$$

which has the maximum $f_3(x^*) = 1$ at $x_i^* = 0.5$, for $i = 1, \ldots, m$.

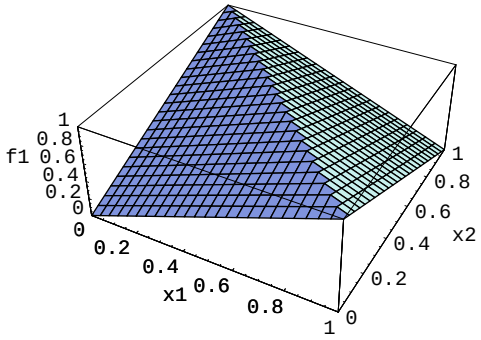

Fig. 1. Test function f_1.

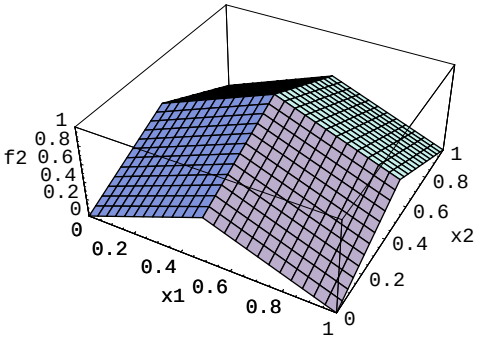

Fig. 2. Test function f_2.

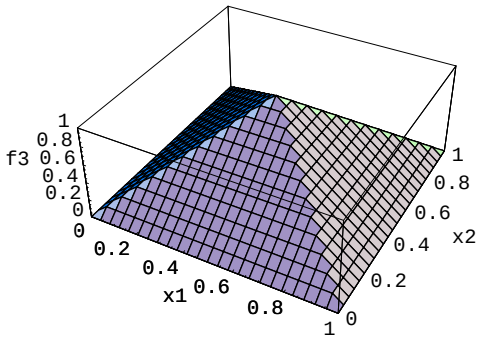

Fig. 3. Test function f_3.

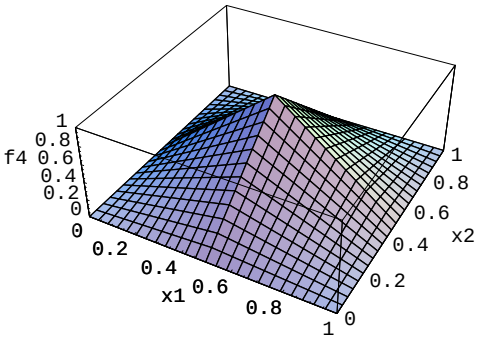

Fig. 4. Test function f_4.

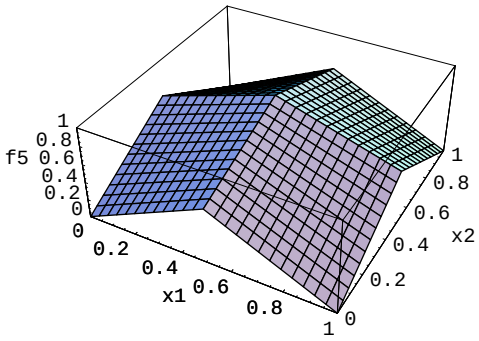

Fig. 5. Test function f_5.

The function f_4 in m dimensions is

$$f_4(x) = \prod_{i=1}^{m} g_i(x),$$

where

$$g_i(x) = \begin{cases} 2x_i, & x_i \leq 1/2, \\ 2(1 - x_i), & \text{otherwise}, \end{cases}$$

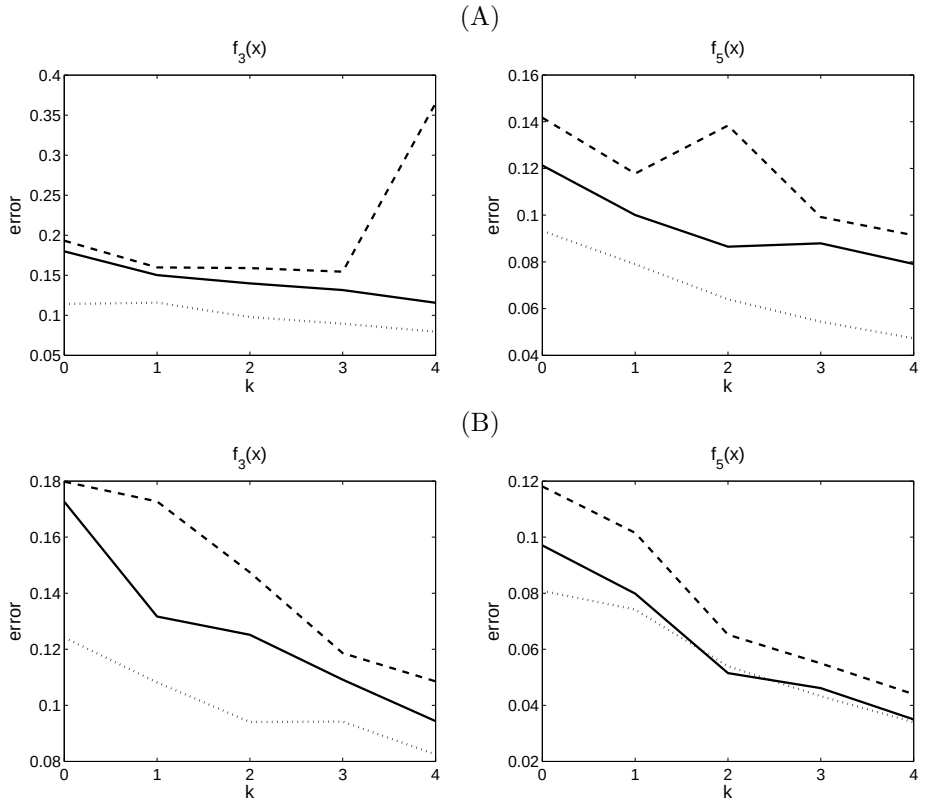

Fig. 6. $f_i(x)$, $i = 3$ and 5, $m = 5$. RMS error plots for LSHEP (solid), robust LSHEP (dashed), and RIPPLE (dotted) versus $n_0 = 100 \cdot 2^k$, $n_1 = 8$. (A) Noisy data. (B) Noisy data with outliers.

which has the maximum $f_4(x^*) = 1$ at $x_i^* = 0.5$, for $i = 1, \ldots, m$.

The function f_5 in m dimensions is

$$1 - \frac{1}{0.5m + 0.5^m} \left(\sum_{i=1}^{m} |x_i - 0.5| + \prod_{i=1}^{m} |x_i - 0.5| \right),$$

which has the maximum $f_5(x^*) = 1$ at $x_i^* = 0.5$, for $i = 1, \ldots, m$.

Each test function was sampled at n_0 random (uniformly distributed) scattered points of $Q = [0,1]^m$ to generate the data set $\{x^{(i)}\}_{i=1}^{n_0}$ used to construct the Shepard interpolants. The approximation error has been characterized using three error metrics. These are the maximum absolute error $e_{\max}$, the mean absolute error $\bar{e}$, and the root mean squared error e_r. The absolute approximation error is defined as

$$e_i = |\tilde{f}(z^{(i)}) - f(z^{(i)})|,$$

where $\tilde{f}(x)$ is the interpolant function, $f(x)$ is the test function, and $z^{(i)}$ are the points of a uniform $n_1 \times \cdots \times n_1$ grid $G \subset [0.1, 0.9]^m$. The total number of grid points is $n = n_1^m$.

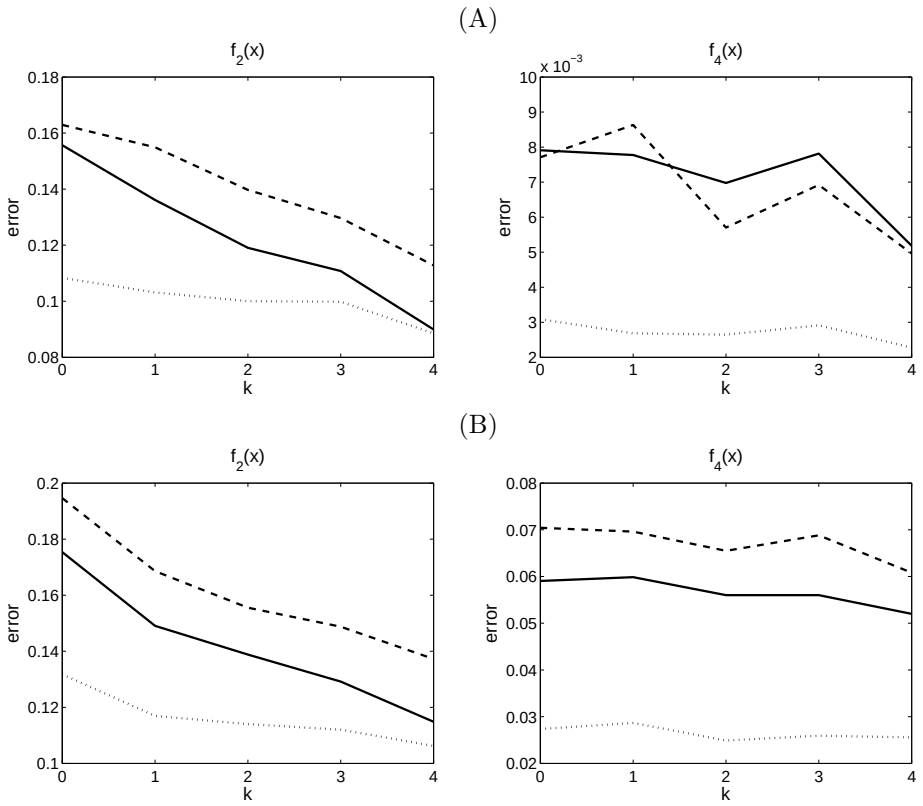

Fig. 7. $f_i(x)$, $i = 2$ and 4, $m = 10$. RMS error plots for LSHEP (solid), robust LSHEP (dashed), and RIPPLE (dotted) versus $n_0 = 100 \cdot 2^k$, $n_1 = 4$. (A) Noisy data. (B) Noisy data with outliers.

Using this notation, the maximum absolute error is

$$e_{\max} = \max_{1 \le i \le n} e_i,$$

the mean absolute error is

$$\bar{e} = \frac{1}{n} \sum_{i=1}^{n} e_i,$$

and the root mean squared error is defined as

$$e_r = \sqrt{\sum_{i=1}^{n} e_i^2 \Big/ n}.$$

Figure 6 shows the interpolation errors for two test problems f_3, f_5 with $m = 5$ (five-dimensional) and the results for the three other test problems are similar. The results are only shown under the root mean square error measure, since the maximum absolute error $e_{\max}$ offers little information as $e_{\max}$ is usually determined

by points $z^{(i)}$ (far from all the interpolation points $x^{(i)}$) where all local linear approximations have little or no effect, and the behavior of the mean absolute error $\bar{e}$ is very similar to e_r.

The first test is conducted with $N(0, 10^{-6})$ normally distributed noise (zero mean, standard deviation 0.001) added, with results shown in Figure 6(A). Next, to test how well outliers in the data are dealt with, contaminated function values

$$f_i(x) + 0.001\,\mu + 0.1\,\chi_{[0,0.2]}(\nu),$$

are used, where μ is a $N(0,1)$ normally distributed random variable, ν is a $U[0,1]$ uniformly distributed random variable, and $\chi_{[0,0.2]}$ is the characteristic function of the interval $[0, 0.2]$. The effect is to add small noise everywhere and large noise producing an outlier with probability 0.2. Note that since the Shepard approximation $\tilde{f}(x)$ interpolates all the data points $\left(x^{(i)}, f_i\right)$, if x is close to an outlier $x^{(i)}$ then $\tilde{f}(x)$ will have a large error. The goal is to have $\tilde{f}(x)$ for x away from outliers to be controlled by inliers, and thus be accurate. Although SHEPPACK does not support this, one could consider modifying the outlier data values to be consistent with the inlier data, and use a Shepard approximation based on these modified data values. This strategy, for piecewise linear functions, is called the "slope facet model" in image processing, and is a special case of more sophisticated "variable order" facet models [Mainguy et al., 1995]. Figure 6(B) shows the results of using the noisey data with outliers. Figure 7 is analogous to Figure 6 for two other test problems f_2 and f_4 with $m = 10$ (ten-dimensional).

In most situations, RIPPLE has less RMS error than LSHEP (the codes using statistically nonrobust least squares estimation) and the robust linear estimation option in LSHEP. However, the robust linear estimation option in LSHEP does not always improve and generally degrades the approximation result without the option. The reason for this is illustrated in Figure 8, which shows two simple one-dimensional examples. Plots on the left side show a case where LSHEP with the robust option could improve the approximation result, however, plots on the right show a case where the robust option makes the approximation worse.

The differences between all these are explained by the different ways weights are determined when constructing the local linear approximation P_k about each sample point $x^{(k)}$. In LSHEP without the robust option, weights are determined totally by distances between other sample points and $x^{(k)}$. However, in LSHEP with the robust option, weights are determined mostly by function values, i.e., points with function values closer to f_k tend to get larger weights. In RIPPLE large weights are credited to points that form the best local fit around $x^{(k)}$ (best in the sense of producing minimum least squares error). The numerical results show the RIPPLE strategy is the best when approximating piecewise functions in high dimensions, while robust LSHEP is not so good and nonrobust LSHEP is somewhere in between. However, RIPPLE is considerably more expensive in high dimensions than the robust LSHEP option, which performs well in the presence of a few outliers compared to LSHEP (cf. the left side of Figure 8). Thus the robust

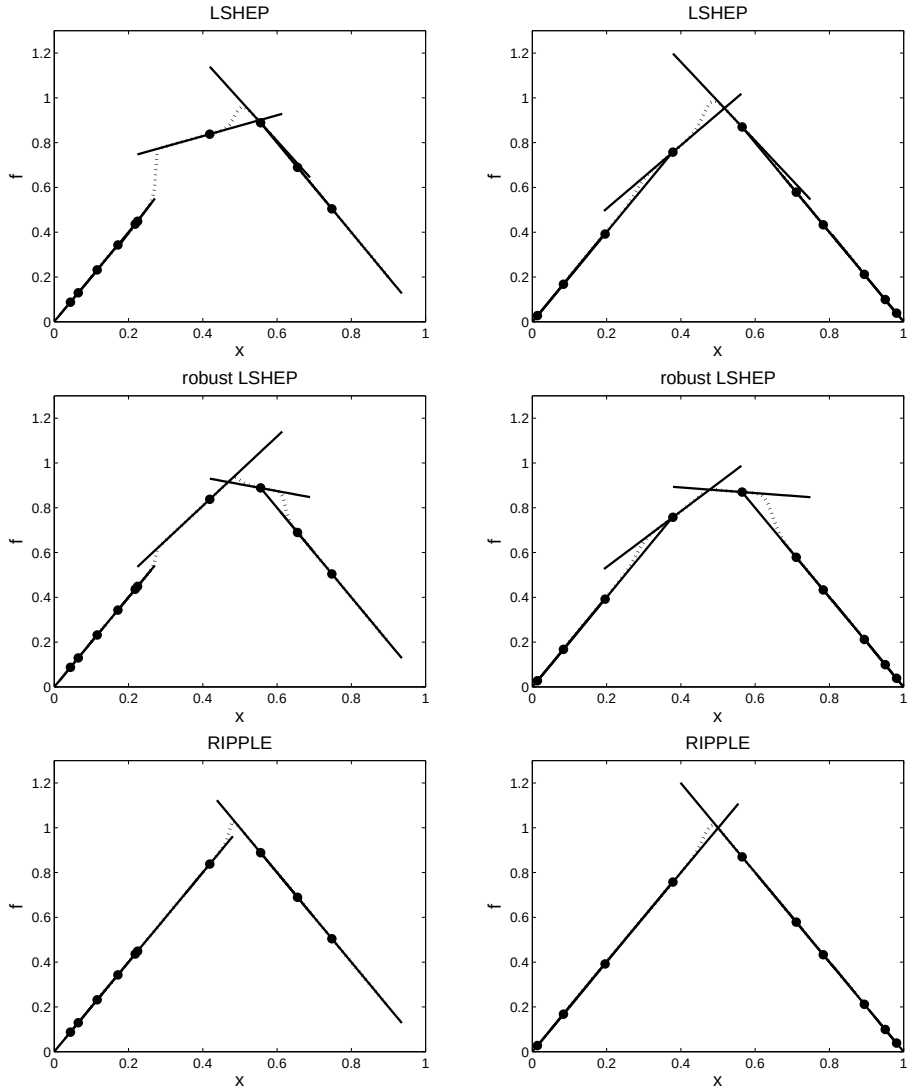

Fig. 8. Local linear approximation functions and Shepard approximation functions for $f_1(x)$ using LSHEP without the robust option, with the robust option, and RIPPLE, from two different sets of sample points. The dotted nodes are the data set $\{(x^{(i)}, f_1(x^{(i)}) + 0.001\mu)\}_{i=1}^{n_0}$ used to build local approximations and the radius of each local approximation (solid lines) is $R_w^{(i)}$. The dotted lines are the interpolant function $\tilde{f}(x)$. The sample size for the local fits is 3.

M-estimation option in LSHEP, while not generally the best choice, is still worth keeping as an option.

Figure 8 also shows an inherent problem of linear Shepard algorithms, especially for RIPPLE. In the two plots for RIPPLE in Figure 8, the local linear approximation functions are very close to the test function f_1, however, the approximation error is still quite significant around the vertex/ridge of the piecewise linear function.

This happens whenever the radius of influence of a sample point on one facet of the piecewise (linear) function extends over other facets of the function. In this case, even if the local approximation is correct, applying the local approximation within its radius of influence still results in a large approximation error. In the new code LSHEPVAL, a warning flag is turned on if the angle between the tangent plane of $\tilde{f}(x)$ at point $z^{(i)}$ and that of some local approximation $P_k(x)$ with nonzero weight $W_k(z^{(i)})$ is over a threshold of 30°, indicating potential proximity to a ridge/vertex of a piecewise (linear) function where the approximation error could be significant.

BIBLIOGRAPHY

Berry, M. W., and Minser, K. S. 1999. Algorithm 798: High-dimensional interpolation using the modified Shepard method. *ACM Transactions on Mathematical Software 25*, 3, 353–366.

Besl, P. J., Birch, J. B., and Watson, L. T. 1989. Robust window operators. *Machine Vision and Applications 2*, 4, 179–191.

Booker, A. J., Conn, A. R., Dennis, J. E., Frank, P. D., Trosset, M., and Torczon, V. 1995. Global modeling for optimization: Boeing/IBM/Rice Collaborative Project. Text provided by Paul D. Frank of Boeing.

Cheney, W., and Light, W. 1999. A Course in Approximation Theory. 1st edition, Brooks/-Cole, Pacific Grove, CA.

Coons, S. 1967. Surfaces for computer-aided design of space forms. Technical Report TR-41, Massachusetts Institute of Technology, Cambridge, MA, USA.

Demetriou, I. C. 2007. Algorithm 863: L2WPMA, a Fortran 77 package for weighted least-squares piecewise monotonic data approximation. *ACM Transactions on Mathematical Software 33*, 1.

Franke, R., and Neilson, G. 1980. Smooth interpolation of large sets of scattered data. *International Journal of Numerical Methods in Engineering 15*, 1691–1704.

Fischler, M. A. and Bolles, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24*, 381–395.

Friedman, J. H. 1991. Multivariate adaptive regression splines. *Annals of Statistics 19*, 1–141.

Gantovnik, V., Gürdal, Z., and Watson, L. T. 2004. Linear Shepard interpolation for high dimensional piecewise smooth functions. Proceedings of 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, NY, CD-ROM, 15 pages.

Giunta, A. A. 1997. Aircraft multidisciplinary design optimization using design of experiments theory and response surface modeling methods. Ph.D. thesis, Department of Aerospace and Ocean Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA.

Huber, P. 1981. *Robust Statistics*. John Wiley and Sons, New York.

Iyer, M. A. and Watson, L. T. 2006. An interpolation method for high dimensional scattered data. Proceedings 2006 Spring Simulation Multiconference, Business and Industry Symposium, J. A. Hamilton, Jr., R. MacDonald, and M. J. Chinni (eds.), Society for Modeling and Simulation International, San Diego, CA, 217–222.

Iyer, M. A. and Watson, L. T. 2007. RIPPLE: Residual Initiated Polynomial-time Piecewise Linear Estimation. Proceedings IEEE Southeastcon 2007, Richmond, VA, 444–449.

Koehler, J. R., and Owen, A. B. 1996. Computer experiments. In *Handbook of Statistics*, S. Ghosh and C. R. Rao (eds.), vol. 13. Elsevier Science, Oxford, UK, 261–308.

Mainguy, Y., Birch, J. B., and Watson, L. T. 1995. A robust variable order facet model for image data. *Machine Vision and Applications 8*, 141–162.

Micchelli, C. A. 1986. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation Theory 2*, 11–22.

Osio, I. G., and Amon, C. H. 1996. An engineering design methodology with multistage Bayesian surrogates and optimal sampling. *Research in Engineering Design 8*, 189–206.

PIEGL, L. 1989a. Modifying the shape of rational B-splines. part 1:curves. *Computer-Aided Design 21*, 8, 509–518.

PIEGL, L. 1989b. Modifying the shape of rational B-splines. part 2:surfaces. *Computer-Aided Design 21*, 9, 538–546.

RENKA, R. J. 1988a. Multivariate interpolation of large sets of scattered data. *ACM Transactions on Mathematical Software 14*, 2, 139–148.

RENKA, R. J. 1988b. Algorithm 660: QSHEP2D: Quadratic method for bivariate interpolation of scattered data. *ACM Transactions on Mathematical Software 14*, 2, 149–150.

RENKA, R. J. 1988c. Algorithm 661: QSHEP3D: Quadratic method for trivariate interpolation of scattered data. *ACM Transactions on Mathematical Software 14*, 2, 151–152.

RENKA, R. J. 1999a. Algorithm 790: CSHEP2D: Cubic Shepard method for bivariate interpolation of scattered data. *ACM Transactions on Mathematical Software 25*, 1, 70–73.

RENKA, R. J. 1999b. Algorithm 791: TSHEP2D: Cosine series Shepard method for bivariate interpolation of scattered data. *ACM Transactions on Mathematical Software 25*, 1, 74–77.

ROUSSEEUW, P. J. 1984. Least median of squares regression. *Journal of American Statistical Association 79*, 388, 871–880.

ROUSSEEUW, P. J. AND LEROY, A. M. 1987. *Robust Regression and Outlier Detection*. John Wiley and Sons, New York.

SACKS, J., WELCH, W. J., MITCHELL, T. J., AND WYNN, H. P. 1989. Design and analysis of computer experiments. *Statistical Science 4*, 4, 409–435.

SCHOENBERG, I. J. 1938. Metric spaces and completely monotone functions. *Annals of Mathematics 49*, 811–841.

SHEPARD, D. 1968. A two-dimensional interpolation function for irregularly spaced data. In *Proceedings of the 1968 23rd ACM National Conference*, 517–523.

SIBSON, R., AND STONE, G. 1991. Computation of thin-plate splines. *SIAM Journal on Scientific and Statistical Computing 12*, 1304–1313.

VERSPRILLE, K. J. 1975. Computer-aided design applications of the rational B-spline approximation form. Ph.D. thesis, Syracuse University, Syracuse, NY.