Acc. Sc. Torino
Memorie Sc. Fis. 33 (2009) ANALISI NUMERICA

Accurate approximation of unknown fault lines from scattered data

Memoria di GIAMPIETRO ALLASIA,
RENATA BESENGHI e ROBERTO CAVORETTO*
presentata dal Socio corrispondente Giampietro ALLASIA nell'adunanza del 26 novembre 2008

Abstract. *In a previous paper we proposed a method for the detection of fault lines of a surface only known at scattered data. Now, we present some techniques suitable to give accurate approximation of the detected faults. First, we improve the technique of detection of fault points, picking out and collecting in a set all the data points close to fault lines. To select these points we use a cardinal radial basis interpolation formula. Then, applying a powerful refinement technique, we discuss different methods for the accurate approximation of fault lines, considering procedures based on polygonal line, least squares, and best l_∞ approximations. Numerical results point out the efficiency of our approach.*

Keywords: scattered data, fault lines, detection and approximation methods, radial basis functions.

Riassunto. *In un precedente articolo abbiamo proposto un metodo per l'individuazione delle linee di faglia di una superficie nota solo su dati sparsi. Ora presentiamo alcune tecniche adatte a dare un'approssimazione accurata delle faglie individuate. Dapprima, miglioriamo la tecnica di individuazione dei punti di faglia, identificando e raccogliendo in un insieme tutti i punti vicini alle linee di faglia. Per selezionare questi punti usiamo una formula di interpolazione a base radiale cardinale. Quindi, applicando una potente tecnica di raffinamento, proponiamo metodi differenti per ottenere un'approssimazione accurata delle linee di faglia, considerando procedure basate su approssimazioni poligonali, sui minimi quadrati e sull'approssimazione ottima. I risultati numerici mostrano l'efficienza del nostro approccio.*

Parole chiave: dati sparsi, linee di faglia, metodi di individuazione e approssimazione, funzioni a base radiale.

Mathematics Subject Classification 2000: 65D05, 65D10, 65D17.

*Department of Mathematics, University of Turin, via Carlo Alberto 10, I–10123 Torino, Italy.
E-mail: `giampietro.allasia, renata.besenghi, roberto.cavoretto@unito.it`

1. Introduction

We consider the problem of the detection and approximation of fault lines of a surface only known on a finite number of scattered data. Fault detection is obviously a preliminary and necessary phase, but here we will focus mainly on accurate fault approximation. Predicting the location of fault lines and obtaining accurate approximations of them enable the construction of constrained approximation models of the surface overcoming common problems such as over-smoothing.

Some applications require only a rough knowledge of faults, in the sense that their number, position, and behaviour are significant, but accurate approximations of fault lines are not so important. This may be the case, for example, in oil industry, where fault detection provides useful information on the occurrence of oil reservoirs (see, e.g., [16, 13, 14]).

On the contrary, other applications call for accurate approximations of fault lines, as it happens in the reconstruction of discontinuous surfaces when images are required to be highly reliable. As an example, we may refer to medical images by magnetic resonance in which discontinuity lines may indicate the presence of some pathology (see, e.g. [23]). Also in some geophysical problems, a faithful representation of irregular surfaces by an accurate data fitting process is of great importance (see, e.g. [15]).

In the last two decades several authors have analyzed the fault detecting and approximating problem, as well as the discontinuous surface approximating problem with different techniques and methodologies. To get an idea one can see References, and, in particular, the monograph by Arcangéli, López de Silanes and Torrens [4].

Each method has its own performance and range of application, but all of them must unavoidably match the quantity of information, namely, the number of data and their distribution. Hence, the quantity of information may lead to choose in a specific application a method rather than another. Clearly, none of the existing methods is universally satisfactory, and understanding their limitations is an important key in applying them successfully.

In the present paper we recall in Section 2 the technique to detect fault lines discussed in a previous work [3], adding explication on the choice of the threshold value σ_0, a crucial parameter. As basic tools for fault point detection, i.e. points close to faults, we use a reliable cell-based search method and a *cardinal radial basis interpolant* (CRBI), that is, Shepard's formula possibly with suitable changes. The output of this procedure is a set of fault points and then a set of barycentres, generally closer to the faults than the fault points. In Section 3 the barycentres are processed in various steps in order to handle complex fault situations. In Section 4 we point out the different techniques of approximation, outlining in details the various procedures. Section 5 contains some numerical results which show the effectiveness of our method. The use of conventional approximation tools, if properly adapted to the topic, does not lead to instability phenomena or undesiderable oscillations which can locally or even globally hinder the approximation. Finally, in Section 6, we make some remarks, giving an idea of possible developments.

2. Detection of fault lines

To characterize the data points on or close to the fault lines, named *fault points*, first we consider a procedure based on local data interpolation by CRBIs, where the difference between any known function value and the related value of the interpolant is computed and compared with a threshold parameter. Then, we introduce and define a new set of points, named *barycentres*, generally closer to the faults than the fault points.

2.1. *Detection of fault points*

Let $S_n = \{P_k,\ k = 1,\dots,n\}$ be a set of distinct and scattered data points in a domain $D \subset \mathbb{R}^2$, and $\{f(P_k),\ k = 1,\dots,n\}$ a set of corresponding values of an unknown function $f : D \to \mathbb{R}$, which is discontinuous across a set $\Gamma = \{\Gamma_j,\ j = 1,\dots,m\}$ of fault lines

$$\Gamma_j = \{\gamma_j(t) : t \in [0,1]\} \subset D,$$

where γ_j are unknown parametric continuous curves. On $D \setminus \Gamma$ the function f is supposed to be sufficiently smooth. The domain D is bounded, closed, simply connected, and contains the convex hull of S_n.

Initially, the cell-based search method by Bentley and Friedman [7] (adopted also by Renka [20]) is applied to find the data point set $\mathcal{N}_{P_k}$ neighbouring to each point P_k of the data set S_n. This preprocessing phase is a classic *nearest-neighbour searching procedure* (NNSP), in which we make a subdivision of the domain D in cells and identify the set $\mathcal{N}_{P_k}$ for each P_k. Then, to localize the fault points of S_n that are close to a fault line, we propose a local interpolation scheme involving CRBIs, as alternative to the scheme based on thin plate splines first suggested by Gutzmer and Iske [16], and then reconsidered and extended by Crampton and Mason [12]. Therefore, considering the global *Shepard's formula* (see, e.g., [1])

$$F(P;S_n) = \sum_{i=1}^{n} f_i \, \frac{d(P,P_i)^{-\mu}}{\sum_{j=1}^{n} d(P,P_j)^{-\mu}}, \qquad F(P_i;S_n) = f_i, \quad i = 1,\dots,n,$$

where $d(P,P_i)$ is the Euclidean distance in $\mathbb{R}^2$ and $\mu > 0$, we adapt it to our purposes and, in particular, to the cell structure of the domain D. Thus, we use a local Shepard's formula on each set $\mathcal{N}_{P_k}$, $k = 1,\dots,n$, (note that $P_k \notin \mathcal{N}_{P_k}$)

$$\begin{aligned} F(P_k;\mathcal{N}_{P_k}) &= \sum_{i=1}^{n_{\mathcal{N}_{P_k}}} f_i \, \frac{d(P_k,P_i)^{-2}}{\sum_{j=1}^{n_{\mathcal{N}_{P_k}}} d(P_k,P_j)^{-2}}, \qquad (1) \\ F(P_i;\mathcal{N}_{P_k}) &= f_i, \quad i = 1,\dots,n_{\mathcal{N}_{P_k}}, \end{aligned}$$

$n_{\mathcal{N}_{P_k}}$ being the number of points of $\mathcal{N}_{P_k}$.

Let us consider now the absolute value of the difference $\sigma(P_k)$ between the function value $f(P_k)$ and the value of the interpolant at P_k, i.e.

$$\sigma(P_k) = \left| f(P_k) - F(P_k;\mathcal{N}_{P_k}) \right|. \qquad (2)$$

Supposing that the interpolant gives a good approximation to f in D, then $\sigma(P_k)$ gives a measure of smoothness of the function around P_k. Choosing a suitable threshold value $\sigma_0 > 0$, we compare the values $\sigma(P_k)$ and σ_0: if $\sigma(P_k)$ is less than or equal to σ_0 then f is smooth in a neighbourhood of P_k, or else there is a steep variation of f at P_k and accordingly P_k will be marked as a fault point. When this procedure has involved all the data points, we have characterized the *set of fault points*

$$\mathcal{F}(f;S_n) = \{P_k \in S_n : \sigma(P_k) > \sigma_0\},$$

which consists of all the data points which are expected either to belong to the faults or to be close to them.

A crucial point is an optimal choice of the threshold value σ_0. It is, in general, a difficult task, because the finite number of function values is the only available information. We start computing the largest deviation

$$S = \max\{|f_i - f_j| : i > j, \text{ for all } i,j = 1,2,\dots,n\},$$

because obviously $0 < \sigma_0 < S$. An appropriate sorting procedure, included in the preprocessing phase, gives the set of all deviations and, at the same time, supplies some additional information on the variation of f.

To get information on the surface, its faults, and the parameter σ_0, starting from the set S_n of data points and the corresponding function values f_i, $i = 1,\dots,n$, we obtain a surface representation by the following local Shepard's formula

$$F(P;I_P) = \sum_{i=1}^{n_P} f_i \, \frac{d(P,P_i)^{-2}}{\sum_{j=1}^{n_P} d(P,P_j)^{-2}}, \qquad F(P_i;I_P) = f_i, \quad i = 1,\dots,n_P, \qquad (3)$$

where I_P is the set of the n_P data points closest to the point P, contained in a ball centred at P. In general, a suitable choice of n_P is 10. This Shepard's surface and the relative contour lines give very useful information about the surface behaviour and the position of possible faults. A different approach is proposed in [24] to obtain additional information about geological surfaces.

Then we go on to find an optimal value of σ_0, introducing a variable threshold parameter, namely

$$\sigma_0^{(t)} = \frac{S}{\sqrt{2^t}}, \qquad \text{for } t = 1, 2, \ldots. \tag{4}$$

Starting up an iterative procedure, we increase the value of t by one unit each time and compare $\sigma(P_k)$ with $\sigma_0^{(t)}$. When the number of the found fault points is considered consistent with information that we acquire analyzing Shepard's surface and the contour lines, the process ends successfully.

In general, a few iterations suffice to determine an optimal threshold value $\sigma_0^{(t^*)}$ and we assume $\sigma_0 \equiv \sigma_0^{(t^*)}$. Moreover, in some cases, one can shorten the process, starting with values of t greater than 1, as suggested by examination of Shepard's surface.

2.2. *Construction of barycentres*

The set of fault points $\mathcal{F}(f;S_n)$, considered in Subsection 2.1, must be further manipulated to obtain a new point set which supplies more information on the faults.

Given a fault point $P_k \in \mathcal{F}(f;S_n)$ and the corresponding nearest-neigh-bor set $\mathcal{N}_{P_k}$, we define $D_{P_k} = \{P \in D : d(P,P_k) \leq R_{P_k}\}$, where $R_{P_k} = \max\{d(P_i,P_k) : P_i \in \mathcal{N}_{P_k}\}$. Then we order the $N_{P_k}+1$ points in $\bar{\mathcal{N}}_{P_k} = \mathcal{N}_{P_k} \cup \{P_k\}$, being $N_{P_k} = \text{card}(\mathcal{N}_{P_k})$, so that $f(P_{k1}) \leq f(P_{k2}) \leq \cdots \leq f(P_{kN_{P_k}+1})$. The expected jump δ_k of f in the subdomain D_{P_k} is evaluated by

$$\delta_k = \max_{1 \leq l \leq N_{P_k}} \Delta f(P_{kl}),$$

where Δ is the forward difference operator. We set l_k the lowest value of the index $l \in \{1, \ldots, N_{P_k}\}$ for which δ_k is obtained.

The part $\Pi_k = D_{P_k} \cap \Gamma$ of a fault line separates the set $\Delta_k^L = \{Q \in \bar{\mathcal{N}}_{P_k} : f(Q) \leq f(P_{kl_k})\}$ of all points of $\bar{\mathcal{N}}_{P_k}$ with lower function values from the set $\Delta_k^H = \{Q \in \bar{\mathcal{N}}_{P_k} : f(Q) > f(P_{kl_k})\}$ of all points of $\bar{\mathcal{N}}_{P_k}$ with higher function values. If Δ_k^L or Δ_k^H is the empty set, then we enlarge $\mathcal{N}_{P_k}$ and repeat the process, so that $\bar{\mathcal{N}}_{P_k}$ contains points lying in the two parts of the subdomain separated by the fault line. Having determined Δ_k^L and Δ_k^H in this way, we calculate the barycentres A_k^L and A_k^H of Δ_k^L and Δ_k^H, respectively. Then we find $A_k = (A_k^L + A_k^H)/2$ and put it in $\mathcal{A}(f;S_n)$, the *set of the barycentres.*

3. Refinement procedure

In Section 2, we took care of detecting the position of fault lines without considering explicity the main goal of our work, i.e. to obtain an accurate approximation of fault lines. Now, we focus on this problem and describe briefly the proposed procedure.

In the first phase, we subdivide the domain D by a regular grid and point out the grid cells containing points of $\mathcal{A}(f;S_n)$, i.e. barycentres. Since each grid cell contains at least a barycentre, we calculate the barycentre of the points of $\mathcal{A}(f;S_n)$ in each cell, namely, the barycentre of the barycentres for each cell. In this manner, we obtain a further set $\mathcal{B}(f;S_n)$, whose points are generally closer to the faults than the barycentres.

In the second phase, we order the points of the set $\mathcal{B}(f;S_n)$, so that each point can be identified by an index. In this way, we obtain the ordered set

$$\mathcal{C}(f;S_n) = \{C_i,\ i = 1,\dots,m\},$$

where m is the number of points of $\mathcal{B}(f;S_n)$.

The nearest-neighbour searching procedure used in the sorting process to build $\mathcal{C}(f;S_n)$ is as follows:

1. Detection of any point of $\mathcal{B}(f;S_n)$ nearest to a side of the domain D and assumption of this one as the first probe point.

2. Search of the nearest-neighbour point to the actual probe point and assumption of this one as the new probe point, excluding the points already considered.

3. Stopping the process, when all points of $\mathcal{B}(f;S_n)$ are ordered.

The knowledge of $\mathcal{C}(f;S_n)$ is fundamental, because it allows us to handle complex situations as the presence of several faults, and intersections or bifurcations of faults. Indeed, the problem of splitting fault lines into their branches is, in general, a hard task for every method. Now, we can connect each point of $\mathcal{C}(f;S_n)$ with the following one by a straight line segment, so obtaining a preliminary, generally low accurate, approximation curve. Since each point of

$\mathcal{C}(f;S_n)$ is labelled and the direction of the ordered points is known, we can simply subdivide the set $\mathcal{C}(f;S_n)$ into a number q of subsets greater or equal to the number of fault lines. Each subset of $\mathcal{C}(f;S_n)$ is denoted by the symbol $\mathcal{C}_j(f;S_n)$, for $j = 1,2,\dots,q$, and it is still an ordered set.

In some cases, it is convenient to consider a further stage of refinement, called *iterative refinement*. It consists of two steps:

1. From the set $\mathcal{C}_j(f;S_n) = \{C_{i,j},\ i = 1,\dots,m_j, j = 1,2,\dots,q\}$, we consider $C_{i-1,j}$, $C_{i,j}$ and $C_{i+1,j}$, for $2 \le i \le m_j - 1$.
2. These three points are considered as the vertices of a triangle; then, we calculate the barycentre $C_{i,j}^{(1)}$, $2 \le i \le m_j - 1$, $j = 1,2,\dots,q$, for each of the $m_j - 2$ triangles, holding $C_{1,j}$ and $C_{m_j,j}$ fixed. These barycentres form the set $\mathcal{C}_j^{(1)}(f;S_n)$, $j = 1,2,\dots,q$.

Step 2 can be repeated, so obtaining the barycentres $C_{i,j}^{(k)}$, forming the set $\mathcal{C}_j^{(k)}(f;S_n)$, for $i = 2,\dots,m_j - 1$, $j = 1,2,\dots,q$, $k = 2,3,\dots$ We can see the good running of this technique in Figure 1 and Figure 2, where we compare the polygonal lines obtained by applying or not the iterative refinement.

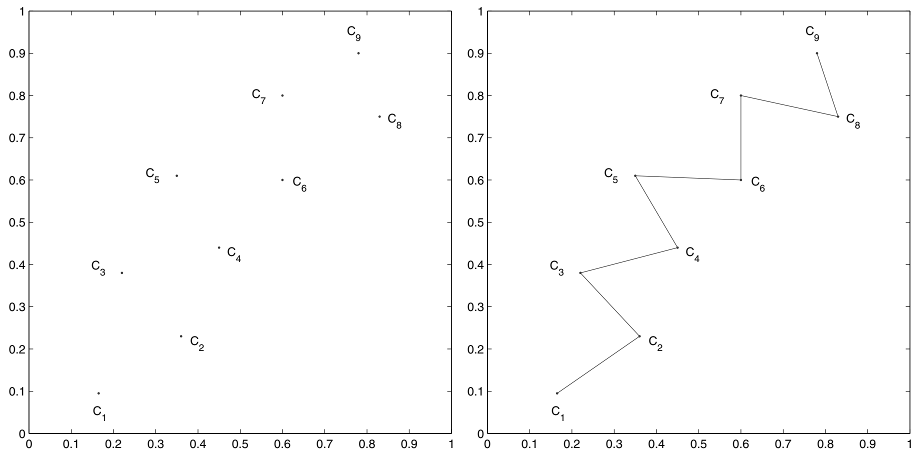

Figure 1: Set $C(f;S_n) = \{C_i,\ i = 1,\dots,9\}$ (left) and polygonal line obtained without applying the iterative refinement (right).

This approach, based on repeated averages, is simple but very powerful, because it allow us to construct smooth curves and obtain accurate approxima-

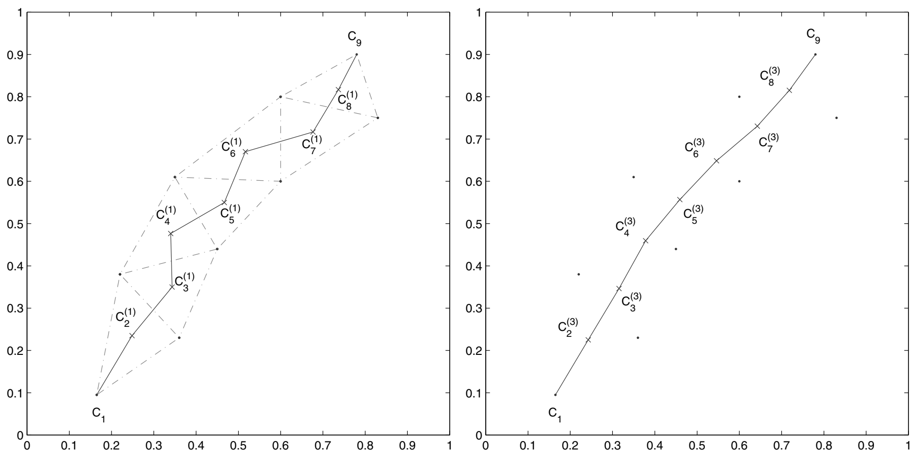

Figure 2: Polygonal line obtained by applying the iterative refinement to $C(f;S_n) = \{C_i,\ i = 1,...,9\}$ (left) and polygonal line obtained by using the iterative refinement with three iterations (right).

tions. In Section 5, we will discuss in detail some particularly difficult cases, in which the presence of bifurcations requires a splitting of fault lines.

4. Accurate approximation of fault lines

We consider different approaches to approximate fault lines, namely the polygonal line method, the least squares method, and the best l_∞ approximation method.

The polygonal method must be applied in all cases, because it is not only useful in itself, but also preliminary to the other methods.

The minimax approach, in general, assigns too much weight to a bit of data that is badly in error. The least squares approach puts substantially more weight on a point that is out of approximation curve with the rest of data but will not allow that point to completely dominate the approximation. In our case, the situation just described is little important, because the refinement procedure works out repeated averages.

4.1. *Polygonal line method*

The polygonal line method that we consider represents an improvement of the procedures developed by Gutzmer and Iske [16] and by Allasia, Besenghi and

De Rossi [3].

The ordered points of $\mathcal{C}(f;S_n)$ are connected by straight line segments, obtaining polygonal lines which approximate the fault lines. The performance of the approximation procedures depends, in particular, on the number of points, the cell dimension, and the form of the faults. Indeed, if a fault is centred in the cells, the polygonal method produces good results; otherwise, the polygonal line may appear too irregular and poorly accurate. To yield smoother (and, possibly, more accurate) polygonal lines, we use the iterative refinement and connect the points $C_{i,j}^{(k)}$ by straight line segments.

4.2. *Least squares method*

The least squares method can be used to approximate a data set $\{(x_i,y_i),\ i = 1,\ldots,m\}$ by an algebraic polynomial

$$P_s(x) = a_0 + a_1x + \cdots + a_{s-1}x^{s-1} + a_sx^s \tag{5}$$

of degree $s < m-1$. In our problem we choose the constants $a_0,a_1,\ldots,a_s$ to solve the *least squares problem*

$$\min_{a_0,\ldots,a_s} \sum_{i=1}^{m} [y_i - P_s(x_i)]^2 ,$$

where $(x_i,y_i) \in \mathcal{C}(f;S_n)$. Replacing the coefficients $a_0,a_1,\ldots,a_s$ in (5) with the values obtained by the least squares method, we get the polynomial approximating the fault line.

If there is only one fault in D, then the discrete least squares method can be advantageously applied. Otherwise, when we deal with complex situations (several faults, intersections or bifurcations of faults), the least square method cannot be applied directly, but a suitable subdivision of $\mathcal{C}(f;S_n)$ is required. Indeed, if the surface shows two or more fault lines, we need to split the set $\mathcal{C}(f;S_n)$ in a number of subsets greater or equal to the number of fault lines before applying the least squares method. This procedure is suggested also when a fault line is not of open type. Acting in this way, the least squares method yields very good results.

Finally, when a fault line is parallel or nearly parallel to y-axis, it is convenient to make first a rotation of the coordinate axes and then to apply the least squares method.

4.3. *Best l_∞ approximation method*

The best l_∞ (or Chebychev) approximation method can be considered as a tool for finding polynomial approximations to fault lines starting from the set $\mathcal{C}(f;S_n)$ or from some its refinement. The polynomial in (5) is chosen such as to minimize the largest absolute value of the differences $P_s(x_k) - y_k$ at $m < n$ discrete abscissae x_k, $1 \le k \le m$.

Given the abscissae x_k and the corresponding ordinates y_k, for $k = 1,2,\dots,m$, the polynomial in (5) is sought to solve the *minimax problem* (see, e.g., [6])

$$\min_{a_0,\dots,a_s} \max_k |P_s(x_k) - y_k| \,. \tag{6}$$

Introducing the *residuals* r_k at the abscissae x_k

$$P_s(x_k) - y_k = r_k, \quad k = 1,2,\dots,m,$$

(6) becomes

$$\min_{a_0,\dots,a_s} \max_k |r_k| \,.$$

Among the residuals r_k there exist at least one with the largest absolute value. Let us denote it by $H = \max_k |r_k| > 0$. Hence, we have the inequalities

$$|P_s(x_k) - y_k| \le H, \quad k = 1,2,\dots,m, \tag{7}$$

and the problem in (6) is equivalent to minimize the value H. Now, we rewrite (7) in the form

$$\left| \sum_{j=0}^{s} \left(\frac{a_j}{H}\right) x_k^j - \left(\frac{1}{H}\right) y_k \right| \le 1, \quad k = 1,2,\dots,m. \tag{8}$$

With the unknowns

$$\xi_1 = \frac{a_0}{H}, \;\; \xi_2 = \frac{a_1}{H}, \;\; \xi_3 = \frac{a_2}{H}, \;\dots,\; \xi_{s+1} = \frac{a_s}{H}, \;\; \xi_{s+2} = \frac{1}{H},$$

the conditions (8) read as follows

$$\left| \xi_1 + x_k \xi_2 + x_k^2 \xi_3 + \dots + x_k^s \xi_{s+1} - y_k \xi_{s+2} \right| \le 1, \quad k = 1,2,\dots,m. \tag{9}$$

To avoid expressions with absolute values, we observe that each inequality of (9) is equivalent to two inequalities. Moreover, the variable ξ_{s+2} equals the reciprocal value of the nonnegative quantity H which has to be minimized. Thus we obtain the following linear program

$$\text{maximize} \qquad Z = \xi_{s+2},$$

subject to the constraints

$$\begin{array}{rcr} \xi_1 + x_k\xi_2 + x_k^2\xi_3 + \ldots + x_k^s\xi_{s+1} - y_k\xi_{s+2} & \leq & 1, \\ \xi_1 + x_k\xi_2 + x_k^2\xi_3 + \ldots + x_k^s\xi_{s+1} - y_k\xi_{s+2} & \geq & -1, \end{array}$$

for $k = 1, 2, \ldots, m$, and

$$\xi_{s+2} \geq 0.$$

As only ξ_{s+2} has to satisfy a sign condition, the variables $\xi_1, \xi_2, \ldots, \xi_{s+1}$ are free. Since the standard procedure requires that all the variables have nonnegativity constrains, any problem containing variables allowed to be negative must be converted to an equivalent problem involving only nonnegative variables before the *simplex method* is applied. Each free variable ξ_j, $1 \leq j \leq s+1$, is replaced throughout the model by the difference of two new nonnegative variables

$$\xi_j = \xi_j^+ - \xi_j^-, \quad j = 1, 2, \ldots, s+1.$$

Since ξ_j^+ and ξ_j^- can have any nonnegative values, the difference $\xi_j^+ - \xi_j^-$ can have any value (positive and negative), so it is a legitimate substitute for ξ_j in the model. Now, applying the simplex method, we determinate the coefficients $a_0, a_1, \ldots, a_s$ of (5) and identify an approximating polynomial to the fault line.

The considerations previously developed for the least squares method are effective also for the best l_∞ approximation method. Hence, problems considering several faults, intersections or bifurcations of faults can be successfully solved. Obviously, in particularly difficult cases, the iterative refinement can be used in both the least squares method and the best l_∞ approximation method; this gives a greater smoothness to approximating curves.

We note that solving a minimization problem to find fault line approximation is proposed and carefully investigated by Crampton and Mason [12], who use the simplex method too.

5. Numerical results

In this section we propose a detailed investigation of a few test functions. They are examples of several numerical and graphical results, obtained by computational procedures developed in *C/C++*, *Matlab*, and *Maple* environments.

In the various tests we consider n randomly scattered data points $P_i = (x_i, y_i)$ in the square $[0,1] \times [0,1] \subset \mathbb{R}^2$, and the corresponding function values f_i, for $i = 1, \dots, n$, supposing that these are the only information relative to f at our disposal. In fact, in many real applications, one does not know the location of faults on the surface or even whether or not the surface is faulted [4].

The detection and approximation schemes are tested against functions with different kinds of faults, varying the dimension n of the sample generated by a uniform distribution and the threshold value σ_0. The results are obtained by using a local Shepard's formula, but other choices of CRBIs could work as well. Although the CRBI choice can appear quite elementary, the use of more advanced mathematical tools do not guarantee remarkable improvements. Moreover, the cell-based search method and the CRBIs allow to partition the domain, to process the data in some stages, and to insert or remove data points. These features are particularly important in surveying phenomena, such as geodetic or geophysical ones, whose data are distributed in regions with different characteristics.

In general, it is remarkable that, also reducing considerably (e.g., few thousand data points are sufficient) the dimension n of the scattered data set, the method of detection holds its efficiency. In this case a loss of approximation accuracy is unavoidable, but it depends essentially on the reduced information, that is, the number of data points. In fact, the method allows to individualize the position of fault lines all the same, outlining roughly their form.

If the set of obtained fault points is not satisfactory, the initial set of data points can be enlarged, but considering only new data points located in a smaller region including the fault. Then we can repeat the detection process and find a new and larger set of fault points. The adaptiveness of this approach ensures good results, though the number of data points is meaningfully reduced.

Test function 1

The test function

$$f_1(x,y) = \begin{cases} 1 + 2\left\lfloor 3.5\sqrt{x^2+y^2} \right\rfloor, & \text{if } (x-0.5)^2 + (y-0.5)^2 < 0.16, \\ 0, & \text{otherwise,} \end{cases}$$

is a surface with discontinuities across the set

$$\begin{aligned} \Gamma &= \Gamma_1 \cup \Gamma_2 \cup \Gamma_3 = \{(x,y) \in \mathbb{R}^2 : (x-0.5)^2 + (y-0.5)^2 = 0.16\} \\ &\cup \{(x,y) \in \mathbb{R}^2 : x^2 + y^2 = 16/49,\ (x-0.5)^2 + (y-0.5)^2 \le 0.16\} \\ &\cup \{(x,y) \in \mathbb{R}^2 : x^2 + y^2 = 36/49,\ (x-0.5)^2 + (y-0.5)^2 \le 0.16\}. \end{aligned}$$

As showed in some recent papers [16, 3, 17, 9], to which we refer for the graphical representation of the analytic function, the surface is always constant in $D \setminus \Gamma$, but presents faults intersected and/or bifurcated. In particular, first Gutzmer and Iske [16], and then Allasia, Besenghi and De Rossi [3] test their detection methods against this function using 2000 scattered data. López de Silanes, Parra and Torrens [17] also detect surface discontinuities using 2800 track data. This kind of data and, more in general, data disposed on a family of lines or curves can be advantageously treated by the method proposed by Allasia [2].

In the following, to compare our results with those obtained by other authors, we will restrict ourselves to use $n = 2000$ data points. Obviously, the proposed method works better considering a larger number of data points, e.g. $n = 4000$, 10000, as showed by errors in Table 1 and Table 2.

Starting from the set S_n of 2000 data points and the function values f_i, $i = 1, \ldots, n$, we use the formula (3) that, in general, is applied to approximate a continuous surface. The obtained surface graphic and the respective contour lines, in Figure 3, give us a preliminary, but very useful, information on the surface, in general, and on the dislocation of possible fault lines in particular.

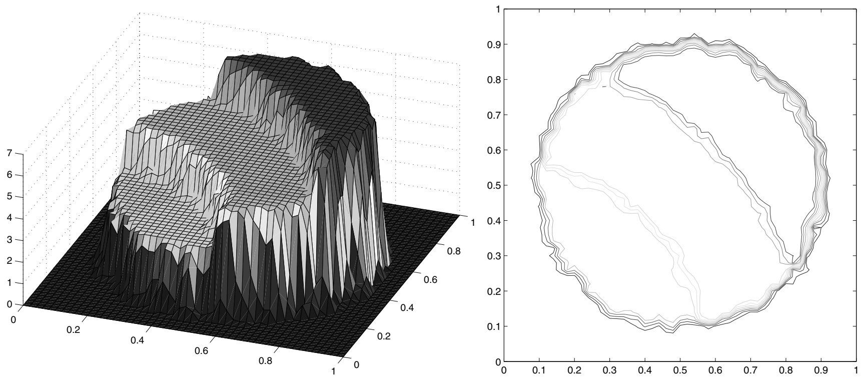

Figure 3: Shepard's surface (left) and contour lines (right) of f_1.

After calculating the largest function oscillation, $S = 7$, and applying the cell-based search method in the preprocessing phase, we proceed with the calculus of the threshold value $\sigma_0^{(t)}$, $t = 1, 2, \ldots$, that occurs with $t^* = 12$ iterations, namely $\sigma_0 \equiv \sigma_0^{(t^*)} = \sigma_0^{(12)} \approx 0.11$. In practice, one can start with the value $\sigma_0^{(6)} = S/\sqrt{2^6}$, based on the information given by Shepard's surface.

Now, exploiting the particular cell structure of the domain D constructed in the previous step, we evaluate $F(P_k;\mathcal{N}_{P_k})$ in (1) and compute $\sigma(P_k)$ in (2). We find 345 barycentres (belonging to the set $\mathcal{A}(f;S_n)$) and 115 barycentres of barycentres (belonging to the set $\mathcal{B}(f;S_n)$), that are showed in Figure 4 (left) and (right) respectively.

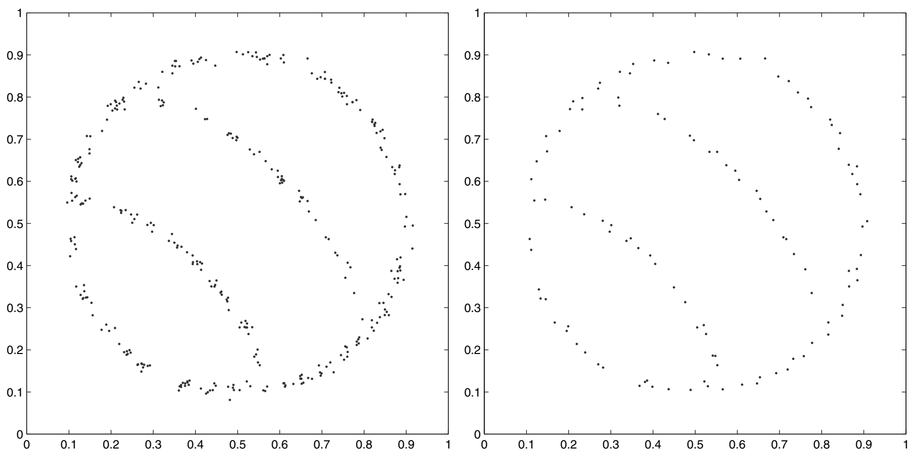

Figure 4: Set of barycentres $\mathcal{A}(f;S_n)$ (left) and set of barycentres of barycentres $\mathcal{B}(f;S_n)$ (right) of f_1.

Applying the NNSP to the set $\mathcal{B}(f;S_n)$ we obtain the set $\mathcal{C}(f;S_n)$ and from this the preliminary, low accurate, polygonal lines. Moreover, $\mathcal{C}(f;S_n)$ provides information on how to split fault lines into different branches. This phase is not completely automatic, but a simple user interaction is required: it is to examine the polygonal lines (see Figure 5 (left)).

Finally, after applying the iterative refinement, we can obtain an accurate approximation of fault lines by one of the proposed methods. In general, if we use the polygonal line method in this phase, three refinements are needed; otherwise, least squares and best l_∞ approximation methods, that hold in themselves the smoothing property, require only one refinement. Figure 5 (right) presents the fault approximation obtained by the polygonal line method, whereas the least squares and best l_∞ approximation methods are showed in Figure 6 (left) and (right), respectively.

In a similar way, we can handle any case, in which the unknown surface is almost constant in $D \setminus \Gamma$ with one or more faults, and with intersections or bifurcations of faults. Functions of this type are studied by various authors

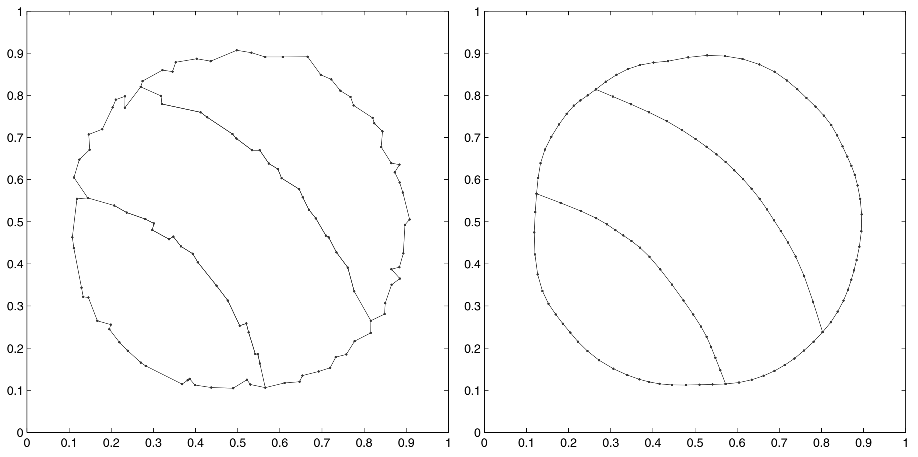

Figure 5: Initial approximation of the fault lines of f_1, obtained by using NNSP and polygonal lines (left), and accurate approximation, obtained by polygonal line method with three iterations (right).

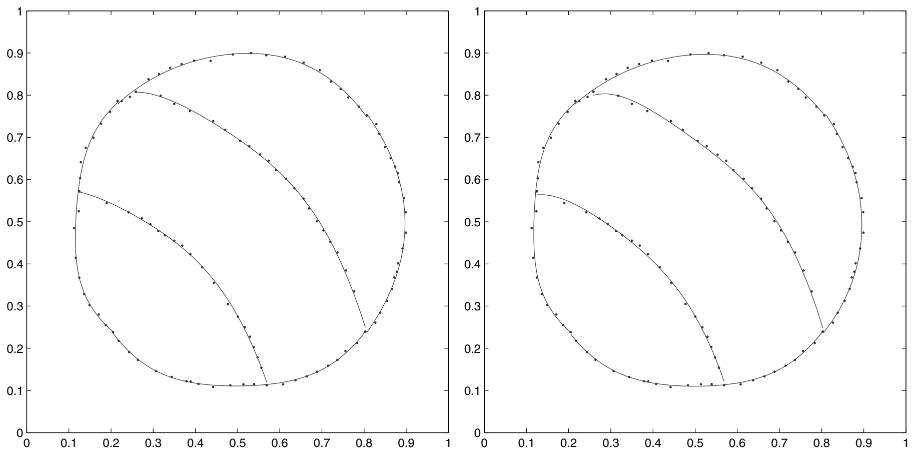

Figure 6: Accurate approximation of fault lines obtained by least squares (left) and best l_∞ approximation methods with one iteration (right) of f_1.

with different techniques and methodologies [19, 21, 8, 3, 17, 9, 12, 23, 18].

Test function 2

As second test, we consider the function (see [5])

$$f_2(x,y) = \begin{cases} g(x,y), & \text{if } \ y < \min(1/3x, 0.25), \\ h(x,y), & \text{if } \ \min(1/3x, 0.25) \leq y < 2x - 0.5, \\ 6, & \text{otherwise,} \end{cases}$$

with

$$\begin{aligned} g(x,y) &= 1 + 1.4375\left(2-(x-1)^2\right)\exp(5\,(1-x)\,(4y-1)), \\ h(x,y) &= 1 + \left(2-(x-1)^2\right)\left(2-(y-1)^2\right), \end{aligned}$$

and discontinuity set

$$\Gamma = \Gamma_1 \cup \Gamma_2 = \left\{ \left(t, \frac{1}{3}t\right) : 0 \leq t \leq 0.75 \right\} \cup \left\{ \left(t, 2t - \frac{1}{2}\right) : 0.3 \leq t \leq 0.75 \right\}.$$

This function gives a discontinuous surface with two faults, which bifurcate at the point (3/10,1/10). The fault line Γ_1 has a jump size that vanishes when x tends to 0.75, while Γ_2 presents a variable jump size on all the domain D. The surface is smooth on $D \setminus \Gamma$, but shows different characteristics, because a part of the function is constant, another is almost constant, and the remainder changes enough quickly. It follows that this surface collects some features, which are difficult to be represented, namely, bifurcated fault lines, rapid surface variations, and changing or vanishing jump sizes.

In [19], Parra, López de Silanes and Torrens evaluate the function f_2 at 10000 points randomly distributed on the domain and then apply their detection method, which determines fault lines using a divergence property of sequences measuring gradient near discontinuities (see also [18] for a different characterization of jump discontinuities and the vertical fault detection).

In the following, we test our method focusing on the case of $n = 2000$ scattered data, but we also report errors obtained by $n = 4000$, 10000 (see Table 1 and Table 2).

In Figure 7, we present Shepard's surface (left) and the corresponding contour lines (right). Such preliminary information highlights the function characteristics, locating not only the two fault lines, but also the presence of rapid variations of the surface.

Thus, since the maximum jump size is $S \approx 4.98$, the procedure finds the threshold value $\sigma_0^{(t)}$ after 9 iterations, so that $\sigma_0 \equiv \sigma_0^{(9)} \approx 0.22$. This value

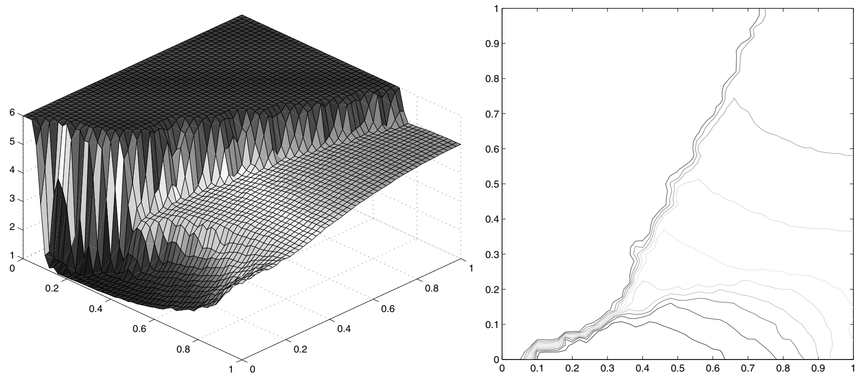

Figure 7: Shepard's surface (left) and contour lines (right) of f_2.

leads, first, to obtain the set $\mathcal{A}(f;S_n)$ consisting of 81 barycentres, and then the set $\mathcal{B}(f;S_n)$ containing 41 points (see Figure 8). To shorten, one can start with $\sigma_0^{(4)} = S/\sqrt{2^4}$.

The use of the NNSP generates the ordered set $\mathcal{C}(f;S_n)$, that allows the construction of polygonal lines connecting each point of $\mathcal{C}(f;S_n)$ with its following by straight line segments.

Now, subdividing the set $\mathcal{C}(f;S_n)$ into two subsets $\mathcal{C}_j(f;S_n)$, $j = 1,2$, we obtain an initial approximation of the fault lines, in Figure 9 (left). Then, we apply the iterative refinement and get more accurate approximation curves, as showed in Figure 9 (right) and Figure 10.

<u>Test function 3</u>

Finally, let us consider the discontinuity set Γ, which consists in the polygonal line of vertices $(0,0.2)$, $(0.2,0.2)$, $(0.35,0.225)$, $(0.42,0.3)$, $(0.48,0.4)$, $(0.49,0.53)$, $(0.5,0.65)$, $(0.65,0.725)$, $(0.8,0.75)$, $(1,0.8)$, and let f be the function defined by

$$f_3(x,y) = \frac{1}{\left[\left(3x-\frac{7}{2}\right)^2 + \left(3y-\frac{7}{2}\right)^2\right]} + g(x,y),$$

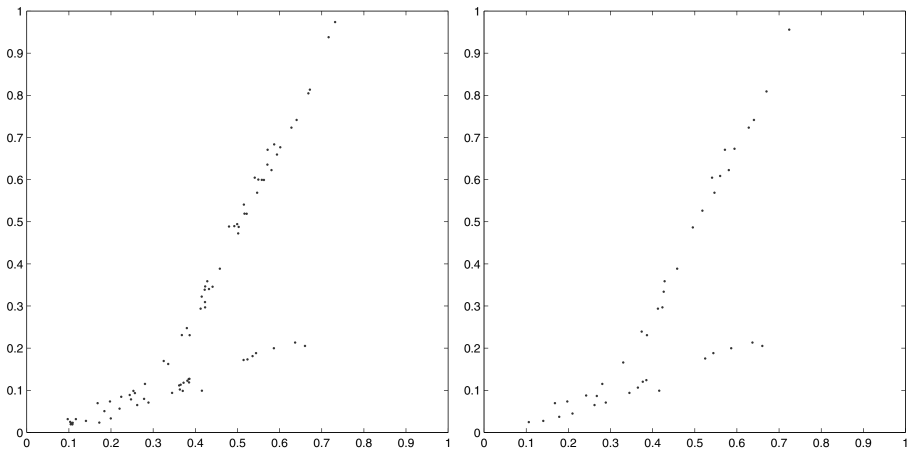
Figure 8: Set of barycentres $\mathcal{A}(f;S_n)$ (left) and set of barycentres of barycentres $\mathcal{B}(f;S_n)$ (right) of f_2.

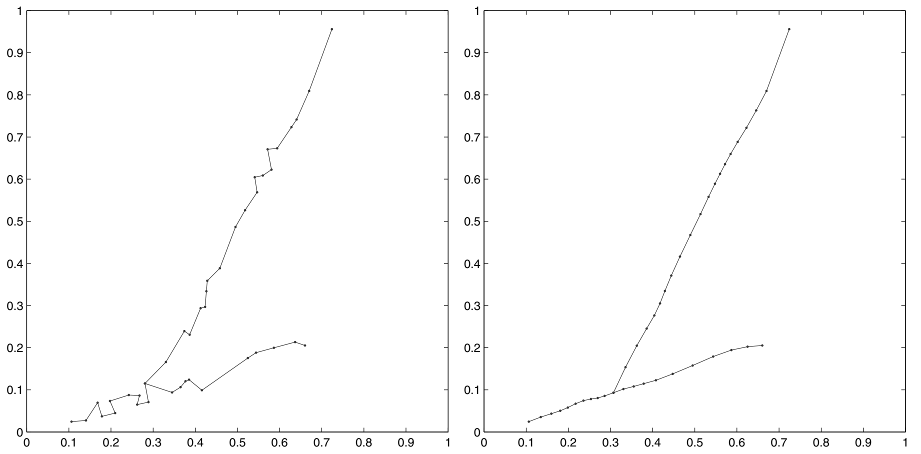
Figure 9: Initial approximation of the fault lines of f_2, obtained by using NNSP and polygonal lines (left), and accurate approximation, obtained by polygonal line method with three iterations (right).

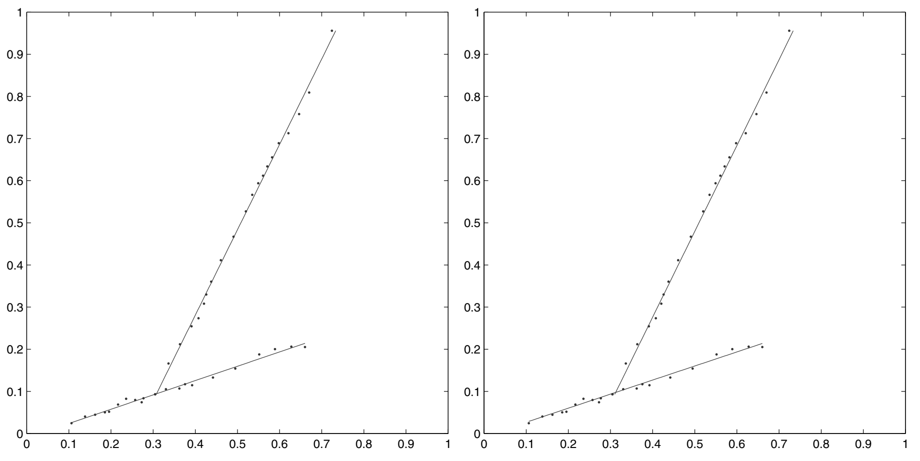

Figure 10: Accurate approximation of fault lines obtained by least squares (left) and best l_∞ approximation methods with one iteration (right) of f_2.

where g denotes the function given by

$$g(x,y) = \begin{cases} 0.35 + \exp\left[-\left(3x - \frac{3}{2}\right)^2 - \left(3y - \frac{3}{2}\right)^2\right], & \text{if } (x,y) \in \omega, \\ 0, & \text{otherwise,} \end{cases}$$

and ω is the part of the domain D located above the fault line (according to the y-axis direction). This test function is considered by Gout et al. [15].

Using a scattered data sample of dimension $n = 2000$, we test our method proposed in the paper and obtain the results as follows:

(a) From Shepard's formula (3) and the relative contour lines (see Figure 11), we have basic information on the behaviour and the orientation of the fault line.

(b) After obtaining the maximum surface variation $S \approx 2.06$, we find the threshold value $\sigma_0 \equiv \sigma_0^{(6)} \approx 0.26$. The detection phase identifies 36 points belonging to the set $\mathcal{A}(f;S_n)$, while the set $\mathcal{B}(f;S_n)$ contains 22 points, represented in Figure 12 (left).

(c) Constructing the set $\mathcal{C}(f;S_n)$, we approximate the fault line by the polygonal line method applying once the iterative refinement as showed in Figure 12 (right).

Analogous results are obtained considering the least squares and best l_∞ approximation methods.

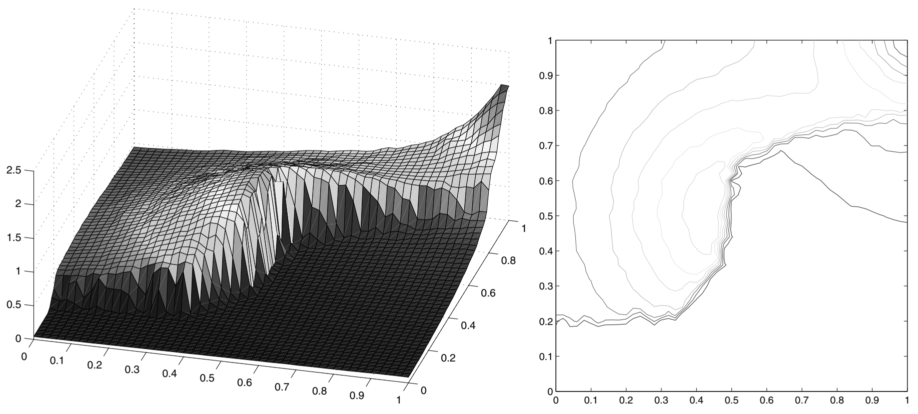

Figure 11: Shepard's surface (left) and contour lines (right) of f_3.

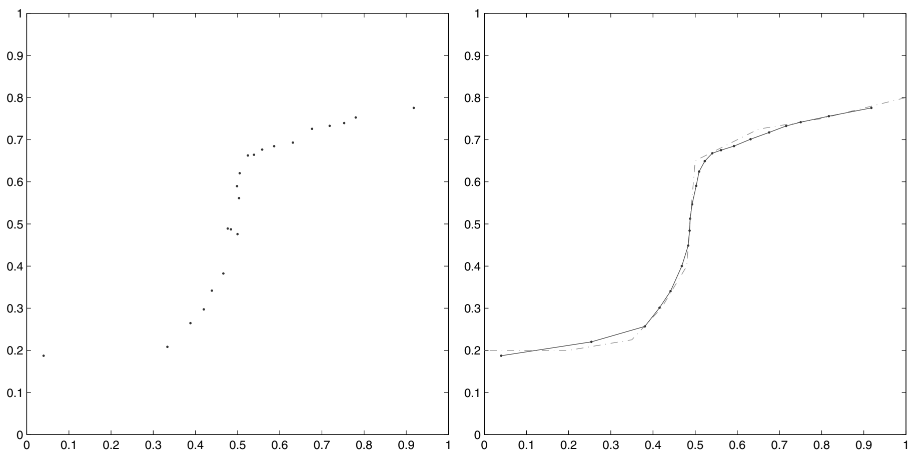

Figure 12: Set of barycentres of barycentres $\mathcal{B}(f;S_n)$ (left) and comparison between the fault line of f_3 (dash-dot line) and the fault line approximation obtained by polygonal line method with one iteration (solid line) (right).

Functions with features similar to those described in the last two tests can be found in [19, 22, 17, 4]. Furthermore, recovering functions with discontinuities sampled on cartesian grid are studied in [10, 11].

For the first two test functions, we compute the maximum absolute error (MAE) and the root mean square error (RMSE), evaluating each fault line at 100 equispaced points in $0 \leq x \leq 1$. If a fault line is parallel or nearly parallel to $y-$axis, it is convenient first to rotate the coordinate axes and then to compute the errors. The results are reported in Table 1 and Table 2. For simplicity, we compare only least squares and best l_∞ approximation curves. In fact, also a polygonal line could be expressed analytically as a linear piecewise polynomial, but there would be too many pieces.

f/n	2000	4000	10000
f_1	$6.83495\text{E}-2$	$2.83227\text{E}-2$	$2.04529\text{E}-2$
	$8.57487\text{E}-3$	$3.63514\text{E}-3$	$2.70869\text{E}-3$
f_2	$1.06565\text{E}-2$	$7.34524\text{E}-3$	$4.41158\text{E}-3$
	$7.97225\text{E}-3$	$5.40742\text{E}-3$	$3.79520\text{E}-3$

Table 1: MAEs and RMSEs resulting from least squares method.

f/n	2000	4000	10000
f_1	$6.89141\text{E}-2$	$2.63607\text{E}-2$	$2.08777\text{E}-2$
	$8.91904\text{E}-3$	$4.57793\text{E}-3$	$3.47217\text{E}-3$
f_2	$1.34226\text{E}-2$	$1.06295\text{E}-2$	$9.24255\text{E}-3$
	$8.33712\text{E}-3$	$5.65426\text{E}-3$	$5.68766\text{E}-3$

Table 2: MAEs and RMSEs resulting from best l_∞ approximation method.

6. Conclusions

In this paper, we have presented and analyzed a method for the detection and accurate approximation of fault lines in an unknown discontinuous surface from scattered data.

The algorithm of detection, under a reasonable choice of the threshold parameter σ_0, yields the set of barycentres $\mathcal{A}(f;S_n)$. This phase allows to understand the form and the number of faults. The approach is similar but alternative to that proposed in [16], where it is used a local approximation scheme based on thin plate splines combined with a triangulation method. Then, from the set $\mathcal{A}(f;S_n)$ we obtain the set of barycentres of barycentres $\mathcal{B}(f;S_n)$, and its ordered version $\mathcal{C}(f;S_n)$. This last set is necessary to handle complex situations, such as several faults, close type faults and intersections or bifurcations of faults. In these cases $\mathcal{C}(f;S_n)$ is subdivided into a number of subsets greater or equal to the number of discontinuities. Moreover, the set $\mathcal{C}(f;S_n)$ is basic to apply the proposed approximation methods, namely, polygonal line, least squares, and best l_∞ approximation methods.

For automaticity and simplicity the polygonal line method turns out to be more suitable in almost all applications, even if least squares or best l_∞ approximations produce good or even better results. Moreover, the iterative refinement allows us to construct smooth, planar curves rather than polygonal curves.

When the fault line is a multivalued function, applying least squares and best l_∞ approximations requires to subdivide the data. This situation can create small errors at points in which two pieces of curve join, but the introduction of the iterative refinement reduces considerably the occurrence of these errors of connection. Nevertheless, the decision of how partitioning the domain can hide several difficulties (see [12]).

The application of our method pointed out at least two problems. If the considered surface show very rapid variations outside the faults, then the detection algorithm may identify false fault points. Another drawback occurs when the jump size varies fastly or vanishes; here the optimal choice of a global parameter σ_0 is difficult and sometimes it is impossible to detect correctly the fault points. To solve these problems, we could split up the domain in a uniform grid and find a threshold parameter for each grid cell. This approach should allow to detect only the actual points of discontinuity. However, it is yet an open problem and we are considering it.

References

[1] G. ALLASIA, *Simultaneous interpolation and approximation by a class of multivariate positive operators*, Numer. Algorithms **34** (2003), 147–158.

[2] G. ALLASIA, *Recursive and parallel algorithms for approximating surface data on a family of lines or curves*, in P. CIARLINI et al. (eds.), *A*dvanced mathematical and computational tools in Metrology VI, World Sci., N. J., 2004, pp. 137–148.

[3] G. ALLASIA, R. BESENGHI and A. DE ROSSI, *A scattered data approximation scheme for the detection of fault lines*, in T. LYCHE and L.L. SCHUMAKER (eds.), *Mathematical methods for curves and surfaces: Oslo 2000*, Vanderbilt Univ. Press., Nashville TN, 2001, 25–34.

[4] R. ARCANGÉLI, M.C. *López de Silanes*, and J.J. T*orrens*, *Multidimensional minimizing splines. Theory and applications.* Kluwer, Boston, MA, 2004.

[5] E. ARGE and M. FLOATER, *Approximating scattered data with discontinuities*, Numer. Algorithms **8** (1994), 149–166.

[6] I. BARRODALE and A. Y*oung*, *Algorithms for best L_1 and L_∞ linear approximation on a discrete set*, Numer. Math. **8** (1966), 295–306.

[7] J.L. BENTLEY and J.H. FRIEDMAN, *Data structures for range searching*, *Comput. Surveys* **11** (1979), 397–409.

[8] R. BESENGHI and G. ALLASIA, *Scattered data near-interpolation with application to discontinuous surfaces*, in A. COHEN, C. RABUT, and L.L. SCHUMAKER (eds.), *Curves and surfaces fitting: Saint-Malo 1999*, Vanderbilt Univ. Press., Nashville TN, 2000, pp. 75–84.

[9] R. BESENGHI, M. COSTANZO and A. DE ROSSI, *A parallel algorithm for modelling faulted surfaces from scattered data*, Int. J. Comput. Numer. Anal. Appl. **3** (2003), 419–438.

[10] M. BOZZINI and L. LENARDUZZI, *Recovering a function with discontinuities from correlated data*, in F. FONTANELLA, K. JETTER and P.-J. LAURENT (eds.), *Advanced topics in multivariate approximation*, World Scientific, Singapore, 1996, pp. 1–16.

[11] M. BOZZINI and M. ROSSINI, *Approximation surfaces with discontinuities*, Math. Comput. Modelling **31** (2000), 193–213.

[12] A. CRAMPTON and J.C. MASON, *Detecting and approximating fault lines from randomly scattered data*, Numer. Algorithms **39** (2005), 115–130.

[13] N.P. FREMMING O. HJELLE and C. TARROU, *Surface modelling from scattered geological data*, in M. DÆHLEM and A. TVEITO (eds.), *Numerical methods and software tools in industrial mathematics*, Birkhäuser, Boston, 1997, pp. 301–315.

[14] C. GOUT and C. LE GUYADER, *Segmentation of complex geophysical structures with well data*, Comput. Geosci. **10** (2006), 361–372.

[15] C. GOUT, C. LE GUYADER, L. ROMANI and A.G. SAINT-GUIRONS, *Approximation of surfaces with fault(s) and/or rapidly varying data, using a segmentation process, D^m-splines and the finite element method*, Numer. Algorithms **48** (2008), 67–92.

[16] T. GUTZMER and A. ISKE, *Detection of discontinuities in scattered data approximation*, Numer. Algorithms **16** (1997), 155–170.

[17] M.C. L*ópez de Silanes*, M.C. PARRA and J.J. TORRENS, *Vertical and oblique fault detection in explicit surfaces*, J. Comput. Appl. Math. **140** (2002), 559–585.

[18] M.C. LÓPEZ DE SILANES, M.C. PARRA and J.J. TORRENS, *On a new characterization of finite jump discontinuities and its application to vertical fault detection*, Math. Comput. Simul. **77** (2008), 247–256.

[19] M.C. PARRA, M.C. LÓPEZ DE SILANES and J.J. TORRENS, *Vertical fault detection from scattered data*, J. Comput. Appl. Math. **73** (1996), 225–239.

[20] R.J. RENKA, *Multivariate interpolation of large sets of scattered data*, ACM Trans. Mathematical Software **14** (2) (1988), 139–148.

[21] M. ROSSINI, *Irregularity detection from noisy data in one and two dimensions*, Numer. Algorithms **16** (1997), 283–301.

[22] M. ROSSINI, *2D-discontinuity detection from scattered data*, Computing **61** (1998), 215–234.

[23] M. ROSSINI, *Detecting discontinuities in two-dimensional signals sampled on a grid*, J. Numer. Anal. Ind. Appl. Math. **1** (2007), 1–13.

[24] J. SPRINGER, *Modeling of geological surfaces using finite elements*, in P.-J. LAURENT, A. LE MÉHAUTÉ and L.L. SCHUMAKER (eds), Wavelets, images and surfaces fitting, A K Peters, Wellesley, Mass., 1994, pp. 467–474.

Testo definitivo pervenuto in redazione il 7 gennaio 2009.