COMMUNICATIONS IN COMPUTATIONAL PHYSICS
Vol. 5, No. 2-4, pp. 694-711

Commun. Comput. Phys.
February 2009

A High Order Method for Determining the Edges in the Gradient of a Function

Rishu Saxena*, Anne Gelb and Hans Mittelmann

Department of Mathematics and Statistics, Arizona State University, Tempe, Arizona 85287, USA.

Received 1 October 2007; Accepted (in revised version) 30 November 2007

Available online 1 August 2008

Abstract. Detection of edges in piecewise smooth functions is important in many applications. Higher order reconstruction algorithms in image processing and post processing of numerical solutions to partial differential equations require the identification of smooth domains, creating the need for algorithms that will accurately identify discontinuities in a given function as well as those in its gradient. This work expands the use of the polynomial annihilation edge detector, (Archibald, Gelb and Yoon, 2005), to locate discontinuities in the gradient given irregularly sampled point values of a continuous function. The idea is to preprocess the given data by calculating the derivative, and then to use the polynomial annihilation edge detector to locate the jumps in the derivative. We compare our results to other recently developed methods.

AMS subject classifications: 41A10, 41A63, 41A58, 65D99

Key words: Multivariate edge detection, derivative discontinuities, piecewise smooth functions, polynomial annihilation, non-uniform grids.

1 Introduction

Edge detection is fundamentally important in image reconstruction, feature extraction and several other applications. While many edge detection algorithms are available, [2, 5, 8, 10, 11, 13, 14], less attention has traditionally been paid to determining edges in the gradients of functions. However, such information can be very useful. For instance, solutions to partial differential equations that arise in gas dynamics and acoustic problems in heterogeneous media often have derivative discontinuities. Locating them can help high order post-processing of such numerical solutions. Their locations may also

*Corresponding author. *Email addresses:* `saxena@mathpost.asu.edu` (R. Saxena), `ag@math.asu.edu` (A. Gelb), `mittelmann@asu.edu` (H. Mittelmann)

aid in determining domain decompositions that avoid shocks and contact discontinuities. This paper addresses the problem of determining discontinuities in the gradients of functions.

Most edge detection methods are either search based, [5, 14], or zero-crossing based, [8]. Both types look directly for jump discontinuities. In contrast, the polynomial annihilation edge detector described in [2] determines intervals of smoothness from which the location of discontinuities in the function can be accurately identified. The method has several advantages over already existing methods, the most important being that it is applicable to multi-dimensional scattered data. Due to its variable order construction, it captures jumps located as close as one pixel apart as well as distinguishes them from steep gradients. The method is robust and is simple to implement numerically. Finally, it has limited dependence on outside thresholding present in many edge detectors.

An attempt to extend the polynomial annihilation edge detector to detect derivative discontinuities was made in [3]. However, the extension requires the use of several stencils, making it less efficient and robust compared to the original method in [2]. Furthermore, the method so far can be implemented in multi-dimensions only by using a dimension by dimension approach. This motivates us to develop a gradient edge detection method that can also be implemented on scattered data in higher dimensions.

The technique proposed in this paper detects jumps in the derivative of the given data using a two-pass approach. We first preprocess the data by numerically approximating the derivative. Then we use the polynomial annihilation edge detector to locate discontinuities in the approximate derivative. Our technique offers significant advantages over existing methods in that it is applicable to data on scattered grids and is multi-dimensional by design.

This paper is organized as follows: Section 2 reviews the polynomial annihilation edge detector and discusses its extension to capturing the derivative jump discontinuities. Section 3 explains our preprocessing algorithm in one and two dimensions followed by the edge detection procedure. Concluding remarks are given in Section 4.

2 Polynomial annihilation edge detection

2.1 Locating jump discontinuities in one dimension

The polynomial annihilation edge detector in [2] differs from other commonly used edge detectors chiefly in the fact that it looks for intervals of smoothness rather than looking for jump locations, [5,14]. In smooth intervals, the method annihilates the first m terms of the function's Taylor expansion. In non-smooth regions, the method essentially estimates the finite projection of the derivative – thereby locating the jump discontinuity.

To see how the edge detector works, let us consider a piecewise continuous function $f:[a,b]\to R$ known only on the set of discrete points $S=\{x_1,x_2,\cdots,x_N\}\subset[a,b]$. Assume that f has well defined one sided limits, $f(x\pm)$, at any point x in the domain. We denote by J the set of the points of discontinuity of f, that is, $J=\{\xi:a<\xi<b\}$, where ξ is a point

of a jump discontinuity in the function. The local jump function is defined as

$$[f](x)=f(x+)-f(x-)=\begin{cases} 0, & \text{if } x\neq\xi, \\ [f](\xi), & \text{if } x=\xi. \end{cases} \tag{2.1}$$

Hence if f is continuous at x, the jump function $[f](x)=0$; if x is a point of discontinuity of f, then $[f](x)$ is equal to the signed magnitude of the jump value.

We seek an approximation to $[f](x)$ that converges rapidly to zero away from the jump discontinuities. To this end, let x be a point inside the domain, i.e., $x\in(a,b)$. For a given positive integer m, we choose a local stencil

$$S_x=\{x_j|x_j\in S\}=\{x_1,\cdots,x_{m+1}\} \tag{2.2}$$

of nearest $m+1$ grid points around x. We denote by Π_m the space of all polynomials of degree $\leq m$ with $dim(\Pi_m)=m+1$. Let p_i, $i=1,\cdots,m+1$, be any basis for this space of polynomials. The polynomial annihilation edge detector introduced in [2] is given by

$$L_m f(x)=\frac{1}{q_m(x)}\sum_{x_j\in S_x} c_j(x)f(x_j)\ . \tag{2.3}$$

In one dimension, the reconstruction points are typically chosen as the mid-points of the grid intervals (cells). The coefficients $c_j(x)$ are determined by solving the following linear system that results from annihilation of polynomials of degree up to $m-1$,

$$\sum_{x_j\in S_x} c_j(x)p_i(x_j)=p_i^{(m)}(x),\ \forall\, i=1,\cdots,m+1, \tag{2.4}$$

w here $p_i^{(m)}(x)$ denotes the mth derivative of $p_i(x)$. Notice here that the solution to (2.4) exists and is unique. Next, we also define

$$S_x^+=\{x_j\in S_x|x_j\geq x\} \text{ and } S_x^-=S_x\setminus S_x^+. \tag{2.5}$$

The normalization factor in (2.3) is then given by

$$q_m(x)=\sum_{x_j\in S_x^+} c_j(x), \tag{2.6}$$

and ensures that $L_m f(x)$ has correct magnitude at the jump discontinuities. Note that (2.6) is non-zero by design. In [2] it was shown that the polynomial annihilation edge detector satisfies the desired property, viz., in regions where f is smooth, (2.3) converges to zero with a rate depending upon m and on the local smoothness of f. Specifically, if the maximum separation $h(x)$ is defined as

$$h(x)=\max\{|x_i-x_{i-1}|:x_{i-1},x_i\in S_x\}, \tag{2.7}$$

then

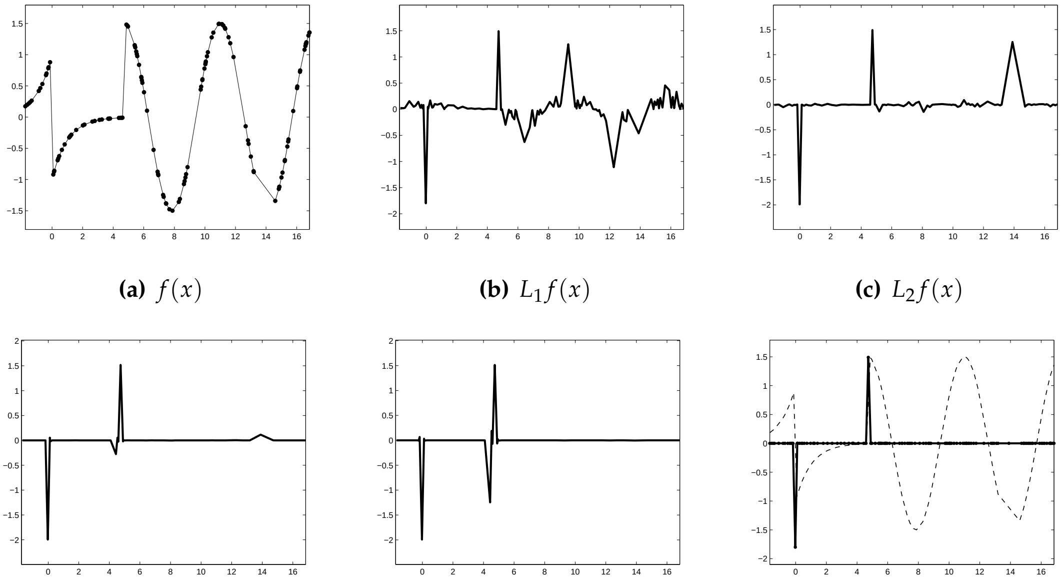
$$L_m f(x) = \begin{cases} [f](\xi) + \mathcal{O}(h(x)), & if\ x_{j-1} \le \xi, x \le x_j, \\ \mathcal{O}(h^{min(m,k)}(x)), & if\ f \in C^k(I_x)\ for\ k > 0. \end{cases} \tag{2.8}$$

Here, I_x is the smallest closed interval such that $S_x \subset I_x$. The proof of convergence is given in [2]. The following example demonstrates the performance of (2.3).

Example 2.1. Consider the function on $[-2,6\pi]$

$$f(x) = \begin{cases} e^x, & x < 0, \\ -e^{-x}, & 0 \le x < \frac{3\pi}{2}, \\ -1.5\sin(x), & \frac{3\pi}{2} \le x \le 6\pi. \end{cases} \tag{2.9}$$

The function in Example 2.1 has jump discontinuities at $x=0$ and $3\pi/2$ and is continuous everywhere else in the domain. We apply (2.3) on a randomly distributed grid. Fig. 1(a) displays the wide gaps located near and about $x=9$ and $x=12$. Although the function is smooth there, these gaps can be easily mistaken by a low order edge detector as jump discontinuities in the data. The results of applying the polynomial annihilation edge detector (2.3) for various orders m are shown in Fig. 1(b)-(e).



(a) $f(x)$ **(b)** $L_1 f(x)$ **(c)** $L_2 f(x)$

(d) $L_4 f(x)$ **(e)** $L_5 f(x)$ **(f)** ***Minmod edge detection***

Figure 1: Example 2.1 (a) underlying function; (b)-(e) Jump function approximation of Example 2.1 using various orders m; (f) Minmod results, (2.10).

Fig. 1 demonstrates the importance of the order m in (2.3). Small order m might cause misidentification of a steep gradient as an edge (due to low resolution). On the other hand, oscillations occur in the vicinity of the discontinuities when m is large. To prevent inaccuracies due to either reason, we apply the *minmod* function, which is typically used to reduce oscillations in flux limiters when solving numerical conservation laws. It was introduced in the context of edge detection in [4, 6] and was used in [2] to enhance the performance of the polynomial annihilation method. Specifically, we apply

$$MM(L_mf(x)) = \begin{cases} \min_{m\in\mathcal{M}} L_mf(x), & \text{if } L_mf(x) > 0 \ \forall\, m\in\mathcal{M}, \\ \max_{m\in\mathcal{M}} L_mf(x), & \text{if } L_mf(x) < 0 \ \forall\, m\in\mathcal{M}, \\ 0, & \text{otherwise}, \end{cases} \tag{2.10}$$

where $\mathcal{M}\subset N$. As shown in Fig. 1(f) for $\mathcal{M}=\{1,2,3,4,5\}$, the minmod function controls the oscillations while still maintaining high order of convergence away from the jumps.

2.2 Locating jump discontinuities in two dimensions

Let $f:\Omega\to\mathbb{R}$ be a piecewise continuous function on a bounded domain $\Omega\subset R^2$ known only on the set of discrete points $S\subset\Omega$. Analogous to the one dimensional case, we let J denote the set of discontinuity points in f. The jump discontinuity at $x=\xi$ is now defined by its enclosed points. Following [2], we use the Delaunay triangulation of S to define the enclosed points, as obtained by

$$\tau_S = \{T_j | T_j \doteq \{x_1^j, x_2^j, x_3^j\} \subset S \text{ for } j=1,\cdots N_T\}, \tag{2.11}$$

where N_T is the number of triangles formed by the triangulation. The set of elementary triangles are non-overlapping and are such that for each triangle, its circumcircle does not contain any other grid point. The triangulation maximizes the minimum angle of all the angles of the triangles. The barycenters of these triangles form the reconstruction points for our polynomial annihilation edge detector. We define a local stencil S_x of

$$m_2 = \left(\begin{array}{c} m+2 \\ 2 \end{array} \right)$$

points around the barycenter x such that $S_x = T_j \cup S_{T_j}$. Here, $T_j\in\tau_S$ is such that $x\in T_j$ and S_{T_j} is the set of m_2-3 closest points to x in the set $S\backslash T_j$. The jump function approximation of order m is again defined as in (2.3). As in the one dimensional case, the coefficients $c_j(x)$ depend upon the local set S_x and are determined by solving the linear system

$$\sum_{x_j\in S_x} c_j(x) p_i(x_j) = \sum_{|\alpha|_1=m} p_i^{(\alpha)}(x), \quad i=1,\cdots,m_2, \quad \alpha\in Z_+^2 . \tag{2.12}$$

The polynomials to be annihilated, $p_i(x)$, $i=1,\cdots,m_2$, are chosen such that they form a basis of Π_m, the space of polynomials of degree up to m. The scaling factor q_m for the two-dimensional case is defined as

$$q_m(x) \doteq \sum_{x_j\in\mathcal{P}_x} c_j(x) ,$$

where $\mathcal{P}_x$ is a suitable subset of S_x such that $q_m(x)\neq 0$ (see [2] for the construction of $\mathcal{P}_x$). In the two-dimensional case the jump magnitude may vary along different paths through the discontinuity. Hence defining the magnitude of the jump is no longer meaningful as it was in one dimension. However, with q_m thus defined, the edge detector $L_m f(x)$ remains uniformly bounded.

The rapid convergence of $L_m f(x)$ to zero away from the discontinuities can be quantified using the spatial density

$$h(x)=\max_{x\in K_{S_x}}\min_{x_j\in S_x}|x-x_j|,$$

where $\mathcal{K}_{S_x}$ is the convex hull of S_x. As proved in [2], if $f\in C^k(K_{S_x})$, for some $k>0$, then

$$L_m f(x)=\mathcal{O}(h^{min(k,m)}(x)).$$

As in the one dimensional case, the edge detector suffers from oscillations in the vicinity of discontinuities. Once again we utilize the (two-dimensional) minmod function, (2.10). Enhancing the edge detector with the minmod function increases the area of convergence for the edge detector away from the discontinuities and helps pinpoint the location of its discontinuities.

2.3 Polynomial annihilation derivative detector

In [3] the polynomial annihilation edge detector was extended to detect discontinuities in the derivative of functions. The underlying formulation is the same, however, we now consider a one dimensional piecewise smooth function $f\in C^{\gamma-1}$, $\gamma\in N=\{1,2,\cdots\}$, known only on the set of discrete points, $S=\{x_1,x_2,\cdots,x_N\}\subset(a,b)$. For ease of presentation, we assume that $f(x)$ and all its derivatives up to $f^{(\gamma-1)}(x)$ are continuous in $[a,b]$ and the jump discontinuity first appears in $f^{(\gamma)}(x)$. [†] We also assume that at any point x in the domain, $f^{(\gamma)}(x)$ has well-defined one sided limits. We denote by J_γ the set of jump discontinuities in the γth derivative, that is,

$$J_\gamma=\{\xi|f^{(\gamma)}(x+)\neq f^{(\gamma)}(x-)\}, \tag{2.13}$$

where $f^{(\gamma)}(x+)$ and $f^{(\gamma)}(x-)$ are the right and left side limits of $f^{(\gamma)}(x)$ at the point $x\in[a,b]$. The local jump function for the γth derivative is defined as

$$[f^{(\gamma)}](x)=f^{(\gamma)}(x+)-f^{(\gamma)}(x-).$$

As before, for $x\notin J_\gamma$, $[f^{(\gamma)}](x)=0$. When $x=\xi\in J_\gamma$,

$$[f^{(\gamma)}](x)=[f^{(\gamma)}](\xi).$$

[†] A multi-pass approach used in [3] enables the method to determine jump discontinuities in subsequent derivatives.

Next, we consider any point $x \in (a,b)$ and let $m > \gamma$ be a positive integer. Following the notations in Section 2.1, let S_x denote a local set of $(m+\gamma+2)$ points around x with S_x^+ and S_x^- defined as in (2.5). For reasons stated later, the stencil S_x must be carefully chosen so that

$$\min(\#S_x^+,\#S_x^-) > \gamma.$$

The $\gamma-$derivative edge detector of order m for stencil S_x is then defined as

$$L_{m,\gamma,S_x} f(x) = \frac{1}{q_{m,\gamma}(x)} \sum_{x_j \in S_x} c_j(x) f(x_j). \tag{2.14}$$

We use the notation $L_{m,\gamma,S_x} f(x)$ since the method will require the use of several stencils. Again c_j are determined from (2.4) but are now subject to the additional constraints

$$\sum_{x_j \in S_x^+} c_j(x) p_l(x_j) = h^{-m+\gamma}(x) p_l^{(\gamma)}(x), \quad l = 1,\cdots,\gamma+1, \tag{2.15}$$

where the grid density $h(x)$ is defined in (2.7). The functions $p_l(x)$, $l = 1,\cdots,m+1$, used in (2.4) and (2.15) represent a basis for $\prod_m$, the space of polynomials up to degree m. The normalization factor $q_{m,\gamma}(x)$ is given by

$$q_{m,\gamma}(x) = \sum_{x_j \in S_x^+} c_j(x) \frac{(x_j - x)^\gamma}{\gamma!}. \tag{2.16}$$

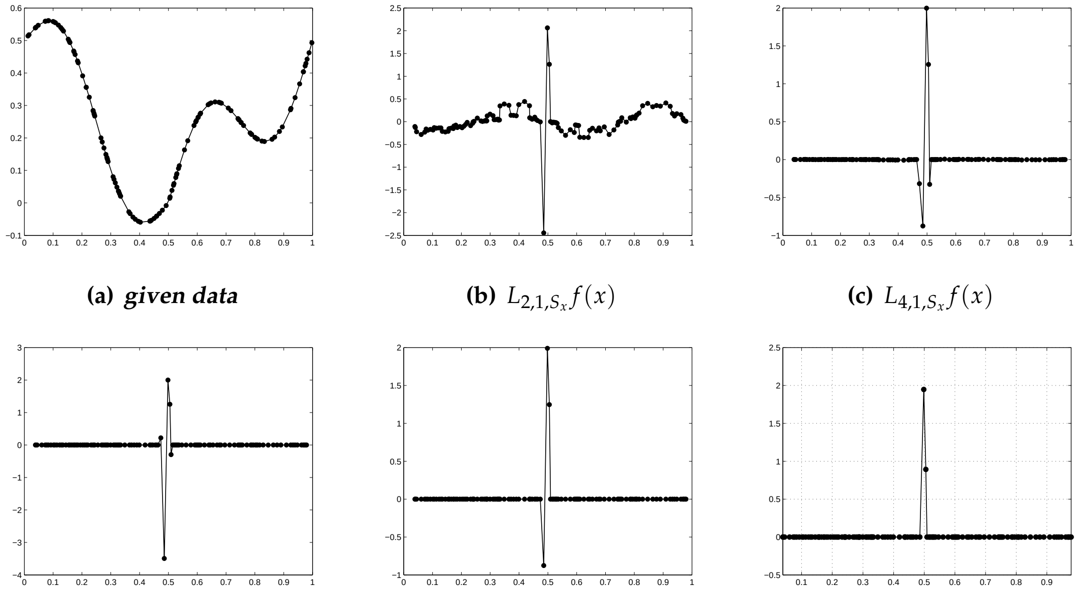
The constraints in (2.15) guarantee this normalization factor to be non-zero. The annihilation of polynomials up to degree m in (2.4) results in all the first m terms in the Taylor expansion of $f(x)$ becoming zero. When the remainder of the Taylor expansion is divided by the scaling factor $q_{m,\gamma}(x)$, a high order of convergence is achieved in the smooth regions. In [3] it was shown that (2.14) converges to $[f^\gamma](x)$. The following example illustrates the application of (2.14) when $\gamma = 1$.

Example 2.2. Let $f:[0,1] \to R$ such that

$$f(x) = \begin{cases} -(x-\frac{1}{2}) + \frac{1}{6}\sin(4\pi x), & if\ x < \frac{1}{2}, \\ (x-\frac{1}{2}) + \frac{1}{6}\sin(4\pi x), & if\ x \geq \frac{1}{2}. \end{cases} \tag{2.17}$$

Fig. 2(a) shows the data sampled on $N = 128$ irregularly distributed grid points. The first derivative, $f'(x)$, has a discontinuity at $\xi = \frac{1}{2}$, with $[f'](\xi) = 2$. The derivative edge detector, (2.14), implemented with orders $m = 2,4,5$ is displayed in Fig. 2(b)-(d).

The polynomial annihilation derivative detector, (2.14), inherits similar problems as the polynomial annihilation edge detector, (2.3). Choosing small $m > \gamma$ may lead to false jumps while choosing large m causes oscillations in the vicinity of the discontinuities.



(a) ***given data*** **(b)** $L_{2,1,S_x}f(x)$ **(c)** $L_{4,1,S_x}f(x)$

(d) $L_{5,1,S_x}f(x)$ **(e)** $MM(L_{m,1,S_x}f(x))$ **(f)** $MM(L_{m,1,S_x}f(x))$

Figure 2: Example 2.2 distributed on $N=128$ randomly distributed points. (a) Original function; (b)-(d) $L_{m,1,S_x}f(x)$ for $m=2,4$ and 5; (e) Edges detected after applying minmod limiter; (f) Edges detected after including stencil shifting.

Once again, we use the minmod limiter, (2.10). However, as shown in Fig. 2(e), the minmod limiter does not remove all oscillations because at some points near the discontinuity, all of the oscillations in L_{m,γ,S_x} are in the same direction. To resolve this problem, multiple stencils were used in [3] to compute L_{m,γ,S_x} around each point. The minmod limiter was then applied to $L_{m,\gamma,S_x^j}f(x)$, with $j=1,2$, and 3 and $m=2,\cdots,5$. The derivative jump discontinuity detected after employing stencil shifting is shown in Fig. 2(f).

Our numerical results indicate that the polynomial annihilation derivative detector is able to capture the edges in the derivative of a function. The method assumes that in any local set of size $m+\gamma+2$ points, there exists at most one derivative discontinuity. This, together with the requirement $m>\gamma$, sets a minimum resolution requirement of $2\gamma+3$ stencil points. Furthermore, since

$$\min(\#S_x^-,\#S_x^+)>\gamma,$$

the discontinuities in the derivative functions must be at least $\gamma+1$ grid points away from the domain boundaries. Finally, the method is designed only for one dimension. Derivative discontinuities in higher dimensions are detected by implementing the method direction by direction. Consequently, the method cannot be implemented when the data is given on irregularly scattered grids. Hence we are motivated to search for a technique that can be implemented on irregular grids in two dimensions as well.

3 Preprocessing by differentiation

As mentioned in the introduction, our approach here is to first approximate the piecewise smooth derivative of a continuous function and then use the (standard) polynomial annihilation method to determine its jump discontinuities. We use the ENO (Essentially Non-Oscillatory) interpolation algorithm, [1,7,9,12], to approximate the derivative.

3.1 One dimension

Recall the divided difference formulation of a function $f(x):[a,b]\rightarrow R$. The first order divided difference is defined as $f[x_i]=f(x_i)$, $\forall\ i=1,\cdots,N$. The second order divided difference is then defined as

$$f[x_{i-1},x_i]=\frac{f[x_i]-f[x_{i-1}]}{x_i-x_{i-1}}\,. \tag{3.1}$$

The kth order divided difference of $f(x)$, based on a stencil consisting of s points to the left of x_i, is recursively defined in terms of the $(k-1)$th order divided differences as

$$f[x_{i-s},\cdots,x_i,\cdots,x_{i-s+k-1}]=\frac{f[x_{i-s+1},\cdots,x_{i-s+k-1}]-f[x_{i-s},\cdots,x_{i-s+k-2}]}{x_{i-s+k-1}-x_{i-s}}. \tag{3.2}$$

The divided difference of $f(x)$ measures its smoothness, as told by [12]:

Theorem 3.1. *If $f(x)$ is smooth inside a stencil $[x_{i-s},\cdots,x_{i-s+k-1}]$, then*

$$f[x_{i-s},\cdots,x_{i-s+k-1}]=\frac{f^{(k)}(\xi)}{k!}, \tag{3.3}$$

for some point $\xi\in(x_{i-1},x_{i-s+k-1})$. If $f(x)$ is discontinuous at some point inside the stencil, then

$$f[x_{i-s},\cdots,x_{i-s+k-1}]=\mathcal{O}(\frac{1}{\Delta x}). \tag{3.4}$$

Let us consider a continuous function $f(x):[a,b]\rightarrow R$, known only on a finite number of points, $S=\{a=x_1<\cdots<x_N=b\}\subset[a,b]$, and assume that the first derivative of $f(x)$ has a discontinuity at some point in the interval (x_i,x_{i+1}). Suppose $x_j\in S$ is the point where we want to reconstruct the derivative. Interpolating the data inside any local stencil about x_j ($\{x_{j-2},\cdots,x_{j+2}\}$, for instance) is accurate if the function is smooth inside the stencil. If, however, the stencil encloses a discontinuity in one of the derivatives then, by Theorem 3.1, the accuracy of the interpolating polynomial and hence the derivative approximation does not remain valid any more. Thus, before computing the derivative value at any interior point, x_j, $j=2,\cdots,N-1$, we must first carefully choose a stencil about x_j that does not contain a derivative discontinuity. The derivative estimation process can be described as a two-step process: (i) Determine a stencil inside which the function is as smooth as possible. Then, (ii) use this stencil to compute the derivative at the point x_j.

Step 1: Stencil choosing

For an interior grid point x_j, we want to determine the smoothest kth order stencil around x_j that can be used for derivative reconstruction. Since we have assumed that the function is continuous and a potential discontinuity lies in the first derivative, we start with the following second order (three-point) stencils:

$$S_{\text{nobias}}=\{x_{j-1},x_j,x_{j+1}\},\ S_{-1}=\{x_{j-2},x_{j-1},x_j\},\ \text{and}\ S_{+1}=\{x_j,x_{j+1},x_{j+2}\}, \tag{3.5}$$

and compute the corresponding second order divided differences. The stencil that corresponds to the minimum absolute value of the divided differences is the one where the function is smoothest. We pick that stencil from the three choices as the second order stencil to be used for derivative reconstruction. Without loss of generality, let us assume that S_{-1} is chosen as the smoothest stencil. Next we want to enlarge this stencil for a still higher order accurate interpolation. This can be done by adding either x_{j+1} or x_{j-3} to S_{-1}. To decide upon which point to add, we now consider the third order divided differences $f[x_{j-3},x_{j-2},x_{j-1},x_j]$ and $f[x_{j-2},x_{j-1},x_j,x_{j+1}]$ and compare their absolute values. If

$$|f[x_{j-3},x_{j-2},x_{j-1},x_j]|<|f[x_{j-2},x_{j-1},x_j,x_{j+1}]|,$$

we add x_{j-3} to S_{-1} to obtain the third order stencil $\{x_{j-3},x_{j-2},x_{j-1},x_j\}$; otherwise, we add x_{j+1} and obtain $\{x_{j-2},x_{j-1},x_j,x_{j+1}\}$. By repeating this process $k-2$ times, adding one point at each step, we obtain a $(k+1)$ point stencil that can be used to obtain a kth order interpolation for the given data.

Step 2: Derivative reconstruction

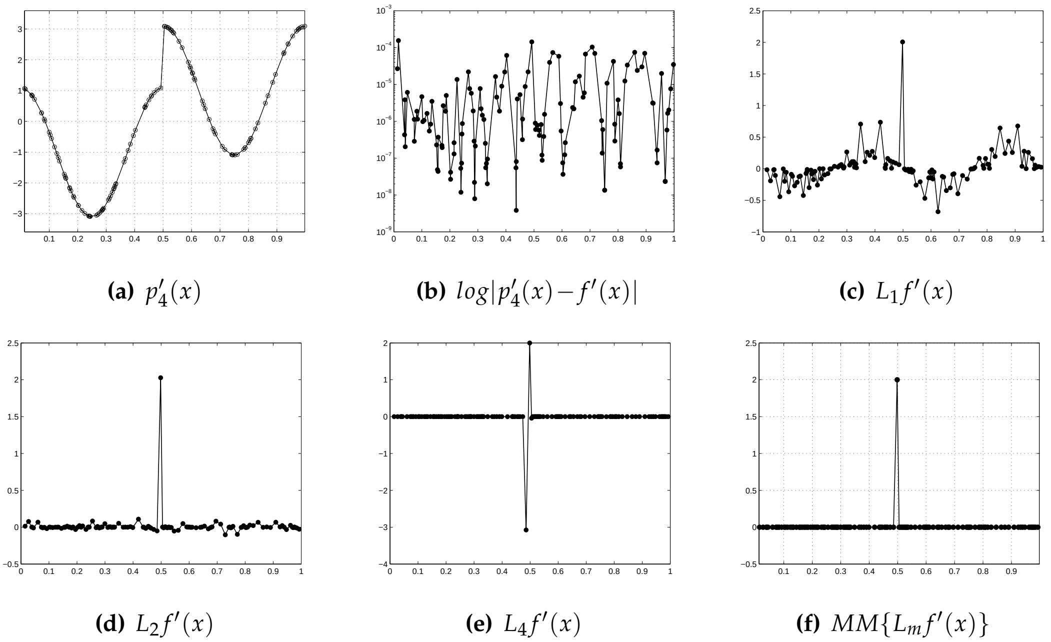
Let us assume that we are doing a $k=4$th order interpolation and $S=\{x_{j-3},x_{j-2},x_{j-1},x_j,x_{j+1}\}$ is the resulting stencil from Step 1. Then the local interpolating polynomial for the function using this stencil is given by

$$\begin{aligned}p(x)=&f[x_j]+f[x_j,x_{j-1}](x-x_j)+f[x_j,x_{j-1},x_{j-2}](x-x_j)(x-x_{j-1})\\&+f[x_j,\cdots,x_{j-3}](x-x_j)(x-x_{j-1})(x-x_{j-2})\\&+f[x_{j+1},\cdots,x_{j-3}](x-x_j)(x-x_{j-1})(x-x_{j-2})(x-x_{j-3})\ .\end{aligned} \tag{3.6}$$

The corresponding derivative at $x=x_j$ is simply

$$\begin{aligned}p'(x_j)=&f[x_j,x_{j-1}]+f[x_j,x_{j-1},x_{j-2}](x_j-x_{j-1})\\&+f[x_j,\cdots,x_{j-3}](x_j-x_{j-1})(x_j-x_{j-2})\\&+f[x_{j+1},\cdots,x_{j-3}](x_j-x_{j-1})(x_j-x_{j-2})(x_j-x_{j-3})\ .\end{aligned} \tag{3.7}$$

In this way, we can obtain an estimate for the derivative at all the interior points. For a kth order approximation to the derivative, the technique requires that the derivative discontinuities in the data be at least k points apart. The points that are close to the boundary are treated more one-sidedly. This completes our preprocessing phase.



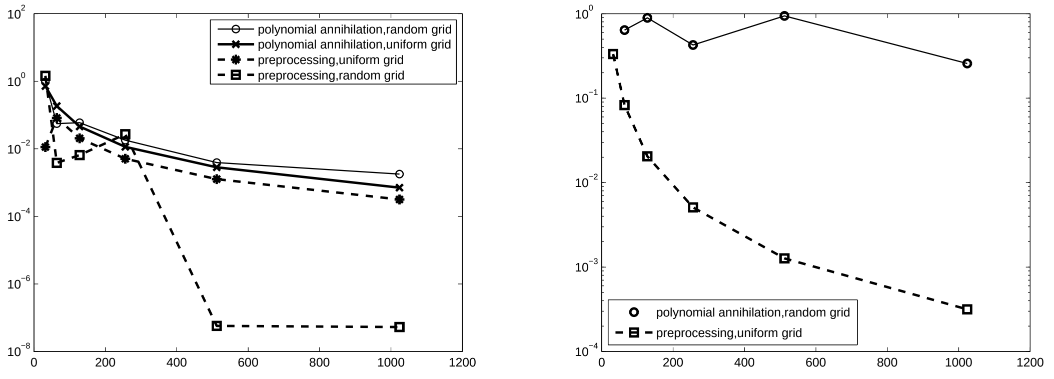
(a) $p_4'(x)$ **(b)** $log|p_4'(x)-f'(x)|$ **(c)** $L_1f'(x)$

(d) $L_2f'(x)$ **(e)** $L_4f'(x)$ **(f)** $MM\{L_mf'(x)\}$

Figure 3: Example 2.2 distributed on $N=128$ randomly distributed points. (a) The derivative computed using 4th order interpolation; (b) Logarithmic plot of the error in preprocessing $|p_4'(x)-f'(x)|$; (c)-(e) Jump function approximation L_mf', $m=1,2,4$; (f) Derivative edge detection, (2.4), after preprocessing.

To demonstrate the performance of the derivative computation described in this section, let us consider Example 2.2 once again. We used a fourth order derivative approximation with ENO type stencil choosing to do the preprocessing. Fig. 3(a) compares the reconstructed derivative to the actual derivative. Fig. 3(b) shows the logarithmic plot of the error in preprocessing.

Once we have the derivative approximation from the preprocessing step, we implement the polynomial annihilation edge detector, (2.3), to locate the derivative jump discontinuities of $f'(x)$. Note that the new function we get after preprocessing is itself a polynomial, so using the polynomial annihilation edge detector on such data should be particularly good. Fig. 3(c), (d) and (e) show the result of implementing (2.3) on (3.7). Fig. 3(f) demonstrates the minmod algorithm, (2.10), implemented on the preprocessed data, using $m=\{1,2,3,4,5\}$. In contrast to the method in [3], which is displayed in Fig. 2(f), no oscillations remain in the derivative jump function approximation. A comparison of the convergence of the two methods to the correct jump height is shown in Fig. 4(a). Fig. 4(b) shows the decay of oscillation observed in an adjacent pixel. As is evident from Fig. 3(f), there are no oscillations in the non-uniform case when we use the preprocessing approach. The same is true for the uniform case using the method in [3]. The polynomial annihilation derivative detector becomes ill-conditioned when the grid density, $h(x)$, becomes too small. Due to decreasing grid density, ill-conditioning occurs for $N\geq 2^{10}$ on uniform grids and for $N\geq 2^8$ on randomly distributed grids.



(a) **(b)**

Figure 4: Example 2.2. (a) Logarithmic plot of $|MM(Lmf(\xi))-[f'](\xi)|$, the error in jump height from the two methods on two different grids (one uniform, the other random), for $N=2^j$, $j=5,\cdots,10$; (b) Logarithmic plot of $|MM(L_m f(\hat{\xi}))|$ ($\hat{\xi}$ is a pixel adjacent to ξ), the height of the oscillation in the neighboring pixel.

3.2 Two dimensions

In this section we present a second order reconstruction of the gradient of a function, which is similar to the ENO idea proposed in [1]. We consider a function $f:\Omega\rightarrow R$, $\Omega\subset R^2$, known only on the set of discrete irregularly scattered points $S=\{(x_i,y_i),i=1,\cdots,N\}\subset\Omega$. We start by obtaining a Delaunay triangulation, (2.11), displayed in Fig. 5. Our derivative reconstruction points will be the grid points themselves. Let $P(x_0,y_0)\in S$ be an interior point of the grid where we want to reconstruct the partial derivatives. A second order accurate interpolating polynomial in the vicinity of (x_0,y_0) is given as

$$\begin{aligned} P(x,y)=&f(x_0,y_0)+(x-x_0)a_{10}+(y-y_0)a_{01}+(x-x_0)^2a_{20}\\ &+(x-x_0)(y-y_0)a_{11}+(y-y_0)^2a_{02}\ . \end{aligned} \tag{3.8}$$

To obtain the coefficients in (3.8), we need at least five points in the neighborhood of the point (x_0,y_0). The points immediately surrounding the point (x_0,y_0) are a natural choice, so let us consider the stencil PABCDEF that encloses $P(x_0,y_0)$ as shown in Fig. 5. We define this stencil as our primary stencil $S_{nobias}=\{(x_i,y_i)\}_{i=1}^6$ with the corresponding function values $\{f(x_i,y_i)\}_{i=1}^6$. Substituting each of the points $\{(x_i,y_i,f_i)\}_{i=1}^6$ into (3.8) yields a 6×5 system of linear equations, resulting in an over-determined system. In such a situation, the coefficients are determined by a least squares approximation. (When the number of neighbors is exactly five, we use interpolation.) Suppose our solution is $(a^*_{10},a^*_{01},a^*_{20},a^*_{11},a^*_{02})$. If $f(x,y)$ is smooth inside this stencil, then $P(x,y)$ in (3.8) is a second order accurate polynomial approximation for $f(x,y)$ inside the nobias[‡] stencil. Then a second order approximation to the function can be written as (3.8) over the region

[‡] By nobias we mean the centered stencil.

$ABCDEF$ with partial derivatives of the function given by

$$\begin{aligned} f_x(x,y) \approx P_x(x,y) = a^*_{10} + 2a^*_{20}(x-x_0) + a^*_{11}(y-y_0)\,, \\ f_y(x,y) \approx P_y(x,y) = a^*_{01} + 2a^*_{02}(y-y_0) + a^*_{11}(x-x_0)\,. \end{aligned} \tag{3.9}$$

Now suppose the stencil S_{nobias} has a derivative discontinuity running through it. Then the constants that we have solved for no longer approximate the corresponding derivatives; instead they are of large order due to the presence of the discontinuity. Therefore we see that in order to determine the constants correctly, we first need to choose a stencil that is smooth enough. In this context, we state the following theorem proved in [1]:

Theorem 3.2. *Let ϵ be a positive real number and S^n an admissible stencil for degree n such that there exists an affine transformation $\mathcal{A}$ for which $\mathcal{A}(S^n) \in \mathcal{P}^n_\epsilon$, where $\mathcal{P}^n_\epsilon$ denotes the set of possible stencils for which the total degree of $P(x,y)$ is exactly n. If S^n is an admissible stencil of degree n, then $\mathcal{K}(S^n)$ denotes the convex hull of the union of the elements of S^n and h denotes the diameter of $\mathcal{K}(S^n)$. Let (x_0,y_0) be any point of the set $\mathcal{K}(S^n)$ and f a real valued function defined on an open set Ω of R^2 containing $\mathcal{K}(S^n)$. We assume that f is $C^{p-1}, p<n$, in Ω and, except on a locally C^1 curve, admits a continuous and bounded pth derivative with a jump $[D^p u]$ so that*

$$|[D^p u]| > M_p > 0.$$

Then the highest degree coefficients of the Taylor expansion of $P(x,y)$ ((3.8) for $n=2$) satisfy

$$\sum_{i+j=n} |a_{ij}| \geq C(n,p,\epsilon)\frac{M_p}{h^{n-p}},$$

where $C(n,p,\epsilon)$ is a constant independent of S^n and invariant by affine transformation.

Theorem 3.2 says that if our function has a jump in the pth derivative inside a stencil of degree n, then the highest degree coefficients in the Taylor's expansion based on this stencil will tend to infinity as the mesh size tends to zero. In a stencil where f is smooth, these coefficients will always remain bounded.

Now suppose $f(x)$ is continuous in the domain (i.e., $f \in \mathcal{C}^0$) but has potential discontinuities in the first derivative. Then Theorem 3.2 applies with $p=1$ and $n=2$, and in this case, the second order coefficients in the Taylor's expansion would be relatively large (depending on h). Such information can be used to determine a stencil that avoids the derivative discontinuity.

We consider each triangle of our current stencil, one at a time. Let us start with $\triangle PDC$, i.e., T_1. We want to solve (3.8), except now using a stencil that is based on T_1 which does not include any other triangle from S_{nobias}. Hence, as shown in Fig. 5, we consider the triangles t_1, t_2 and t_3. We choose the vertices G, H and K of these triangles to complete our stencil. Our new stencil, therefore, is $S_1 = PDKGHC$. We solve the 5×5 system that we get from this stencil again to get a new set of constants $\{a^*_{10}, a^*_{01}, a^*_{20}, a^*_{11}, a^*_{02}\}$.

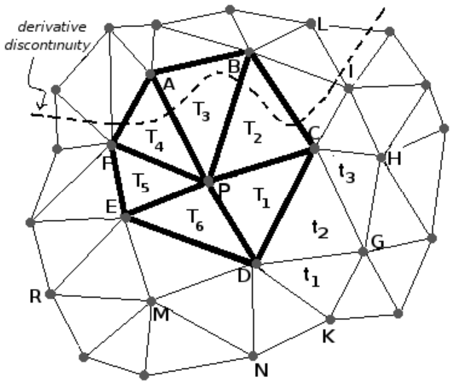


Figure 5: Derivative discontinuity running through the nobias stencil.

We repeat the process for each of the triangles T_1,T_2,T_3,T_4,T_5 and T_6 (and thus, in six different directions). Next, we consider the following sum for each stencil

$$\sum_{i+j=2} |a^*_{ij}|. \tag{3.10}$$

For the stencils that contain a discontinuity somewhere (for example, the stencil S_{nobias} and those based on T_3,T_4,T_5), (3.10) will be a large value, whereas the stencils that are completely smooth will result in coefficients that approximate the corresponding partial derivative. Thus, the stencil that minimizes (3.10) is the direction in which the function is smoothest. We choose the coefficients resulting from this stencil as our final coefficients and approximate the x- and y-derivatives at the point P by the corresponding first order coefficients as

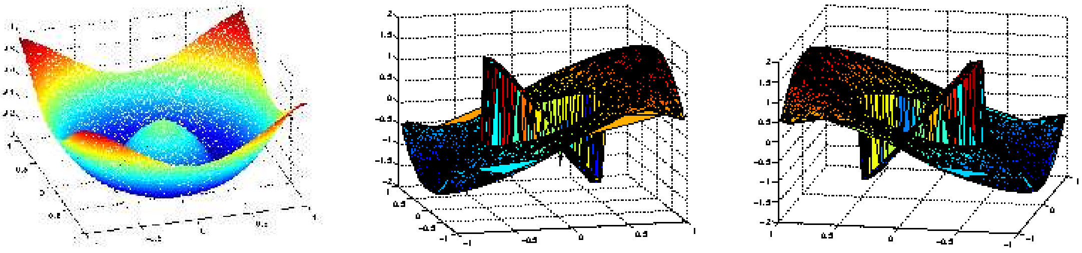
$$\begin{aligned} f_x(x_0,y_0) &\approx P_x(x_0,y_0) = a^*_{10}\ , \\ f_y(x_0,y_0) &\approx P_y(x_0,y_0) = a^*_{01}\ . \end{aligned} \tag{3.11}$$

Conceivably, there are several other stencil choices available (eg. stencil ABCDPF) that were ignored in our method. We limit ourselves to stencil choices described in the procedure here. In this way, the method remains simple to implement while still allowing a reasonable range of stencil choices. For points that have fewer than five immediate neighbors, the 'nobias' stencil does not have enough points to determine a second order approximation. In this case we only consider the different 'directional' stencils and then choose the smoothest of these as our final stencil.

The following example demonstrates the performance of the method:

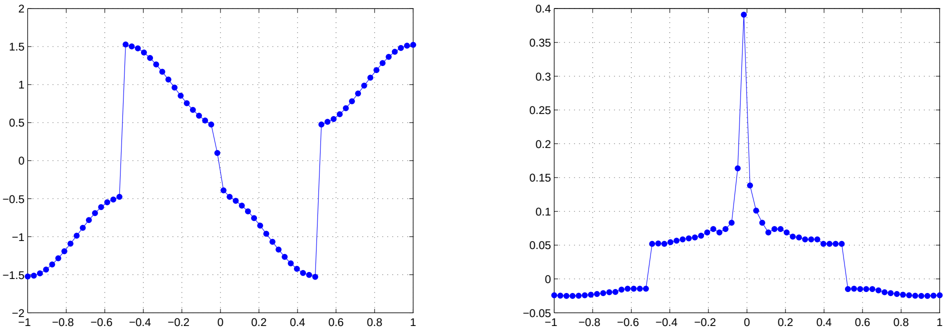
Example 3.1. $f:[-1,1]\times[-1,1]\rightarrow R$ such that

$$f(x,y)=\left\{\begin{array}{ll} -(\sqrt{x^2+y^2}-\frac{1}{2})+\frac{1}{12}\sin(2\pi\sqrt{x^2+y^2}), & \text{if } \sqrt{x^2+y^2}<\frac{1}{2}, \\ (\sqrt{x^2+y^2}-\frac{1}{2})+\frac{1}{12}\sin(2\pi\sqrt{x^2+y^2}), & \text{if } \sqrt{x^2+y^2}\geq\frac{1}{2}. \end{array}\right. \tag{3.12}$$



(a) ***given data*** **(b)** ***x-derivative*** **(c)** ***y-derivative***

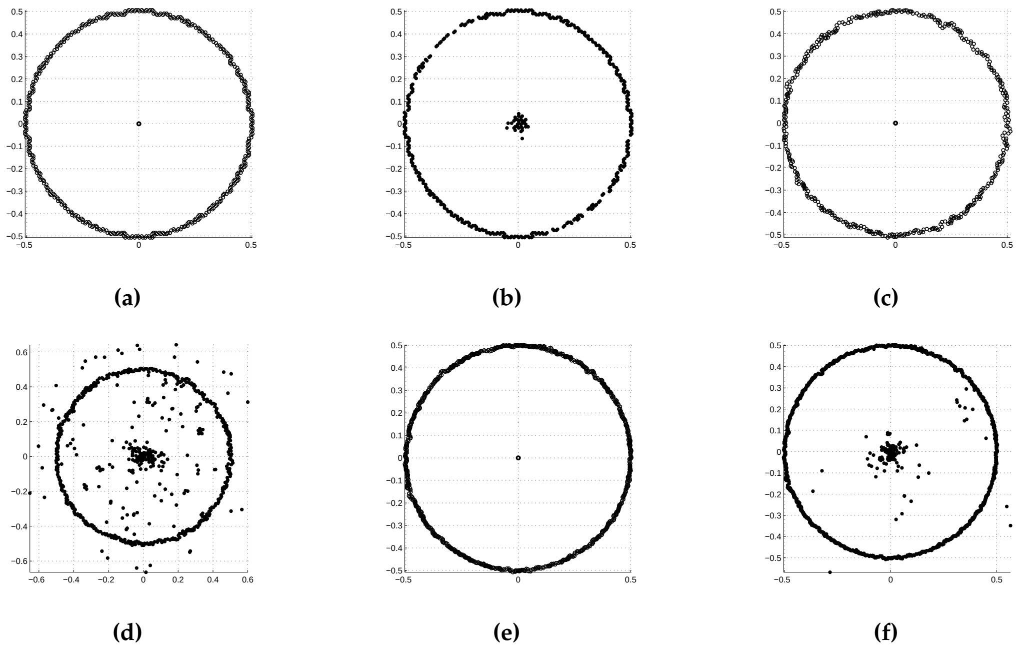
Figure 6: Figures for Example 3.1. (a) Original data; (b) x derivative computed using ENO; (c) y derivative computed using ENO.



(a) ***x-derivative*** **(b)** ***y-derivative***

Figure 7: Cross-sections of $f_x(x,y_j)$ and $f_y(x,y_j)$, where $j=Ny/2$, of Example 3.1 computed using the ENO method on a uniform grid of 64×64 points.

The data sampled on a grid of randomly distributed $N=64^2$ data points is shown in Fig. 6(a). It is continuous everywhere on the domain. The computed partial derivatives f_x and f_y are shown in Fig. 6(b) and (c). The one-dimensional cross sections are displayed in Fig. 7. Edges in the gradient are detected by taking a two pass approach, just as was done in one dimension. In the first phase, we preprocess the data using the ENO method to compute f_x and f_y. In the second phase, we implement the polynomial annihilation edge detector on this preprocessed information separately for each gradient. This gives us the edges in the $x-$ and $y-$derivatives. The edges from the two directions are then suitably combined (we used the discrete L_2 norm) to produce the desired result. Fig. 8 demonstrates the gradient jump function for Example 3.1 on uniform and random grids with a total of 128^2 and 256^2 points. The polynomial annihilation edge detector was used with orders $m=1,2,3$, followed by minmod application, (2.10). In addition, any edges whose magnitude fell below a threshold, τ, were discarded. Our experiments show that



(a) **(b)** **(c)**

(d) **(e)** **(f)**

Figure 8: Comparisons of the real edge locations with the computed ones. (a) True edges on a uniform grid of size 128 × 128; (b) Detected edges on a uniform grid of size 128 × 128; (c) True edges on a random grid with a total of 128^2 points; (d) Detected edges on a random grid with a total of 128^2 points; (e) True edges on a random grid with a total of 256^2 points; (f) Detected edges on a random grid with a total of 256^2 points. We used $\tau = 0.05\max(MM(L_m f(x)))$ for the results displayed in (b), (d) and (f).

Table 1: Performance on uniform grids (left) and non-uniform grids (right) for $\tau = 0.05\max(MM(L_m f(x)))$.

#S	*points missed*	*false alarms*	*#S*	*points missed*	*false alarms*
32 × 32	5(≈4.55%)	191(≈64.52%)	32^2	10(≈9.71%)	285(≈75.39%)
64 × 64	10(≈4.65%)	70(≈25.45%)	64^2	10(≈4.52%)	419(≈66.51%)
128 × 128	42(≈9.81%)	33(≈7.87%)	128^2	12(≈2.79%)	251(≈37.57%)
256 × 256	96(≈10.91%)	57(≈6.78%)	256^2	6(≈0.69%)	151(≈14.83%)

the false alarms mostly disappear for

$$\tau > 0.1\max(MM(L_m f(x))).$$

Table 1 shows the performance of the method on different grids as we increase the resolution. This does not reflect completely how well the method works, however, since there is only a single point of discontinuity at $(x,y) = (0,0)$, and all of the neighboring detected discontinuities are counted as false alarms. Enhancements to further reduce the number of false alarms will be introduced in future investigations. In the case of uniform grids, the method seems to miss more points in the second and fourth quadrants. This is possibly because of the way the Delaunay triangulation was obtained but requires more investigation. Further, the tables are perhaps not the best way to compare the performance of the method for uniform and nonuniform point distributions. For example, it is entirely possible that a false alarm would affect two cells in the uniform case while only one cell is affected in the nonuniform case, or vice versa. Similar behavior can be observed when a point of discontinuity is missed.

4 Conclusion

The preprocessed gradient edge detection discussed in this paper works well and is competitive with other recently proposed techniques. One of the most important assets of the method is that it is applicable to irregular grids. The edges in the derivatives are detected with greater accuracy as the resolution is increased. Further, the polynomial preprocessor is naturally suitable for use with the polynomial annihilation edge detector. In this paper, we assume that the given data is continuous everywhere. However, in a more general situation where the function itself has discontinuities, the polynomial annihilation method, [2], can first determine the discontinuities. Their corresponding cells can subsequently be removed and our gradient preprocessing edge detection method will then detect derivative jump discontinuities in cells away from the already identified jump locations. By repeating the process a number of times as demanded by the situation, jumps in successive higher order derivatives (or gradients) can be located. This method is particularly useful for identifying smooth regions for solving partial differential equations. The use of the least squares approach for the often over-determined nobias stencils in two dimensions might be advantageous when the data is noisy, and will be studied in future investigations.

Acknowledgments

We would like to thank the anonymous referees for their useful comments. This work was partially supported by NSF grants CNS 0324957, DMS 0617867, DMS 0608844 (AG), DMS 0510813 (AG and HM), and DMS 0421846 (AG and HM).

References

[1] R. Abgrall, On essentially non-oscillatory schemes on unstructured meshes: Analysis and implementation, J. Comput. Phys., 114 (1994), 45-58.

[2] R. Archibald, A. Gelb and J. Yoon, Polynomial fitting for edge detection in irregularly sampled signals and images, SIAM J. Numer. Anal., 43 (2005), 259-279.

[3] R. Archibald, A. Gelb and J. Yoon, Determining the locations and discontinuities in the derivatives of functions, Appl. Numer. Math., online Feb 13 (2007).

[4] R. B. Bauer, Band Pass Filters for Determining Shock Locations, Ph.D. thesis, Applied Mathematics, Brown University, 1995.

[5] J. Canny, A computational approach to edge detection, IEEE T. Pattern Anal. Machine Intell., 8 (1986), 679-698.

[6] A. Gelb and E. Tadmor, Adaptive edge detectors for piecewise smooth dara based on the minmod limiter, J. Sci. Comput., 28 (2006), 279-306.

[7] A. Harten, B. Engquist, S. Osher and S. R. Chakravarthy, Uniformly high order accurate essentially non-oscillatory schemes, III, J. Comput. Phys., 131 (1997), 3-47.

[8] E. Heldrith and D. Marr, Theory of edge detection, Proc. Roy. Soc. Lond. B, 207 (1980), 187-217.

[9] C. Hu and C.-W. Shu, Weighted essentially non-oscillatory schemes on triangular meshes, J. Comput. Phys., 150 (1999), 97-127.

[10] W. Hwang and S. Mallet, Singularity detection and processing with wavelets, IEEE T. Info. Theor., 38 (1992), 617-643.

[11] A. Jain, Fundamentals of Digital Image Processing, Prentice Hall, New Jersey, 1986.

[12] C.-W. Shu, Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws, Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, Vol. 1687 of Lecture Notes in Mathematics, 325-432, 1998.

[13] K. Siddiqi, B. B. Kimia and C.-W. Shu, Geometric shock capturing ENO-schemes for subpixel interpolation, Computation and Curve Evolution, Graphical Models and Image Processing, 59 (1997), 278-302.

[14] I. Sobel, An isotropic 3 image gradient operator, Machine Vision for Three-Dimensional Scenes (H. Freeman ed.), Academic Press, Boston, 1990.