J Sci Comput (2011) 46: 485–518
DOI 10.1007/s10915-010-9416-8

Edge Detection by Adaptive Splitting

Bernardo Llanas · Sagrario Lantarón

Received: 6 October 2009 / Revised: 1 July 2010 / Accepted: 24 August 2010 /
Published online: 5 September 2010

Abstract In this paper we propose an algorithm (EDAS-d) to approximate the jump discontinuity set of functions defined on subsets of $\mathbb{R}^d$. We have limited our study to the 1D (EDAS-1) and 2D (EDAS-2) versions of the algorithm. Theoretical and computational results prove its effectiveness in the case of piecewise continuous 1D functions and piecewise constant 2D functions. The algorithm is based on adaptive splitting of the domain of the function guided by the value of an average integral. EDAS-d exhibits a number of attractive features: accurate determination of the jump points, fast processing, absence of oscillatory behavior, precise determination of the magnitude of the jumps, and ability to differentiate between real jumps (discontinuities) and steep gradients. Moreover, low-dimensional versions of EDAS-d can be used for solving higher dimensional problems. Computational experiments also show that EDAS-d can be applied to solve some problems involving general piecewise continuous functions. EDAS-1 and EDAS-2 have been used to determine edges in 2D-images. The results are quite satisfactory for practical purposes.

Keywords Edge detection · Jump discontinuity set · Adaptive splitting · Implicit smoothing · Stratified edges

1 Introduction

1.1 The Edge Detection Problem

Edges can be defined as the boundaries of homogeneous regions. Their determination is important in many practical applications. Examples include image segmentation [18, 32], grain

B. Llanas (✉) · S. Lantarón
Departamento de Matemática Aplicada, ETSI de Caminos, Universidad Politécnica de Madrid, Ciudad Universitaria s/n, 28040 Madrid, Spain
e-mail: ma07@caminos.upm.es

S. Lantarón
e-mail: sagrario.lantaron@upm.es

boundary extraction in materials science [36], modeling of organs and internal structures of the human body [8, 30] and detection of solid-liquid interfaces [36].

Edge detection can also be used to enhance the result of some numerical algorithms:

- In function reconstruction using Fourier data, the Gegenbauer method can avoid Gibbs oscillations, but *a priori* knowledge of the edges of the function is required [21].
- Post-processing techniques for the discontinuous Galerkin method were introduced for reconstructing solutions near computational boundaries and discontinuities in the solution, as well as for changes in mesh size. These techniques require *a priori* knowledge of shock locations [2].

A function $f : \mathbb{R}^d \to \mathbb{R}$ can be specified in two main ways:

- Data in the physical space: Given a vector $\mathbf{x}$, its image $f(\mathbf{x})$ is available.
- Fourier spectral data: The Fourier coefficients of f are available.

According to this classification, many methods have been proposed to determine edges:

- In the case of data in the physical space:
  - Smoothing splines. These methods proceed to find a spline approximation of the discontinuous function and use the differentiability properties of the approximant for obtaining edges. The smoothing property helps to solve the problems that arise due to the presence of noise [10–12]. Other authors use related wavelet techniques [23, 38].
  - Variational methods. The edges of the function are obtained by minimizing a suitable functional (Mumford-Shah, etc.). In this category we can include the active contour methods (snakes) [25, 32].
  - Finite difference methods. Local difference formulas have been studied in [16]. They are typically limited to low-order of accuracy due to the oscillatory behavior of high-order detectors.
  - Polynomial fitting method. This method, proposed in [3], is based on a local polynomial annihilation property on a set of irregularly distributed points in a bounded domain of $\mathbb{R}^d$. This procedure successfully captures discontinuities that are identified by their enclosed cells by characterizing the convergence away from the discontinuities.
  - Spatial filtering. These methods use filters or masks that operate on digital images. They obtain a new raster image performing some computations on the neighbors of each pixel. If these computations are linear the operation is called linear spatial filtering. Otherwise it is called nonlinear spatial filtering. Examples of these filters are the Sobel, Prewitt and Canny edge detectors [20].
- In the case of Fourier spectral data:
  - Concentration factor methods. They are based on the fact that the conjugate partial sum of N terms multiplied by $-\pi/\log N$ converges to the jump function $[f](x) = f(x^+) - f(x^-)$. Due to the slow rate of convergence ($O(1/\log N)$) the so-called "concentration factors" were introduced in [14]. These functions accelerate the convergence of the conjugate partial sums [14, 15]. The spurious oscillations near the discontinuities are avoided by an outside threshold parameter. In [16] these oscillations are suppressed by an adaptive edge detector based on a cross-heading between local and global detectors achieved by means of the *minmod* limiter. Refinements to the concentration method that reduce artificial oscillations and improve its effectiveness in noisy environments were proposed in [17]. Examples of region extraction from Fourier coefficients using the above techniques are given in [18].

Most methods for determining edges, try to approximate the target function $f(x)$ (smoothing splines, etc.) or the jump function $[f](x)$ (polynomial fitting, concentration methods, etc.). The first type of methods requires a considerable computational effort. The second type of procedures presents problematic oscillations in the neighbourhood of discontinuities [3, 14]. This problem arises also in the case of high-order finite difference methods [16].

1.2 Edge Detection Methods Based on Adaptive Meshing

Adaptive meshing methods for edge detection have been studied mainly in the context of image processing (discrete functions). Below we describe the most significant approaches in this field

1. Image Multiscale Analysis. In this approach an image can be modeled as a real function $I_0(\mathbf{x})$ representing values of gray-level intensity, defined on some subset of $\mathbb{R}^d$ $(d = 2, 3)$. Basic image processing operations such as selective image smoothing, edge detection and segmentation can be modeled by the application of an evolutionary PDE to $I_0(\mathbf{x})$. Such approach is known as image multiscale analysis, since the initial image $I_0(\mathbf{x})$ is associated with a sequence of images $I_t(\mathbf{x})$ depending on an abstract parameter $t > 0$ called scale. Examples of such PDEs are the nonlinear anisotropic diffusion equations of Perona-Malik type [34], and the generalized mean curvature flow equations [1]. The Perona-Malik equation was motivated by the fact that conventional diffusion can eliminate the noise but the edges are blurred in the process. Anisotropic diffusion can eliminate the noise and enhance edges while running forward in time.

   Adaptive (nonuniform) grids for the finite element method applied to the Perona-Malik equation with the modification suggested in [9] were studied in [4] (and generalized to the 3D case in [5]). This method is based on simplicial grids generated by bisection and then again successively coarsened in the diffusion process. An improvement of storage requirements by procedural handling of adaptive grids is proposed in [35].

   These methods are computationally expensive because they require the multiplication of large matrices, the solution of large linear systems, etc.
2. Image representation. Image mesh modeling techniques intend to build a mesh that minimizes an error measurement, usually the approximation error between the original image and that represented by the mesh (see [27] for a recent account). Many algorithms devoted to represent images by meshes are based on adaptive approaches. Most adaptive methods produce an image segmentation using a variant of the split and merge method. This method, first proposed by Horowitz and Pavlidis [24], considers an homogeneity property (split and merge criterion) and successively subdivides the regions which do not satisfy it. In this process the neighboring regions whose union satisfies the homogeneity property are merged into a single region. Several criteria have been proposed as homogeneity property, for example, oscillation of the gray-level pixel intensity in the region under consideration, degree of approximation by polynomials, etc. In [24] the homogeneity property is defined by the degree of approximation by constants and the subdivision is performed using quadtrees. The edges of the subsets generated by this algorithm can have only two orientations (horizontal and vertical), and their positioning is restricted by the borders of the quadtree nodes. Therefore, it is difficult to achieve good matching in the case of image edges of arbitrary orientation or position. A postprocessing procedure consisting of boundary elimination and contour modification has been proposed in [33]. A new subdivision scheme that allows diagonal splitting and uses binary trees is proposed in [39]. The results obtained with these methods do not seem satisfactory. An

improved split and merge algorithm is proposed in [19]. The authors consider as homogeneity property the degree of approximation by polynomials of degree 2. In each region the local error is computed as the sum of the squares of the individual pixel errors divided by the number of pixels. The meshing procedure is an incremental Delaunay triangulation. When a triangle does not satisfy the homogeneity property, auxiliary algorithms are applied to determine the "edge" within the triangle: Canny's edge detector, corner detectors and Y-junctions detectors. Then an edge pixel is used as new vertex in the incremental Delaunay process. Experiments with simple images show that the method gives acceptable results even with noisy images but complex images are not tested.

To sum up, split and merge methods present several drawbacks in edge detection:

– Their aim is to produce an image segmentation. Edges are a subproduct of this process.
– Since edges are considered as the boundaries of homogeneous regions, the definition of edge depends on the homogeneity property considered.
– They determine edges as sets of sides of the grid elements. This method has proved to be excessively rigid in practice.
– The evaluation of the homogeneity property involves all the pixels within a region. This can be computationally expensive.

1.3 Contribution of the Paper

An adaptive splitting approximation algorithm was proposed in [29]. In the present paper we prove that the average integral used in its splitting criterion is an effective jump detector (Propositions 3, 4 and Theorem 2). These results imply that the algorithm is divergent for functions with jump discontinuities, but they allow us to modify it to obtain an efficient edge detection method. The parameters of the approximation algorithm have a new additional meaning. The local error gives the detection threshold. The stopping criterion gives the precision of the obtained jump points. We introduce new parameters to optimize the search for jumps and the numerical computations. We call EDAS-d the resulting algorithm.

EDAS-d is an algorithm to detect jump discontinuities of functions defined by data in the physical space. The domain of the function is supposed to be the difference of a convex set and a set of Lebesgue measure zero. This method does not try to accurately approximate $f(x)$ or $[f](x)$. Its output consists of an approximation of the jump discontinuity set of the function (see Sect. 2). It is based on an integral inequality fulfilled by the sets containing a jump discontinuity. It enables distinction between jumps discontinuities (Fig. 1(a)) and steep gradients (Fig. 1(b)); see also [28]. Since it exhibits a non-oscillatory behavior we can determine the magnitude of the jumps.

The algorithm builds a piecewise affine function which provides a rough approximation of f away from the discontinuities. In the neighbourhood of an edge, this function is a good approximant. The integrals whose evaluation is necessary to obtain the piecewise affine function are computed by numerical methods which are exact for polynomials of a certain degree. This provides an "implicit smoothing" of the target function, that accelerates the convergence away from the discontinuities and makes the algorithm robust in the presence of continuous noise.

The choice of the local error is guided by the magnitude of the searched jumps rather than by the degree of approximation away from the discontinuities. This allows to obtain good results with a poor approximation (homogeneity property) when the magnitude of the jumps is large enough.

When we apply EDAS-d to determine the edges of a 2D image (discrete function), we extend it to a function defined on a square containing the discrete domain (see Sect. 4.2).

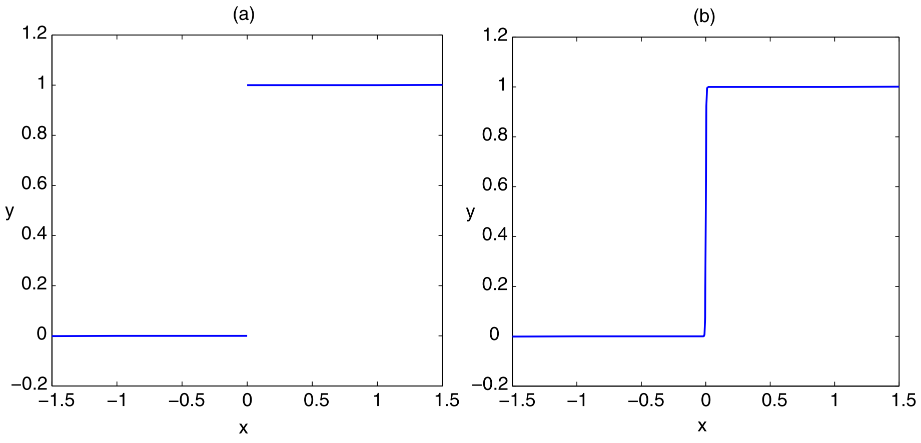


Fig. 1 (**a**) Graph of the Heaviside function. (**b**) Graph of $y = 1/(1 + e^{-500x})$

The resulting function is piecewise constant (each pixel is represented by a square and the function is constant on it). EDAS-d does not try to segmentate the image, because it is not concerned about its accurate approximation. EDAS-d determines a set of triangles containing jump discontinuity points. The approximate result is a flexible set of points instead of a rigid set of lines. The numerical cubature process is faster than operations involving large amounts of pixels. EDAS-d can consider triangles with arbitrary size. Since they can intersect the boundary of two or more pixels, the splitting criterion based on Theorem 2 can be applied. Finally, the average integral criterion, the implicit smoothing and a suitable choice of the minimum magnitude of jump reported can reduce the noise of the image.

The paper is organized as follows. Section 2 gives some mathematical preliminary results. Section 3 describes the EDAS-d algorithm. Section 4 presents computational experiments and compares EDAS-d with other edge detection algorithms. Section 5 provides some concluding remarks. The paper is closed with an Appendix containing the proof of some results.

2 Mathematical Preliminaries

Let $R \subset \mathbb{R}^d$ be a compact d-interval. We say that a function $g : R \to \mathbb{R}$ is *quasi-continuous* if the set of points where g is not continuous has zero Lebesgue measure.

Consider a finite collection $\{C_i\}_{i=1}^n$ of connected sets with pairwise disjoint nonempty interiors such that

$$R = \bigcup_{i=1}^{n} C_i.$$

Let $\{f_i : i = 1, \dots, n\}$ be a set of continuous functions on R. Define the function

$$f(\mathbf{x}) \equiv f_i(\mathbf{x}) \quad \text{if } \mathbf{x} \in \mathring{C}_i,$$

where $\mathring{C}$ denotes the interior of C. We say that f is a *general piecewise continuous function*. If C_i, $i = 1, \dots, n$, are closed and convex sets, we say that f is a *piecewise continuous* *function*. Since the boundary of a convex set has zero Lebesgue measure, all piecewise continuous functions are quasi-continuous.

In the above definition, if the functions f_i, $i = 1, \dots, n$, are constant, we say that f is a *general piecewise constant function* and a *piecewise constant function*, respectively.

Let f be a general piecewise continuous function and let Γ_i be the boundary of C_i, $i = 1, \dots, n$. Define $\Gamma \equiv \bigcup_{i=1}^{n} \Gamma_i$. Let $\mathbf{x} \in \Gamma$, then for some $m \leq n$, $\mathbf{x} \in \Gamma_{i_j}$, $j = 1, \dots, m$, where $i_j \in \{1, 2, \dots, n\}$. Define

$$A \equiv \max_{j=1,\dots,m} \{f_{i_j}(\mathbf{x})\}, \quad \text{and} \quad B \equiv \min_{j=1,\dots,m} \{f_{i_j}(\mathbf{x})\}.$$

If $A \neq B$ we say that f has a *jump discontinuity* at $\mathbf{x}$ (we also say that $\mathbf{x}$ is a *jump point*). We call $|A - B|$ magnitude of the *jump* of f at $\mathbf{x}$. The set of points in Γ with jump discontinuities is called *jump discontinuity set* and denoted by Γ^J. We call *edge* any subset of the jump discontinuity set. The set of points in Γ^J with magnitude of jump greater than l is denoted by Γ_l^J. The set of points in Γ^J with magnitude of jump greater than l and lower than u is denoted by Γ_{lu}^J. Our aim is to obtain a good approximation of these sets for a given function.

We use the following notation: Let $\{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_d\}$ be a set of $d + 1$ vectors in $\mathbb{R}^d$. This set is called affinely independent if the vectors $\{\mathbf{v}_1 - \mathbf{v}_0, \mathbf{v}_2 - \mathbf{v}_0, \dots, \mathbf{v}_d - \mathbf{v}_0\}$ are linearly independent. Suppose now that $\{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_d\}$ is an affinely independent subset of $\mathbb{R}^d$. The *d-simplex T* generated by $\{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_d\}$, denoted by $\langle \mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_d \rangle$, is defined to be the convex hull of the vectors $\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_d$. We denote by $|T|$ the diameter of a d-simplex T.

Proposition 1 *Let $T = \langle \mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_d \rangle$ be a d-simplex and let $w_0, w_1, \dots, w_d$ be arbitrary real numbers. Then there is a unique affine map L_T from T to $\mathbb{R}$ such that $L_T(\mathbf{v}_j) = w_j$, $j = 0, 1, \dots, d$.*

Proof See [13]. □

Consider a function $f : T \subset \mathbb{R}^d \to \mathbb{R}$. We denote by $L_T f$, the unique affine map such that

$$L_T f(\mathbf{v}_j) = f(\mathbf{v}_j), \quad j = 0, 1, \dots, d.$$

In this section we study the average integral

$$AI_T(f) \equiv \frac{\int_T |f(\mathbf{x}) - L_T f(\mathbf{x})| d\mathbf{x}}{\mathrm{v}(T)},$$

where $\mathrm{v}(T)$ denotes the Lebesgue measure of T.

As we prove below, the behavior of $AI_T(f)$ depends on the continuity or lack of continuity of f on T. This fact lays the foundation for the algorithm proposed in the next section. The comportment of $AI_T(f)$ when f is continuous is given by the following results.

Proposition 2 *Let $S \subset \mathbb{R}^d$ be compact and let $f : S \to \mathbb{R}$ be continuous. Then given $\varepsilon > 0$, there exists $\delta > 0$ such that*

$$|f(\mathbf{x}) - L_T f(\mathbf{x})| < \varepsilon$$

for all $\mathbf{x} \in T$, where T is a d-simplex contained into S with $|T| < \delta$.

Proof See [13]. □

Corollary 1 *Let $S \subset \mathbb{R}^d$ be compact and let $f : S \to \mathbb{R}$ be continuous. Then given $\varepsilon > 0$, we have that $AI_T(f) < \varepsilon$ for any small enough d-simplex $T \subset S$.*

If f is smooth the above result can be sharpened. Consider the vectors $\mathbf{z}^1 = (z_1^1, z_2^1, \dots, z_d^1)$ and $\mathbf{z}^2 = (z_1^2, z_2^2, \dots, z_d^2)$ with $\| \mathbf{z}^1 \| = 1$ and $\| \mathbf{z}^2 \| = 1$ ($\| \, . \, \|$ denotes the Euclidean norm on $\mathbb{R}^d$). If $f : \mathbb{R}^d \to \mathbb{R}$, define

$$D_{\mathbf{z}^1} f(\mathbf{x}) \equiv \sum_{k=1}^{d} z_k^1 D_k f(\mathbf{x}),$$

$$D_{\mathbf{z}^1 \mathbf{z}^2} f(\mathbf{x}) \equiv D_{\mathbf{z}^2}(D_{\mathbf{z}^1} f(\mathbf{x})) = \sum_{k,l=1}^{d} z_l^2 z_k^1 D_{lk} f(\mathbf{x}).$$

Let $G \subset \mathbb{R}^d$, denote by $W_d^2 M(G)$ the class of functions with continuous derivatives $D_{\mathbf{z}^1 \mathbf{z}^2} f(\mathbf{x})$ on G, not exceeding in absolute value $M > 0$, for any $\mathbf{z}^1, \mathbf{z}^2$.

Theorem 1 *Let T be a d-simplex in $\mathbb{R}^d$ and let $f \in W_d^2 M(T)$, then*

$$\max_{\mathbf{x} \in T} |f(\mathbf{x}) - L_T f(\mathbf{x})| \le \frac{dM|T|^2}{4(d+1)}.$$

Proof See [37]. □

Corollary 2 *Let $f \in W_d^2 M(S)$, where S is a nonempty compact set in $\mathbb{R}^d$, and let T be a d-simplex contained in S. Then*

$$AI_T(f) \le \frac{dM|T|^2}{4(d+1)}.$$

From the above results, we can conclude, in the case of continuous functions, that $AI_T(f) \to 0$ as $|T|$ approaches zero. We study below the case in that f presents jump discontinuities on T, when $d = 1$ and $d = 2$.

Consider the piecewise continuous function

$$f(x) = \begin{cases} Q(x), & \text{if } x < 0, \\ P(x), & \text{if } x > 0, \end{cases}$$

where $P(x) = a_0 + a_1 x + \cdots + a_n x^n$ and $Q(x) = b_0 + b_1 x + \cdots + b_m x^m$. It is clear that $f(0^+) = a_0$ and $f(0^-) = b_0$. Define $T(\eta) \equiv [-\eta, \eta]$ ($\eta > 0$). We have the following result

Proposition 3 *Given $\varepsilon > 0$ there exists a number $\delta > 0$ such that*

$$\frac{1}{2\eta} \int_{-\eta}^{\eta} |f(x) - L_{T(\eta)} f(x)| dx > \frac{|a_0 - b_0|}{4} - \varepsilon$$

for all η such that $0 < \eta < \delta$.

The proof of this proposition is given in the Appendix.

The above result shows that the expression

$$\frac{1}{2\eta}\int_{-\eta}^{\eta}|f(x)-L_{T(\eta)}f(x)|dx$$

is always greater than a positive quantity irrespective of the smallness of η. We shall use this fact as a criterion to find intervals containing jump points. From now on, we limit ourselves to the case where $f(x)$ is a function of Heaviside type. This suffices to study the image processing applications given in this paper and simplifies the results.

Consider the Heaviside function

$$H(x)\equiv\begin{cases}0, & \text{if } x\le 0,\\ 1, & \text{if } x>0.\end{cases}$$

Define the function $h(x)\equiv JH(x-\alpha)$ where $J>0$ and α are real numbers. We have the following result

Proposition 4 *Let T be a compact interval in $\mathbb{R}$ and $\alpha\in\mathring{T}$. Then*

$$\frac{J}{4}\le\frac{\int_T|h(x)-L_Th(x)|dx}{\mathrm{v}(T)}<J, \tag{1}$$

where $\mathrm{v}(T)$ is the length of the interval T.

Proof To prove (1) we use the following equivalent approach. Assume that $T=[a,b]$ is a fixed interval and vary α within T. If $\alpha\in(a,b)$

$$L_Th(x)=\frac{J(x-a)}{b-a}.$$

Define

$$\begin{aligned}I(\alpha)&\equiv\int_a^b|h(x)-L_Th(x)|dx=\frac{J}{b-a}\left(\int_a^\alpha(x-a)dx+\int_\alpha^b(b-x)dx\right)\\&=\frac{J}{b-a}(\alpha^2-(a+b)\alpha+(a^2+b^2)/2).\end{aligned}$$

$I(\alpha)$ attains a minimum at $\alpha=(a+b)/2$. Therefore

$$I(\alpha)\ge I((a+b)/2)=J(b-a)/4\quad\text{for all }\alpha\in(a,b).$$

This proves the left hand side inequality in (1).

Moreover

$$\begin{aligned}I(\alpha)&=\frac{J}{b-a}\left(\int_a^\alpha(x-a)dx+\int_\alpha^b(b-x)dx\right)\\&<\frac{J}{b-a}((b-a)(\alpha-a)+(b-a)(b-\alpha)=J(b-a).\end{aligned}$$

This proves the right hand side inequality in (1). □

Below we generalize the above result to the 2D case.

Theorem 2 *Let $h : \mathbb{R}^2 \to \mathbb{R}$ be the function defined by*

$$h(\mathbf{x}) \equiv JH(ax_1 + bx_2 + c),$$

where a, b, c and J are real numbers such that either a or b are distinct from zero and $J > 0$. Define $r \equiv \{\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2 : ax_1 + bx_2 + c = 0\}$ and let T be an arbitrary triangle in $\mathbb{R}^2$ such that $r \cap \mathring{T} \neq \emptyset$. Then

$$\frac{J}{4} \leq \frac{\int_T |h(\mathbf{x}) - L_T h(\mathbf{x})| d\mathbf{x}}{\mathrm{v}(T)} \leq J, \tag{2}$$

where $\mathrm{v}(T)$ is the area of T.

The proof of this theorem is given in the Appendix.

3 Edge Detection by Adaptive Splitting Algorithm

3.1 Statement of the Algorithm

We state the algorithm in the d-dimensional case. Consider the d-interval $R = [\mathbf{a}, \mathbf{b}]$, where $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$. We can find n (closed) simplices T_i such that

$$\mathring{T}_i \cap \mathring{T}_j = \varnothing, \quad i \neq j,$$
$$\bigcup_{i=1}^{n} T_i = R,$$

(for example, see [7, 31]). We call $P \equiv \{T_i\}_{i=1}^n$ a partition of R. The set of partitions of R is denoted by $\mathcal{P}(R)$. Given a d-simplex T we denote by $\mathcal{V}(T)$ the set of its vertices.

The proposed algorithm builds a partition $P \in \mathcal{P}(R)$ and the associated piecewise affine approximant $L(\mathbf{x})$ defined by

$$L(\mathbf{x}) \equiv L_T f(\mathbf{x}) \quad \text{if } \mathbf{x} \in T \subset P.$$

The average integral

$$AI_T(f) \equiv \frac{\int_T |f(\mathbf{x}) - L_T f(\mathbf{x})| d\mathbf{x}}{\mathrm{v}(T)},$$

can be considered as the local error of the approximant L.

Given a function $f : R \to \mathbb{R}$ and an initial partition P_1 of R, denote by E_1 the maximum local error of the approximant L. In the cases where Propositions 3, 4 or Theorem 2 can be applied, E_1 also provides a detection threshold, because the algorithm will detect the jumps with magnitude greater than $4E_1$. Denote by E_2 the approximation error of the points in Γ^J. Let E_4 be a positive real parameter.

If $(AI_T(f) \leq E_1$ or $|T| \leq E_2)$ and $|T| \leq E_4$ we call T a *good* simplex, otherwise T is called *bad*.

We call E_4 exploration parameter (in difficult problems, it is necessary to make E_4 small. This is due to the fact that an inaccurate numerical evaluation of $AI_T(f)$ can eliminate simplices containing points of Γ^J).

Edge Detection by Adaptive Splitting Algorithm (EDAS-d)

Step 1. The good simplices in the initial partition P_1 are put into the set $\mathcal{G}_1$ and the bad simplices are put into the set $\mathcal{B}_1$.

Step 2. At each step j we have a set of good simplices $\mathcal{G}_j$ and a set of bad simplices $\mathcal{B}_j$. Divide each bad simplex into two simplices, by splitting its largest edge. Test whether these children are good or bad to obtain the sets $\mathcal{G}_{j+1}$ and $\mathcal{B}_{j+1}$.

Step 3. The algorithm stops if $\mathcal{B}_j = \varnothing$. Then $G = \mathcal{G}_j$ is the searched partition. If the stopping criterion is not satisfied, go to Step 2.

Step 4. Obtain the following subset of G

$$A\Gamma^J_{E_3} \equiv \{T \in G : AI_T(f) > E_1 \quad \text{and} \quad j^a \equiv \max_{\mathbf{v}_i \in \mathcal{V}(T)} \{f(\mathbf{v}_i)\} - \min_{\mathbf{v}_i \in \mathcal{V}(T)} \{f(\mathbf{v}_i)\} > E_3\}. \tag{3}$$

where E_3 is the minimum magnitude of jump reported. E_2 is also a stopping criterion. The convergence of this algorithm is guaranteed by the definition of good simplex. $A\Gamma^J_{E_3}$ is a set of simplices containing points of $\Gamma^J_{E_3}$. This constitutes an approximation of $\Gamma^J_{E_3}$. In the cases considered by Proposition 4 and Theorem 2, we have that $A\Gamma^J_{E_3} \supset \Gamma^J_{E_3}$. In Sect. 4, instead of describing $A\Gamma^J_{E_3}$, we have considered the set of barycenters of $T \in A\Gamma^J_{E_3}$ to better visualize the result.

3.2 Practical Implementation of the Algorithm

In practice we have implemented the above algorithm using a tree whose leaves correspond to good simplices. Tree construction follows a technique similar to that detailed in [29].

The integral $\int_T |f(\mathbf{x}) - L_T f(\mathbf{x})| d\mathbf{x}$ has been computed in the following way:

– Dimension 1: We have used the Gauss-Legendre integration formula given in [26], that is

$$\int_{-1}^{1} f(x)dx = \sum_{k=1}^{n} A_k^{(n)} f(x_k^{(n)}).$$

This formula is exact for polynomials of degree less or equal than $2n-1$.

– Dimension $d \neq 1$: We have used the Grundmann and Möller's formula [22]. Let $\Delta = \langle (1,0,\dots,0), (0,1,\dots,0), \dots, (0,0,\dots,1) \rangle$ be the standard d-simplex in $\mathbb{R}^d$. If $p = 2s+1$ and $s \in \mathbb{Z}^+ \cup \{0\}$, the cubature rule of degree p is given by

$$\int_\Delta g d\mathbf{x} \simeq \sum_{i=0}^{s} (-1)^i 2^{-2s} \frac{(p+d-2i)^p}{i!(p+d-i)!} \sum_{\substack{|\beta|=s-i \\ \beta_0 \geq \cdots \geq \beta_d}} g\left(\left(\frac{2\beta_0+1}{p+d-2i}, \dots, \frac{2\beta_d+1}{p+d-2i}\right)_p\right), \tag{4}$$

where $\beta \equiv (\beta_0, \beta_1, \dots, \beta_d)$ is a $(d+1)$-tuple of nonnegative entire numbers with $|\beta| \equiv \beta_0 + \beta_1 + \cdots + \beta_d$. $(\mathbf{y})_p \equiv (y_0, y_1, \dots, y_d)_p$ denotes the set of all d-tuples that are the last

d components of all $(d+1)$-tuples derived from $\mathbf{y}$ by any permutation of the $d+1$ components of $\mathbf{y}$. The number of integration points in (4) is $\binom{d+s+1}{s}$. This formula is exact for polynomials of degree p.

Consider now the positive real parameter E_5 (adaptivity of cubature formulas parameter). This factor allows to apply higher degree cubature formulas in large regions and low degree cubature formulas in small regions. In this way the computational cost is optimized. The adaptive cubature procedure is described below.

- If $|T| < E5$
  - In 1D, the computations have been done using the Gauss-Legendre quadrature formulas with number of points $n = 5, 6, 7, 8, 9,$ and 10. Since the integrand functions are not smooth we have not analytical error estimates. We have followed the usual practice of comparing successive values of the integral corresponding to an increasing number of points and taking the value obtained when the sequence becomes stationary. We denote by $\widehat{AI}^n_T(f)$ the value of $AI_T(f)$ computed by the quadrature formula with n points. We denote by $\widehat{AI}_T(f)$ the final estimate of $AI_T(f)$. The proposed procedure is described by the following C-like pseudocode.

```
Set $n = 5$;
Compute $\widehat{AI}^5_T(f)$ and $\widehat{AI}^6_T(f)$;
while $(|\widehat{AI}^n_T(f) - \widehat{AI}^{n+1}_T(f)| \geq E_1/10$ && $n < 9)$
{
    $n = n + 1$;
}
if $(|\widehat{AI}^n_T(f) - \widehat{AI}^{n+1}_T(f)| < E_1/10)$   $\widehat{AI}_T(f) = \widehat{AI}^{n+1}_T(f)$;
else                                                              $\widehat{AI}_T(f) = 10^6$;
```

  - In 2D, the computations have been done using the Grundmann and Möller's rules of degree $p = 5, 7, 9, 11, 13,$ and 15 with $d = 2$. Due to the lack of error estimates for the Grundmann and Möller's rules, we follow a method similar to that used in the 1D case. We denote by $\widehat{AI}^p_T(f)$ the value of $AI_T(f)$ computed by the cubature rule of degree p. The pseudocode is described by the following statements.

```
Set $p = 5$;
Compute $\widehat{AI}^5_T(f)$ and $\widehat{AI}^7_T(f)$;
while $(|\widehat{AI}^p_T(f) - \widehat{AI}^{p+2}_T(f)| \geq E_1/10$ && $p < 13)$
{
    $p = p + 2$;
}
if $(|\widehat{AI}^p_T(f) - \widehat{AI}^{p+2}_T(f)| < E_1/10)$   $\widehat{AI}_T(f) = \widehat{AI}^{p+2}_T(f)$;
else                                                              $\widehat{AI}_T(f) = 10^6$;
```

- If $|T| \geq E_5$
  - In 1D, the computations have been done using the Gauss-Legendre quadrature formulas with number of points $n = 9$ and 10. The corresponding pseudocode is shown below.

```
if $(|\widehat{AI}^9_T(f) - \widehat{AI}^{10}_T(f)| < E_1/10)$   $\widehat{AI}(T) = \widehat{AI}^{10}_T(f)$;
else                                                           $\widehat{AI}_T(f) = 10^6$;
```

- In 2D, the computations have been done using the Grundmann and Möller's rules of degree $p = 13$ and 15 with $d = 2$. The corresponding pseudocode is listed below.

```
**if** $(|\widehat{AI}_T^{13}(f) - \widehat{AI}_T^{15}(f)| < E_1/10)$   $\widehat{AI}_T(f) = \widehat{AI}_T^{15}$;
**else**                                   $\widehat{AI}_T(f) = 10^6$;
```

NOTE: In many adaptive algorithms, the local error estimate *ERR* is simply taken as the absolute value of the difference between two cubature approximations [6]. In this way we have the rough bound $|I - \hat{I}| < ERR$, where I is the exact value of the integral and $\hat{I}$ is the value computed by the cubature rule. When $\hat{I} \simeq E_1$, it seems natural to consider that $ERR \ll E_1$. In the above pseudocode, we have taken $ERR = E_1/10$.

To sum up, the algorithm requires the following positive real parameters

- E_1: Maximum local error of the approximant L and detection threshold.
- E_2: Approximation error of the points in Γ^J and stopping criterion.
- E_3: Minimum magnitude of jump reported.
- E_4: Exploration parameter.
- E_5: Adaptivity of cubature formulas parameter.

3.3 Theoretical Foundation of EDAS-d

Results in Sect. 2 guarantee the convergence away from the discontinuities and the suitable local behavior of EDAS-d when $d = 1$ and $d = 2$.

- If $d = 1$, Proposition 4 provides a direct procedure for choosing the parameter E_1 in the case of piecewise constant functions. Proposition 3 and the Weierstrass approximation theorem justify the application of EDAS-1 to piecewise continuous problems.
- If $d = 2$, Theorem 2 provides the way of selecting E_1 in the case of piecewise constant functions. Inequality (2) may not be valid if the intersection of a triangle with the jump discontinuity set is not a straight line segment. In image processing, the set of points where Γ^J presents an angular shape is finite, therefore the above drawback has little practical importance.

4 Computational Experiments

The experimental environment has been the following:

- CPU QuadCore Intel Core 2 Quad 6700, 2666 MHz.
- RAM 4 GB.
- Operating system Windows XP.
- Running software Microsoft Visual C 6.0.

In all experiments, CPU time is expressed in seconds and denoted by CPU (s).

4.1 Computational Experiments on Piecewise Continuous 1D Functions

Given a function $f : T \subset \mathbb{R} \to \mathbb{R}$, where T is a closed interval, we denote by x_i^e its jump points and by j_i^e the magnitude of their corresponding jumps. The performance of EDAS-1

Table 1 Performance of EDAS-1 on the function H

E_1	E_2	E_3	E_4	E_5	NJ	NI	E_x	E_j	CPU (s)
10^{-1}	10^{-1}	10^{-2}	10^6	10.	1	6	3.1×10^{-2}	0.	0.0
10^{-1}	10^{-2}	10^{-2}	10^6	10.	1	9	3.9×10^{-3}	0.	0.0
10^{-1}	10^{-3}	10^{-2}	10^6	10.	1	12	4.8×10^{-4}	0.	0.0
10^{-1}	10^{-4}	10^{-2}	10^6	10.	1	16	3.1×10^{-5}	0.	0.0
10^{-1}	10^{-5}	10^{-2}	10^6	10.	1	19	3.8×10^{-6}	0.	0.0
10^{-1}	10^{-6}	10^{-2}	10^6	10.	1	22	4.8×10^{-7}	0.	0.0

on 1D functions is measured by its precision and compute time. We have studied the mean error of the approximate jump points x_i^a and the mean error of the magnitude j_i^a (3) of their corresponding approximate jumps. If the function has NJ jump points, we define

$$E_x \equiv \frac{\sum_{i=1}^{NJ} |x_i^e - x_i^a|}{NJ},$$
$$E_j \equiv \frac{\sum_{i=1}^{NJ} |j_i^e - j_i^a|}{NJ}.$$

We denote by NI the number of intervals of the resulting partition.

4.1.1 Distinction Between Jumps and Steep Gradients

We study the behavior of EDAS1 when applied to the Heaviside function

$$H(x) = \begin{cases} 0, & \text{if } x \leq 0, \\ 1, & \text{if } x > 0, \end{cases}$$

and compare with that corresponding to the function

$$s(x) = \frac{1}{1 + e^{-500x}}.$$

We assume that both functions are defined on $[-1, 1]$; see Fig. 1.

Table 1 shows that EDAS-1 can always find the jump point $x = 0$. E_x decreases with E_2. The number of intervals of the partition is moderate even in high precision cases. The large value of E_4 implies that subdivisions of the intervals based only on their size, are not performed.

Table 2 shows that EDAS-1 only finds jumps near $x = 0$ when E_2 is relatively large. If E_2 decreases it does not find jumps even if we search for jumps of very small magnitude ($E_3 = 10^{-4}$).

Table 2 Performance of EDAS-1 on the function s

E_1	E_2	E_3	E_4	E_5	NJ	NI	CPU (s)
10^{-2}	10^{-2}	10^{-2}	10^6	10.	2	16	0.0
10^{-2}	5×10^{-3}	10^{-2}	10^6	10.	4	18	0.0
10^{-2}	10^{-3}	10^{-2}	10^6	10.	0	22	0.0
10^{-2}	10^{-4}	10^{-2}	10^6	10.	0	22	0.0
10^{-2}	10^{-5}	10^{-2}	10^6	10.	0	22	0.0
10^{-2}	10^{-3}	10^{-4}	10^6	10.	0	22	0.0
10^{-2}	10^{-4}	10^{-4}	10^6	10.	0	22	0.0
10^{-2}	10^{-5}	10^{-4}	10^6	10.	0	22	0.0

Table 3 Jump discontinuities of f

x	j
$-3\pi/4$	1.707107
$-\pi/2$	1.000000
$\pi/4$	0.414214
$\pi/8$	0.847759
$3\pi/8$	2.234633
$7\pi/8$	1.152241

4.1.2 A Function with Several Jumps

We have considered the function [18]

$$f(x) = \begin{cases} 0, & \text{if } -3 \le x < -3\pi/4, \\ 1 + \cos(3x), & \text{if } -3\pi/4 \le x < -\pi/2, \\ 0, & \text{if } -\pi/2 \le x < -\pi/4, \\ 1 - 2\cos(5x + 3\pi/2), & \text{if } -\pi/4 \le x < \pi/8, \\ 0, & \text{if } \pi/8 \le x < 3\pi/8, \\ 2\cos(x - 3\pi/4) - 3, & \text{if } 3\pi/8 \le x < 7\pi/8, \\ 0, & \text{if } 7\pi/8 \le x < 3. \end{cases}$$

Figure 2(a) shows the graph of f. Its jump points and the magnitude of their corresponding jumps are reported in Table 3.

The performance of EDAS-1 on f is summarized in Table 4. From the results we can state

- The obtained results depend little on E_1. It suffices that it is less than the quotient of E_3 and four, such as it is stated in Proposition 3. *NI* decreases as E_1 grows. *NI* is moderate even for high precision results.
- Figure 2(b) shows the resultant piecewise affine function L. It is remarkable the absence of oscillations in it. In this case L is a relatively accurate approximant; see Figs. 2(c) and 2(d).
- The choice of a small E_2 allows to obtain arbitrary precision in the jump points and in the magnitude of their corresponding jumps.

Table 4 Performance of EDAS-1 on the function f

E_1	E_2	E_3	E_4	E_5	NJ	NI	E_x	E_j	CPU (s)
5×10^{-2}	10^{-3}	0.4	10^6	10.	6	79	1.8×10^{-4}	8.7×10^{-4}	0.0
5×10^{-2}	10^{-5}	0.4	10^6	10.	6	121	1.5×10^{-6}	7.1×10^{-6}	0.0
5×10^{-2}	10^{-8}	0.4	10^6	10.	6	181	1.8×10^{-9}	5.3×10^{-9}	0.0
5×10^{-2}	10^{-3}	0.6	10^6	10.	5	79	2.0×10^{-4}	7.0×10^{-4}	0.0
5×10^{-2}	10^{-5}	0.6	10^6	10.	5	121	1.5×10^{-6}	6.2×10^{-6}	0.0
5×10^{-2}	10^{-8}	0.6	10^6	10.	5	181	1.7×10^{-9}	5.6×10^{-9}	0.0
5×10^{-2}	10^{-3}	0.9	10^6	10.	4	79	1.8×10^{-4}	7.6×10^{-4}	0.0
5×10^{-2}	10^{-5}	0.9	10^6	10.	4	121	1.3×10^{-6}	7.0×10^{-6}	0.0
5×10^{-2}	10^{-8}	0.9	10^6	10.	4	181	1.5×10^{-9}	6.8×10^{-9}	0.0
5×10^{-2}	10^{-3}	1.1	10^6	10.	3	79	2.0×10^{-4}	7.7×10^{-4}	0.0
5×10^{-2}	10^{-5}	1.1	10^6	10.	3	121	1.7×10^{-6}	6.8×10^{-6}	0.0
5×10^{-2}	10^{-8}	1.1	10^6	10.	3	181	1.4×10^{-9}	4.5×10^{-9}	0.0
5×10^{-2}	10^{-3}	1.5	10^6	10.	2	79	1.8×10^{-4}	1.1×10^{-3}	0.0
5×10^{-2}	10^{-5}	1.5	10^6	10.	2	121	1.2×10^{-6}	8.1×10^{-6}	0.0
5×10^{-2}	10^{-8}	1.5	10^6	10.	2	181	1.7×10^{-9}	6.0×10^{-9}	0.0
5×10^{-2}	10^{-3}	2.0	10^6	10.	1	79	3.3×10^{-6}	6.8×10^{-4}	0.0
5×10^{-2}	10^{-5}	2.0	10^6	10.	1	121	4.8×10^{-7}	6.1×10^{-6}	0.0
5×10^{-2}	10^{-8}	2.0	10^6	10.	1	181	2.0×10^{-9}	9.0×10^{-9}	0.0

– We have tested the discrimination power of the algorithm by giving different values to E_3. The number of jump points decreases as E_3 grows.

4.1.2.1 Function f with a Periodic Continuous Perturbation We have studied the performance of the algorithm when f is perturbed by the periodic function $p(x) = k\sin(100000x)$, where k is a positive real number. We have applied EDAS-1 to the function $g_p(x) = f(x) + p(x)$. Table 5 shows the obtained results.

From the results, we can conclude

– *NI* is large due to the complexity of the function. This increases the compute time.
– The choice of a small E_2 allows to obtain arbitrary precision in the value of the jump points. The error in the magnitude of the corresponding jumps is greater than that of the unperturbed case, due to the highly oscillatory behavior of $p(x)$.
– Figure 3 shows a detail of the piecewise affine function L. In this case L is a very poor approximant, nevertheless the algorithm is able to find the jump points. Cubature procedures perform an "implicit smoothing" of the function away from the jump discontinuity set. That is, cubature formulas give the exact result for a polynomial interpolant of a certain degree, not the exact value corresponding to the target function.

4.1.2.2 Function f with a random continuous perturbation We have studied the performance of EDAS-1 when f is perturbed by the random function $p(x)$ defined as the piecewise affine function such that $p(x_i) = y_i$, $i = 0, \ldots, np$, where:

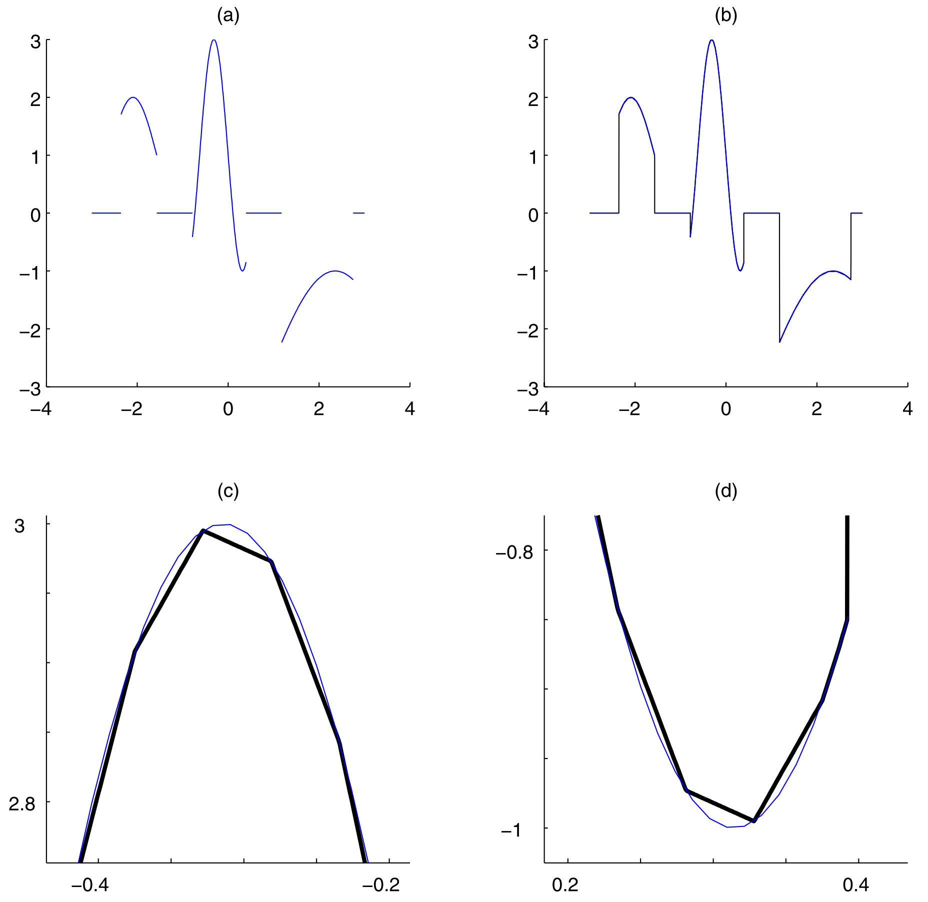


Fig. 2 Performance of EDAS-1 on f ($E_1 = 10^{-2}$, $E_2 = 10^{-3}$, $E_3 = 10^{-1}$, $NI = 95$). (**a**) Graph of f. (**b**) Graph of f (*blue*) and L (*black*). (**c**)–(**d**) Details of the approximation of f by L

- $x_i = a + (b - a)i/np$, $i = 0, \dots, np$.
- $y_i = k\,\mathrm{sgn}(i)\, r(i)$, $i = 0, \dots, np$, where sgn is a random function with values -1 or 1, r is a random function with values in [0, 1], and k is a positive real number.

We have applied EDAS-1 to the function $g_r(x) = f(x) + p(x)$.

The results obtained are similar to those in the previous case. To sum up, we can affirm that periodic or random continuous perturbations do not seem to affect the effectiveness of EDAS-1.

4.2 Computational Experiments on 2D Images

An image is defined as an $n \times m$ matrix I. In the 256 grayscale the entries of I are numbers between 1 (black) and 256 (white).

Table 5 Performance of EDAS-1 on the function g_p (periodic perturbation of f)

k	E_1	E_2	E_3	E_4	E_5	NJ	NI	E_x	E_j	CPU (s)
0.15	5×10^{-2}	10^{-5}	10^{-1}	10^{6}	10.	6	323729	1.5×10^{-6}	8.3×10^{-2}	2.6
0.15	5×10^{-2}	10^{-7}	10^{-1}	10^{6}	10.	6	323765	2.0×10^{-8}	1.3×10^{-3}	2.6
0.15	5×10^{-2}	10^{-9}	10^{-1}	10^{6}	10.	6	323807	2.0×10^{-10}	1.0×10^{-5}	2.6
0.50	5×10^{-2}	10^{-5}	10^{-1}	10^{6}	10.	6	590155	1.4×10^{-6}	2.8×10^{-1}	4.5
0.50	5×10^{-2}	10^{-7}	10^{-1}	10^{6}	10.	6	590191	2.0×10^{-8}	4.5×10^{-3}	4.5
0.50	5×10^{-2}	10^{-9}	10^{-1}	10^{6}	10.	6	590233	2.0×10^{-10}	3.5×10^{-5}	4.5
1.00	5×10^{-2}	10^{-5}	10^{-1}	10^{6}	10.	6	883963	1.5×10^{-6}	5.1×10^{-1}	6.4
1.00	5×10^{-2}	10^{-7}	10^{-1}	10^{6}	10.	6	883999	2.0×10^{-8}	8.9×10^{-3}	6.4
1.00	5×10^{-2}	10^{-9}	10^{-1}	10^{6}	10.	6	884041	2.0×10^{-10}	7.0×10^{-5}	6.3

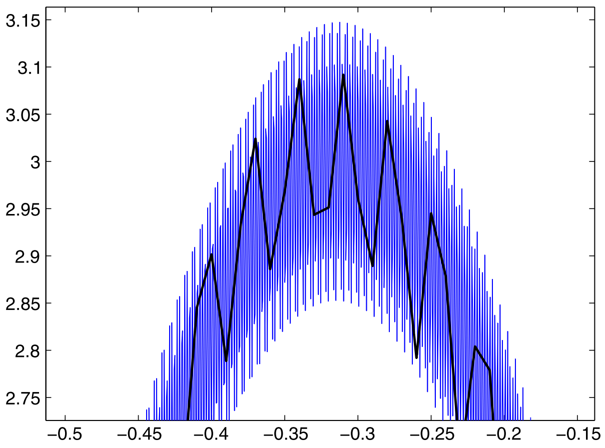


Fig. 3 Performance of EDAS-1 on the periodically perturbed function f ($k = 0.15$, $E_1 = 10^{-3}$, $E_2 = 10^{-3}$, $E_3 = 4\times10^{-1}$, $NI = 8192$). Details of the graphs of g_p (*blue*) and L (*black*)

From this matrix I, one can build a piecewise constant function $\widetilde{I}$, defined on the rectangle $R = [0.5, n + 0.5] \times [0.5, m + 0.5]$, in the following way.

$$\widetilde{I}(x_1,x_2) = \begin{cases} I_{11}, & \text{if } (x_1,x_2) \in [0.5, 1.5] \times [0.5, 1.5], \\ I_{i1}, & \text{if } (x_1,x_2) \in (i-0.5, i+0.5] \times [0.5, 1.5] \ \ (i > 1), \\ I_{1j}, & \text{if } (x_1,x_2) \in [0.5, 1.5] \times (j-0.5, j+0.5] \ \ (j > 1), \\ I_{ij}, & \text{if } (x_1,x_2) \in (i-0.5, i+0.5] \times (j-0.5, j+0.5] \ \ (i, j > 1). \end{cases}$$

In image processing, it is usual consider a special coordinate system in the plane. The positive x_2-axis is the standard positive x_1-axis and the positive x_1-axis is the standard positive x_2-axis oriented downward.

In this section we apply EDAS-1 and EDAS-2 to piecewise constant functions generated by images. We use the following procedures:

– The first method consists of applying EDAS-1 to search for intervals containing jump points, in the following $n + m$ one-dimensional problems:

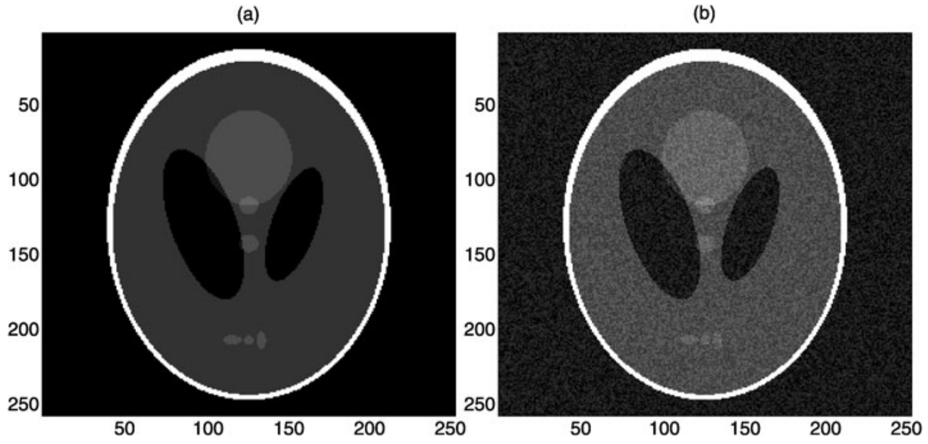


Fig. 4 (**a**) Original Shepp-Logan phantom image. (**b**) Perturbed Shepp-Logan phantom image ($k = 50$)

- $\widetilde{I}(i, x_2)$, $\quad i = 1, \ldots, n$, in $[0.5, m + 0.5]$.
- $\widetilde{I}(x_1, j)$, $\quad j = 1, \ldots, m$, in $[0.5, n + 0.5]$.

  Considering the midpoint of each one of such intervals, we obtain the following set of approximate jump points. (i, x_{2k}), $i = 1, \ldots, n$, $k = 1, \ldots, J(i)$ and (x_{1l}, j), $j = 1, \ldots, m$, $l = 1, \ldots, J(j)$. From this set, we can build a binary $n \times m$ matrix $\widehat{I}$ by means of the following pixelation process. Consider each one of the nm squares of side length 1 contained into R. If one or several approximate jump points belong to the ij square, we define $\widehat{I_{ij}} = 1$ and $\widehat{I_{ij}} = 256$ otherwise. In this way the matrix $\widehat{I}$ contains an approximate representation of the edges of the initial image. We call *TNI* the total number of intervals and *TNJI* the total number of intervals containing jumps, generated in the $n + m$ applications of EDAS-1.
- The second method divides R into two triangles with vertices: $(0.5, 0.5)$, $(n + 0.5, 0.5)$, $(n + 0.5, m + 0.5)$, and $(0.5, 0.5)$, $(n + 0.5, m + 0.5)$, $(0.5, m + 0.5)$. The initial domain partition P_1 consist of these two triangles (in complex problems the subdivision has been different as we detail later on). Each triangle in P_1 is divided into subtriangles according to EDAS-2. When the algorithm stops we have a tessellation of R consisting of *TNT* triangles. We also have *TNJT* triangles which contain jump points inside them. The binary image $\widehat{I}$ is obtained from the barycenters of such triangles and proceeding as in the first method.

4.2.1 Randomly Perturbed Shepp-Logan Phantom

In this subsection, we consider a well-known medical image, the Shepp-Logan phantom image that is widely used as a benchmark for image reconstruction algorithms. The Shepp-Logan phantom image used in this work is obtained from the MATLAB built-in function, f = phantom (modifiedShepp-Logan); see Fig. 4(a). The image is defined as an 256×256 matrix with only six different entries. We have studied the effectiveness of the proposed algorithms on the Shepp-Logan phantom with a random perturbation. We have

Table 6 Performance of EDAS-1 on the perturbed Shepp-Logan phantom ($k = 50$)

E_1	E_2	E_3	E_4	E_5	*TNI*	*TNJI*	CPU (s)
10^{-3}	10^{-7}	25.	10.	10.	3264248	34512	55.8
10^{-3}	10^{-7}	50.	10.	10.	3264248	1951	55.6

Table 7 Performance of EDAS-2 on the perturbed Shepp-Logan phantom ($k = 50$)

E_1	E_2	E_3	E_4	E_5	*TNT*	*TNJT*	CPU (s)
10^{-1}	10^{-1}	25.	10.	10.	8806457	1102587	306.5
10^{-1}	10^{-1}	50.	10.	10.	8806457	62083	306.4

considered the matrix $G = I + k * rand(256)$, where I is the matrix of the original image and $rand(n)$ is the MATLAB function that generates a random $n \times n$ matrix with entries between 0 and 1. We have studied different values of k. In this subsection we give the results obtained with $k = 50$; see Fig. 4(b). In the case of EDAS-2 and due to RAM limitations, we have divided the domain $[0.5, 256.5] \times [0.5, 256.5]$ into 4 squares of the same size. Then, each one of these squares has been divided into four triangles by means of its diagonals. The resulting set of 16 triangles has been considered as the initial domain partition P_1.

The results obtained with EDAS-1 and EDAS-2 are reported in Tables 6 and 7, respectively. Since the perturbation is discontinuous, the number of jump points grows considerably. Figures 5(a) and 5(c) show the capability of EDAS-1 and EDAS-2 to detect them. The only way to avoid the noise consists of increasing E_3; see Figs. 5(b) and 5(d).

4.2.2 Thin "A" Image

This is a binary 218×262 image; see Fig. 6. We use it to test the capability of EDAS-1 and EDAS-2 to find scarce and close jump discontinuities in large domains. Thin "A" image is composed of lines which are only two pixels wide ($x = 50$, 51, 150, 151, and $y = 91$, 92, 170, 171).

Figure 7 shows the results obtained by EDAS-1 and EDAS-2 without the pixelation process.

4.2.3 Lena Image

This 512×512 image has become a standard test; see Fig. 8. Its complexity degree is considerably greater than that of previous examples because I has many different entries that are irregularly distributed.

After several experiments we have selected the value $E_3 = 13$ as the one that seems to maximize the "description level" of Γ_{E_3} and minimizes the noise.

The data obtained with EDAS-1 are reported in Table 8.

In the case of EDAS-2, we have divided the domain $[0.5, 512.5] \times [0.5, 512.5]$ into 16 squares of the same size. Then, each one of these squares has been divided into four triangles

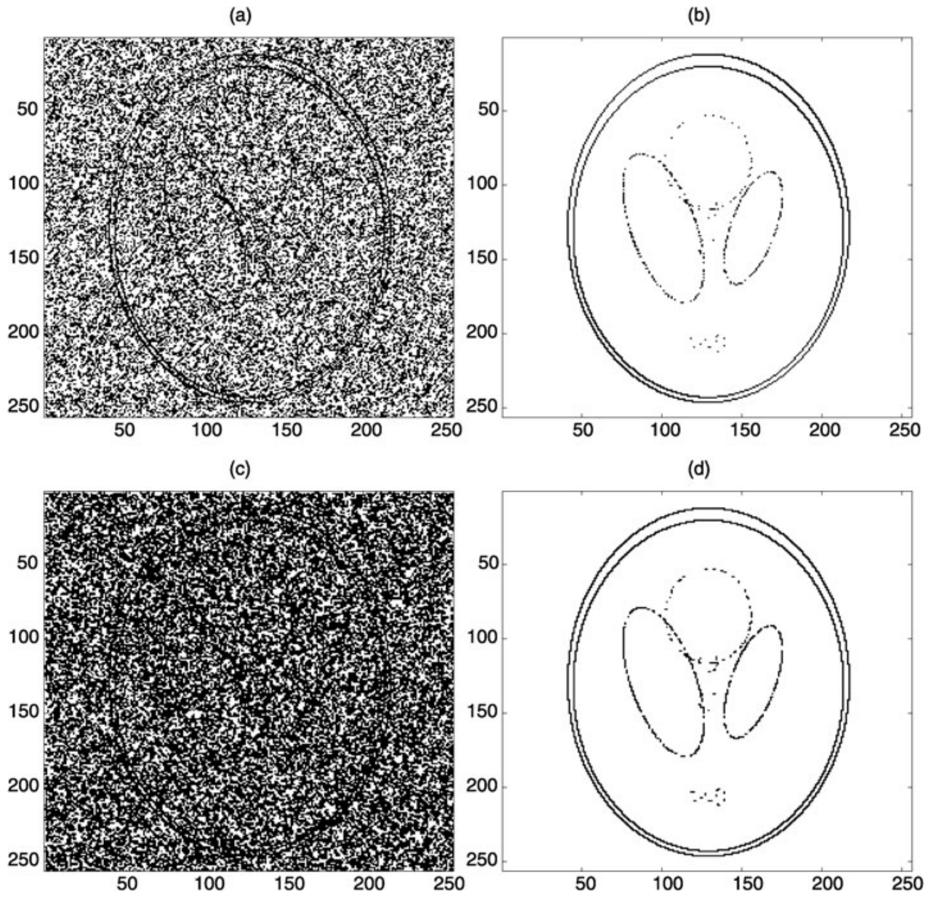


Fig. 5 Edges of the perturbed Shepp-Logan phantom image ($k = 50$). (**a**) Γ^J_{25} approximated by EDAS-1. (**b**) Γ^J_{50} approximated by EDAS-1. (**c**) Γ^J_{25} approximated by EDAS-2. (**d**) Γ^J_{50} approximated by EDAS-2

by means of its diagonals. The resulting set of 64 triangles has been considered as the initial domain partition P_1. Table 9 shows the results obtained with EDAS-2.

Figures 9(a) and 9(b) show the graphs of the approximations of Γ^J_{13} obtained by EDAS-1 and EDAS-2, respectively.

4.2.4 Randomly Perturbed Lena Image

We have studied the effectiveness of the proposed methods on the Lena image with a random perturbation. We have considered the matrix $G = I + k * rand(512)$, where I is the matrix of the original Lena image. We have studied different values of k. In this subsection we give the results obtained with $k = 25$. In the case of EDAS-2, the initial partition P_1 has been that in Sect. 4.2.3.

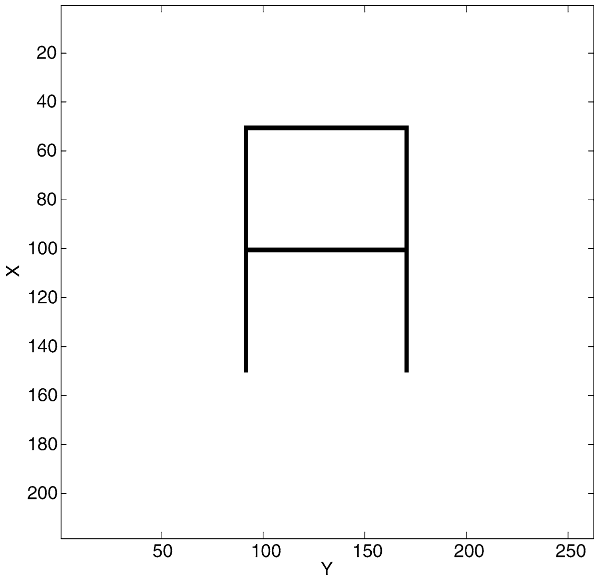


Fig. 6 Original thin "A" image

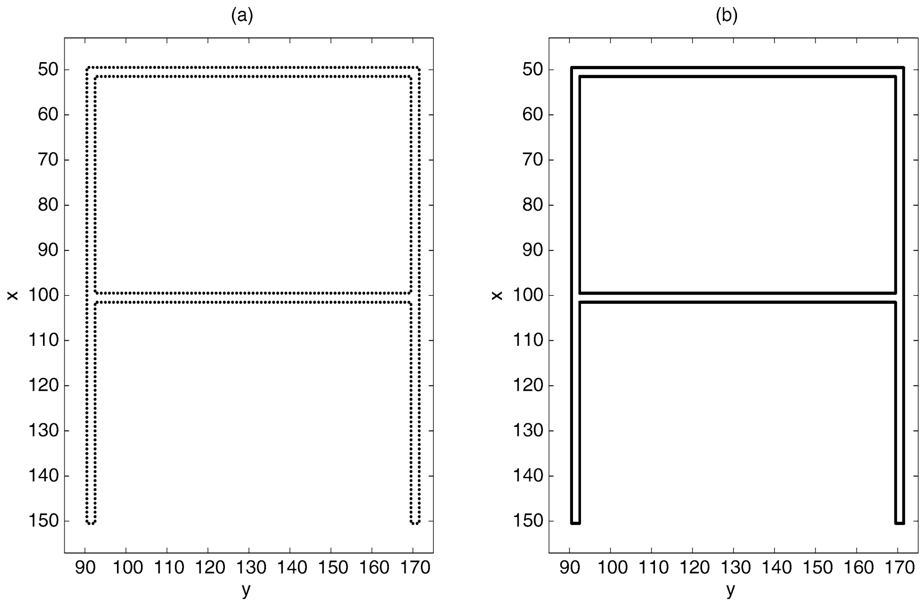
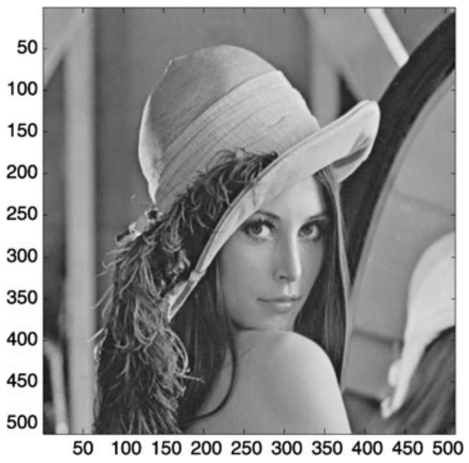


Fig. 7 Thin "A" image. (**a**) Γ^J approximated by EDAS-1 ($E_1 = 10^{-3}$, $E_2 = 10^{-7}$, $E_3 = 250.$). (**b**) Γ^J approximated by EDAS-2 ($E_1 = 10^{-2}$, $E_2 = 5 \times 10^{-2}$, $E_3 = 250.$)

The results obtained with EDAS-1 and EDAS-2 are reported in Tables 10 and 11, respectively. Since the perturbation is discontinuous, the number of jump points grows considerably. Figures 10(a) and 10(c) show the capability of EDAS-1 and EDAS-2 to detect them. The only way to avoid the noise consists of increasing E_3; see Figs. 10(b) and 10(d).

Fig. 8 Original Lena image



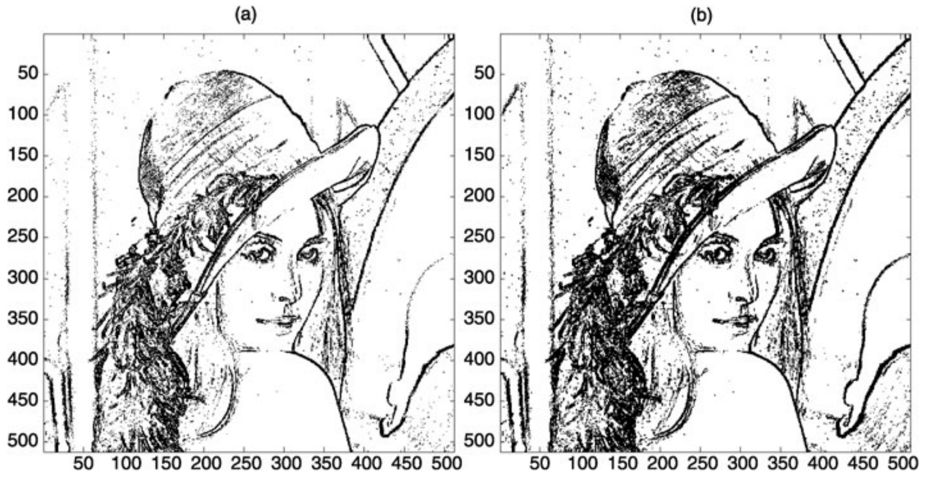


Fig. 9 Edges of Lena image. (**a**) Γ^J_{13} approximated by EDAS-1 ($E_1 = 5 \times 10^{-2}$, $E_2 = 10^{-3}$). (**b**) Γ^J_{13} approximated by EDAS-2 ($E_1 = 10^{-1}$, $E_2 = 10^{-1}$)

4.2.5 Analysis of Image Experiments

- EDAS-1 and EDAS-2 are able to find a good approximation of the jump discontinuity set Γ^J of the piecewise constant function associated with an image. While EDAS-1 restricts its search to two orthogonal straight lines passing through each pixel, EDAS-2 performs an exhaustive search in the whole square corresponding to the pixel. This makes EDAS-2 slow in comparison with EDAS-1.
- The absence of oscillations makes possible to obtain accurate values of the magnitude of the jumps. This allows to obtain stratified edges (Γ^J_l, Γ^J_{lu}) in a straightforward manner.
- Continuous noise does not change the set Γ^J of a function. It increases the computational effort but it does not affect the result of EDAS-*d* (Sect. 4.1.2). On the contrary, discontinuous noise can change Γ^J drastically. Experiments in Sects. 4.2.1 and 4.2.4 confirm this fact.

Table 8 Performance of EDAS-1 on the Lena image

E_1	E_2	E_3	E_4	E_5	*TNI*	*TNJI*	CPU (s)
10^{-3}	10^{-7}	13.	10.	10.	11867770	48689	292.9
5×10^{-2}	10^{-3}	13.	10.	10.	5247716	48689	94.8
10^{-1}	10^{-2}	13.	10.	10.	3829133	48689	71.5
10^{-1}	5×10^{-2}	13.	10.	10.	2883411	48689	56.0

Table 9 Performance of EDAS-2 on the Lena image

E_1	E_2	E_3	E_4	E_5	*TNT*	*TNJT*	CPU (s)
10^{-1}	10^{-1}	13.	10.	10.	32549881	1556466	1128.2

Table 10 Performance of EDAS-1 on the perturbed Lena image ($k = 25$)

E_1	E_2	E_3	E_4	E_5	*TNI*	*TNJI*	CPU (s)
5×10^{-2}	10^{-2}	13.	10.	10.	4163939	164011	78.2
5×10^{-2}	10^{-2}	28.	10.	10.	4163939	21804	76.8

Table 11 Performance of EDAS-2 on the perturbed Lena image ($k = 25$)

E_1	E_2	E_3	E_4	E_5	*TNT*	*TNJT*	CPU (s)
10^{-1}	10^{-1}	15.	10.	10.	35109547	4116009	1221.9
10^{-1}	10^{-1}	31.	10.	10.	35109547	527207	1221.8

4.3 Comparison with other Edge Detection Methods

In this section we compare the results obtained by EDAS-d ($d = 1, 2$) with those obtained by the Canny and Sobel edge detectors (see Sect. 1.1). We have used the MATLAB EDGE (‘Canny’/‘Sobel’) function with automatic choice of parameters. The four algorithms have been tested with the synthetic images obtained in the following way. Consider the function defined on $[-4, 4]^2$

$$f(x,y) = \begin{cases} x^2 + y^2, & \text{if } 0 \le \sqrt{x^2+y^2} < 1, \\ 1 + x^2 + y^2 + 0.1 * \sin(x), & \text{if } 1 \le \sqrt{x^2+y^2} < 2, \\ 2 + x^2 + y^2 + 0.1 * \sin(y), & \text{otherwise.} \end{cases}$$

Notice that f is not constant on sets of Lebesgue measure different from zero. From it, define the functions:

$$g(\bar{x}, \bar{y}) = f(x, y)$$
$$i(\bar{x}, \bar{y}) = Ag(\bar{x}, \bar{y}) + B$$

obtained by the following change of variables

$$\begin{pmatrix} x \\ y \end{pmatrix} = a \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix} + b \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

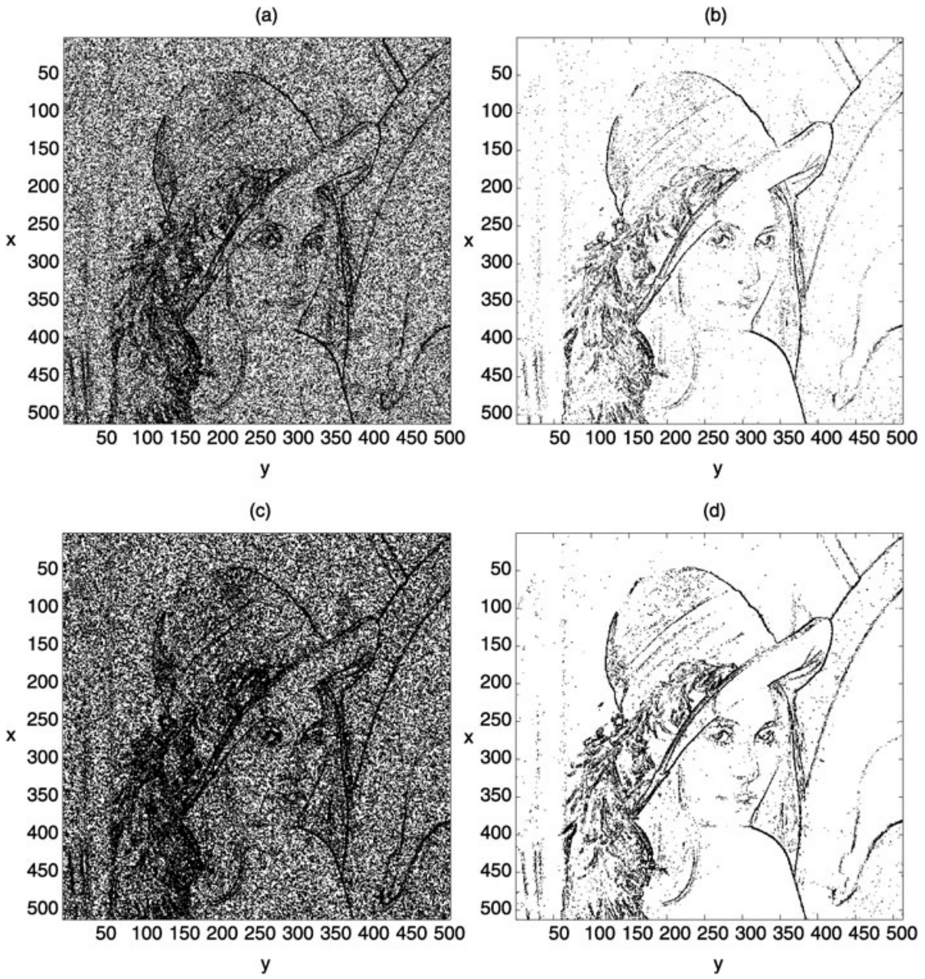


Fig. 10 Edges of the perturbed Lena image approximated by EDAS-1. (**a**) Γ^J_{13} ($E_1 = 5 \times 10^{-2}$, $E_2 = 10^{-2}$). (**b**) Γ^J_{28} ($E_1 = 5 \times 10^{-2}$, $E_2 = 10^{-2}$). Edges of the perturbed Lena image approximated by EDAS-2. (**c**) Γ^J_{15} ($E_1 = 10^{-1}$, $E_2 = 10^{-1}$). (**d**) Γ^J_{31} ($E_1 = 10^{-1}$, $E_2 = 10^{-1}$)

If $a = 4/125$, $b = -502/125$, $A = 51/7$, and $B = 1$, then the continuous function $i(\bar{x}, \bar{y})$ is defined on $[0.5, 250.5] \times [0.5, 250.5]$ and its range is contained in the interval $[1, 256]$.

Proceeding as in Sect. 4.2 we obtain an 250×250 image I defined by the values of i at the node points of a rectangular grid. The corresponding piecewise constant function $\tilde{I}$ is defined on the square $[0.5, 250.5] \times [0.5, 250.5]$. Image I is shown in Fig. 11(a). We have considered the matrices $I_k = I + k * rand(250)$, for $k = 0, 10$. Figure 11(b) shows I_{10}.

The results obtained with EDAS-1 and EDAS-2 are reported in Tables 12 and 13, respectively. The four algorithms show a very good behavior for $k = 0$. Figure 12 shows their comparative performance for $k = 10$.

The results suggest that EDAS-d ($d = 1, 2$) have a random noise elimination behavior comparable to that of the best spatial filtering methods.

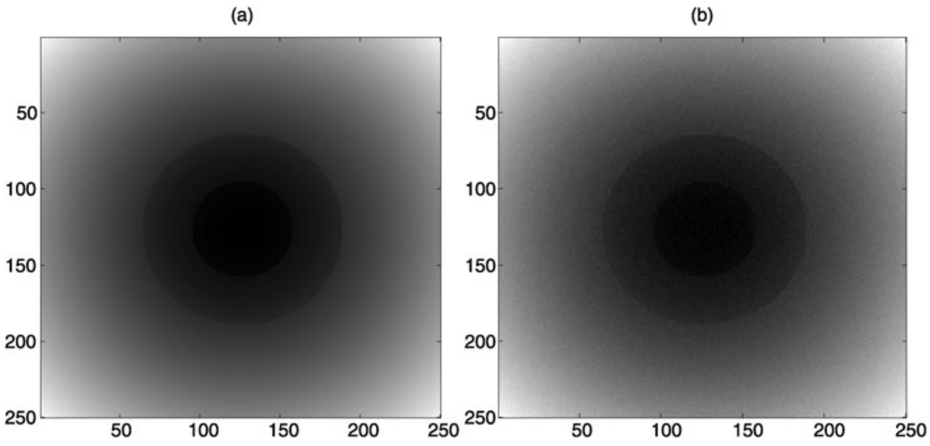


Fig. 11 (**a**) Original synthetic image. (**b**) Randomly perturbed synthetic image ($k = 10$)

Table 12 Performance of EDAS-1 on the images I_k

image	E_1	E_2	E_3	E_4	E_5	*TNI*	*TNJI*	CPU (s)
I_0	10^{-3}	10^{-7}	6.	10.	10.	3108512	744	106.1
I_{10}	10^{-2}	10^{-5}	12.	10.	10.	2228804	133	76.6

Table 13 Performance of EDAS-2 on the images I_k

image	E_1	E_2	E_3	E_4	E_5	*TNT*	*TNJT*	CPU (s)
I_0	10^{-1}	5×10^{-2}	6.	10.	10.	15366102	48374	1294.4
I_{10}	10^{-1}	10^{-1}	13.	10.	10.	8263157	3462	713.8

4.4 Computational Experiments on General Piecewise Continuous 2D Functions

In the case of general piecewise continuous functions defined on a rectangle with non-convex sets in its partition, results in Sect. 2 are not directly applicable. This is due to the complexity of the jump discontinuity set Γ^J. For example, if Γ^J is curved, the intersection of a given triangle and Γ^J is no longer a straight line segment. In spite of this fact, EDAS-2 can be applied to solve some of these problems as we see below.

Consider the function f defined in Sect. 4.3. Its Γ^J is a set of two concentric circumferences.

The data obtained with EDAS-2 are reported in Table 14.

Figures 13(a) and 13(b) show the grid generated by EDAS-2 and the approximation of Γ^J, respectively.

5 Concluding Remarks

In this paper we propose an algorithm (EDAS-d) to approximate the jump discontinuity set of functions defined almost everywhere on convex subsets of $\mathbb{R}^d$. EDAS-d can be applied to

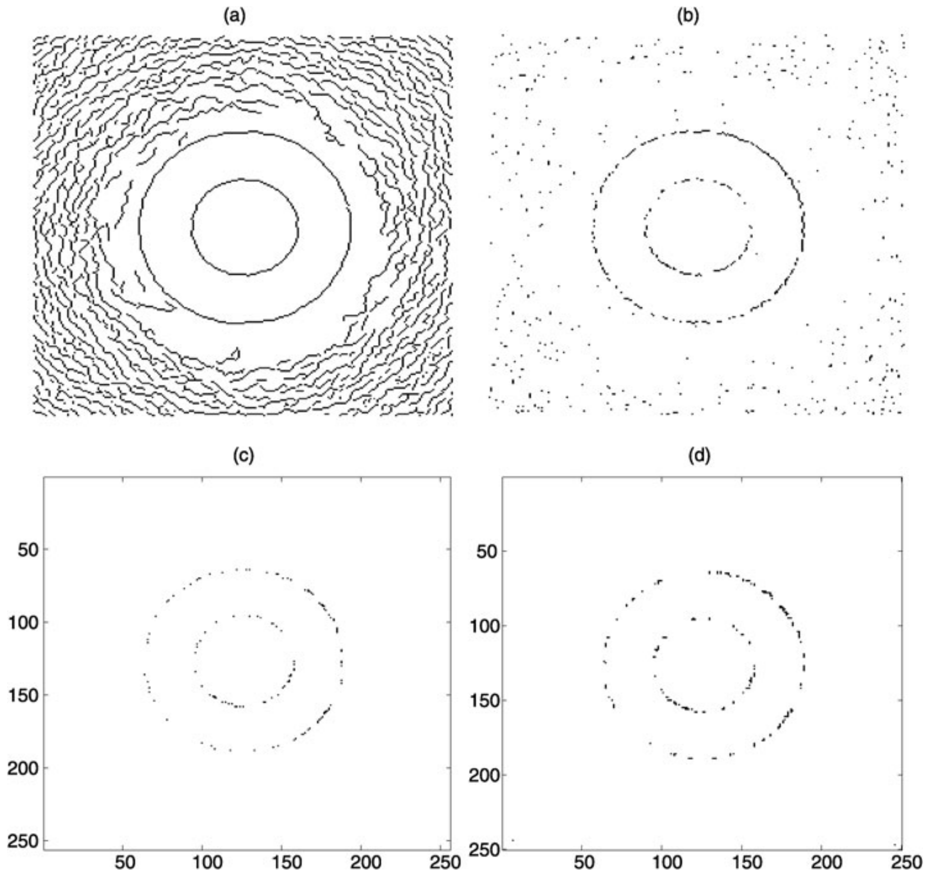


Fig. 12 Randomly perturbed synthetic image ($k = 10$): (**a**) Edges obtained by the Canny detector. (**b**) Edges obtained by the Sobel detector. (**c**) Γ^J_{12} approximated by EDAS-1. (**d**) Γ^J_{13} approximated by EDAS-2

Table 14 Performance of EDAS-2 on the function g

E_1	E_2	E_3	E_4	E_5	*TNT*	*TNJT*	CPU (s)
5×10^{-2}	10^{-2}	10^{-1}	10.	10.	18560	7416	0.3

discrete functions extending their domain by interpolation and using the procedure described in Sect. 4.2. The method is based on adaptive splitting of the domain of the function guided by the value of an average integral. The numerical computation of these integrals introduces an "implicit smoothing" that allows a fast convergence away from the jump points. The algorithm needs to specify five parameters which have an intuitive meaning. They can be easily fixed in most problems. We have studied from a theoretical and computational point of view the cases $d = 1$ and $d = 2$. In the case of piecewise continuous 1D functions and piecewise constant 2D functions, we can draw the following conclusions:

– EDAS-d provides a precise determination of the jump points.
– The resulting piecewise affine function L does not show oscillatory behavior near the jump points. This makes possible to obtain accurate values of the magnitude of the jumps.

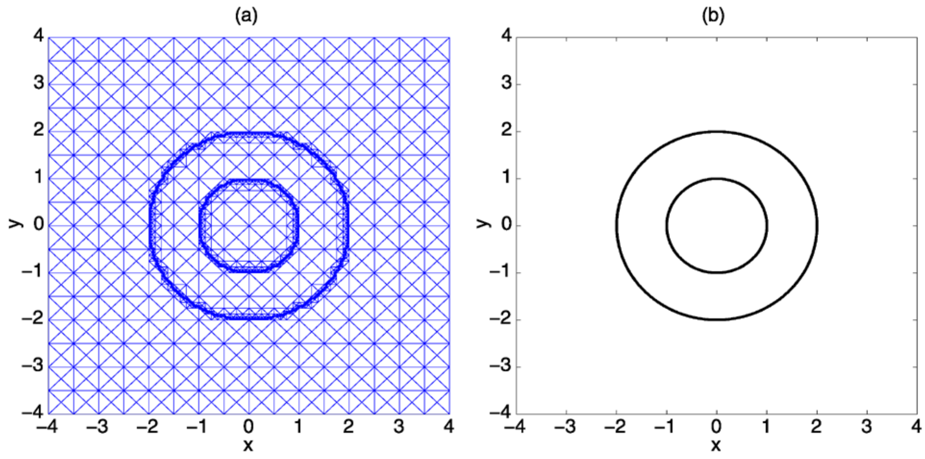


Fig. 13 Performance of EDAS-2 on a function with curved jump discontinuity set. (**a**) Grid generated by EDAS-2. (**b**) Γ^J of f approximated by EDAS-2

As a consequence, we can obtain stratified edges (Γ_l^J, Γ_{lu}^J) in a straightforward manner. In the case of images, this fact is important because stratified edges correspond to "individual objects" in a scene.

– EDAS-d allows to discriminate between real jumps (discontinuities) and steep gradients.
– EDAS-d is robust against continuous perturbations. Discontinuous perturbations change the set Γ^J of the unperturbed function. This affects the results of the algorithm, due to its high sensitivity to detect jumps. In the case of random perturbation of images, the experimental results suggest that EDAS-d has a noise elimination behavior comparable to that of the best spatial filtering methods.
– In all computational experiments, EDAS-d runs quite fast. EDAS-1 is preferable to EDAS-2 when fast image processing is required.

EDAS-2 has proved its computational effectiveness in some problems involving general piecewise continuous functions with curved Γ^J.

A drawback of EDAS-d is the approximate method of evaluation of integrals that can lead to the oversight of some jump points. This can be avoided by decreasing the exploration parameter E_4 or by modifying the adaptive procedure to compute integrals.

Future work should address the analysis of EDAS-d when $d > 2$. It would also be interesting to study theoretically the global behavior of the algorithm.

Acknowledgements We would like to thank the anonymous reviewers for their careful reading of our manuscript and the insightful comments they provided.

Appendix: Proofs of Proposition 3 and Theorem 2

Proposition 3 *Given* $\varepsilon > 0$ *there exists a number* $\delta > 0$ *such that*

$$\frac{1}{2\eta}\int_{-\eta}^{\eta} |f(x) - L_{T(\eta)} f(x)| dx > \frac{|a_0 - b_0|}{4} - \varepsilon$$

for all η *such that* $0 < \eta < \delta$.

Proof

$$\begin{aligned}\frac{1}{2\eta}\int_{-\eta}^{\eta}|f(x)-L_{T(\eta)}f(x)|dx &= \frac{1}{2\eta}\int_{0}^{\eta}|f(x)-L_{T(\eta)}f(x)|dx\\ &\quad+\frac{1}{2\eta}\int_{-\eta}^{0}|f(x)-L_{T(\eta)}f(x)|dx\\ &\geq\left|\frac{1}{2\eta}\int_{0}^{\eta}(f(x)-L_{T(\eta)}f(x))dx\right|\\ &\quad+\left|\frac{1}{2\eta}\int_{-\eta}^{0}(f(x)-L_{T(\eta)}f(x))dx\right|.\end{aligned}$$

It suffices to show that the limit of the right hand side of the above inequality as η approaches zero is $|a_0-b_0|/4$.

We can assume that $m=n$ by making as many coefficients as necessary equal to zero. Then $L_{T(\eta)}f(x)=l_1x+l_2$, where

$$l_1=\frac{1}{2\eta}\sum_{k=0}^{n}(a_k+(-1)^{k+1}b_k)\eta^k,$$

$$l_2=\frac{1}{2}\sum_{k=0}^{n}(a_k+(-1)^{k}b_k)\eta^k.$$

Hence

$$\begin{aligned}\frac{1}{2\eta}\int_{0}^{\eta}(f(x)-L_{T(\eta)}f(x))dx &= \frac{1}{2\eta}\int_{0}^{\eta}\left(\sum_{k=0}^{n}a_kx^k-l_1x-l_2\right)dx\\ &=\frac{1}{2\eta}\sum_{k=0}^{n}\frac{a_k\eta^{k+1}}{k+1}-\frac{1}{8}\sum_{k=0}^{n}(a_k+(-1)^{k+1}b_k)\eta^k\\ &\quad-\frac{1}{4}\sum_{k=0}^{n}(a_k+(-1)^{k}b_k)\eta^k,\end{aligned}$$

$$\begin{aligned}\frac{1}{2\eta}\int_{-\eta}^{0}(f(x)-L_{T(\eta)}f(x))dx &= \frac{1}{2\eta}\int_{-\eta}^{0}\left(\sum_{k=0}^{n}b_kx^k-l_1x-l_2\right)dx\\ &=\frac{1}{2\eta}\sum_{k=0}^{n}\frac{(-1)^{k+2}b_k\eta^{k+1}}{k+1}+\frac{1}{8}\sum_{k=0}^{n}(a_k+(-1)^{k+1}b_k)\eta^k\\ &\quad-\frac{1}{4}\sum_{k=0}^{n}(a_k+(-1)^{k}b_k)\eta^k.\end{aligned}$$

And the result follows. □

Theorem 2 *Let $h:\mathbb{R}^2\to\mathbb{R}$ be the function defined by*

$$h(\mathbf{x})\equiv JH(ax_1+bx_2+c),$$

where a, b, c and J are real numbers such that either a or b are distinct from zero and $J > 0$. Define $r \equiv \{\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2 : ax_1 + bx_2 + c = 0\}$ and let T be an arbitrary triangle in $\mathbb{R}^2$ such that $r \cap \mathring{T} \neq \emptyset$. Then

$$\frac{J}{4} \leq \frac{\int_T |h(\mathbf{x}) - L_T h(\mathbf{x})| d\mathbf{x}}{\mathrm{v}(T)} \leq J, \tag{2}$$

where $\mathrm{v}(T)$ is the area of T.

Proof Let $A(a_1, a_2)$, $B(b_1, b_2)$ and $C(c_1, c_2)$ be the vertices of the triangle T. Consider the change of variables given by $\mathbf{x} = \tau(\mathbf{y})$

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 - a_1 & c_1 - a_1 \\ b_2 - a_2 & c_2 - a_2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}.$$

Define

$$M_\tau \equiv \begin{pmatrix} b_1 - a_1 & c_1 - a_1 \\ b_2 - a_2 & c_2 - a_2 \end{pmatrix}.$$

We have that $T = \tau(\Delta)$ where Δ is the triangle in the plane (y_1, y_2) with vertices $P_1(0,0)$, $P_2(1,0)$, and $P_3(0,1)$.

If we use the change of variables formula for multiple integrals and consider that $\mathrm{v}(T) = |\det(M_\tau)|/2$

$$\begin{aligned} AI_T(h) &= \frac{\int_T |h(\mathbf{x}) - L_T h(\mathbf{x})| d\mathbf{x}}{\mathrm{v}(T)} = \frac{\int_\Delta |h(\tau(\mathbf{y})) - L_T h(\tau(\mathbf{y}))| |\det(M_\tau)| d\mathbf{y}}{\mathrm{v}(T)} \\ &= 2 \int_\Delta |h(\tau(\mathbf{y})) - L_T h(\tau(\mathbf{y}))| d\mathbf{y}. \end{aligned} \tag{5}$$

There are two possible cases:

(i) h is equal to J at two vertices of T and is equal to zero at the third vertex.
(ii) h is equal to J at one vertex of T and is equal to zero at the other two vertices.

(i) Assume that h is equal to J at A and C and is equal to zero at B. For finding $L_T h(\tau(\mathbf{y}))$ it suffices to find the affine interpolant L_Δ such that $L_\Delta(P_1) = L_\Delta(P_3) = J$ and $L_\Delta(P_2) = 0$. It is clear that $L_\Delta(\mathbf{y}) = J(1 - y_1)$. Therefore

$$L_T h(\tau(\mathbf{y})) = L_\Delta(\mathbf{y}) = J(1 - y_1),$$

and (5) can be written as

$$AI_T(h) = 2 \int_\Delta |h(\tau(\mathbf{y})) - J(1 - y_1)| d\mathbf{y}.$$

Assume first that $h(\tau(\mathbf{y})) = JH(\alpha - y_1)$, that is

$$h(\tau(\mathbf{y})) = \begin{cases} J, & \text{if } y_1 < \alpha, \\ 0, & \text{if } y_1 \geq \alpha. \end{cases}$$

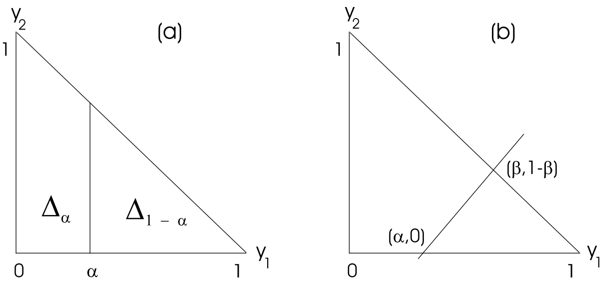


Fig. 14 Illustration for the proof of Theorem 2

If $\alpha \in (0,1)$, define

$$\Delta_\alpha \equiv T \cap \{\mathbf{y} : y_1 \le \alpha\},$$
$$\Delta_{1-\alpha} \equiv T \cap \{\mathbf{y} : y_1 \ge \alpha\},$$
$$AI(\alpha) \equiv 2\int_\Delta |h(\tau(\mathbf{y})) - J(1-y_1)| d\mathbf{y}.$$

Figure 14(a) shows the sets Δ_α and $\Delta_{1-\alpha}$. Then

$$\begin{aligned} AI(\alpha) &= 2\int_{\Delta_\alpha} |J - J(1-y_1)| dy_1 dy_2 + 2\int_{\Delta_{1-\alpha}} |0 - J(1-y_1)| dy_1 dy_2 \\ &= 2J\int_0^\alpha dy_1 \int_0^{1-y_1} y_1 dy_2 + 2J\int_\alpha^1 dy_1 \int_0^{1-y_1} (1-y_1) dy_2 \\ &= \frac{J}{3}(2 - 6\alpha + 9\alpha^2 - 4\alpha^3). \end{aligned}$$

The global minimum of $AI(\alpha)$ in $[0,1]$ is attained at $\alpha = 1/2$, therefore

$$AI(\alpha) \ge AI(1/2) = \frac{J}{4}, \quad \alpha \in (0,1). \tag{6}$$

We must prove that inequality (6) can be extended to the general case

$$h(\tau(\mathbf{y})) = JH((\beta-\alpha)y_2 - (1-\beta)y_1 + \alpha(1-\beta)),$$

where $0 \le \alpha, \beta < 1$. Figure 14(b) shows the straight line

$$(\beta-\alpha)y_2 - (1-\beta)y_1 + \alpha(1-\beta) = 0.$$

There are eight possible cases:

1) $\alpha < 1/2, \quad \alpha < \beta \le 1/2,$

2) $\alpha < 1/2, \quad 0 < \beta < \alpha,$

3) $\alpha < 1/2, \quad 1/2 < \beta < 1,$

4) $\alpha > 1/2, \quad \alpha < \beta < 1,$

5) $\alpha > 1/2, \quad 1/2 \le \beta \le \alpha,$

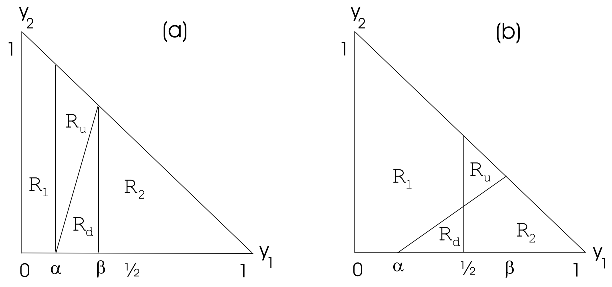


Fig. 15 Illustration for the proof of Theorem 2

6) $\alpha > 1/2, \quad 0 < \beta < 1/2,$

7) $\alpha = 1/2, \quad 1/2 < \beta < 1,$

8) $\alpha = 1/2, \quad 0 < \beta < 1/2.$

Define

$$AI(\alpha,\beta) \equiv 2J \int_\Delta |H((\beta-\alpha)y_2 - (1-\beta)y_1 + \alpha(1-\beta)) - (1-y_1)|dy_1dy_2.$$

It suffices to prove that $AI(\alpha,\beta) \geq J/4$ in the eight cases. We detail only two of them. The remaining cases can be proved similarly.

– Case (1) Using the notation in Fig. 15(a)

$$\begin{aligned} AI(\alpha,\beta) &= 2J\int_{R_1} y_1dy_1dy_2 + 2J\int_{R_u} y_1dy_1dy_2 \\ &\quad + 2J\int_{R_d}(1-y_1)dy_1dy_2 + 2J\int_{R_2}(1-y_1)dy_1y_2, \qquad (7) \\ AI(\beta,\beta) &= 2J\int_{R_1} y_1dy_1y_2 + 2J\int_{R_u} y_1dy_1dy_2 + 2J\int_{R_d} y_1dy_1dy_2 \\ &\quad + 2J\int_{R_2}(1-y_1)dy_1dy_2. \end{aligned}$$

In R_d, $y_1 \leq 1/2$, therefore $y_1 \leq 1-y_1$. Hence

$$\int_{R_d}(1-y_1)dy_1dy_2 \geq \int_{R_d} y_1dy_1dy_2.$$

From this inequality and using (6)

$$AI(\alpha,\beta) \geq AI(\beta,\beta) = AI(\beta) \geq \frac{J}{4}.$$

– Case (3) Using the notation in Fig. 15(b)

$$\begin{aligned} AI(\alpha,\beta) &= 2J\int_{R_1} y_1dy_1dy_2 + 2J\int_{R_u} y_1dy_1dy_2 + 2J\int_{R_d}(1-y_1)dy_1dy_2 \\ &\quad + 2J\int_{R_2}(1-y_1)dy_1dy_2, \end{aligned}$$

$$AI(1/2,1/2) = 2J\int_{R_1} y_1 dy_1 dy_2 + 2J\int_{R_u}(1-y_1)dy_1 dy_2 + 2J\int_{R_d} y_1 dy_1 dy_2 + 2J\int_{R_2}(1-y_1)dy_1 dy_2.$$

In R_u, $y_1 \geq 1/2$, therefore $y_1 \geq 1-y_1$. Hence

$$\int_{R_u} y_1 dy_1 dy_2 \geq \int_{R_u}(1-y_1)dy_1 dy_2.$$

In R_d, $y_1 \leq 1/2$, therefore $y_1 \leq 1-y_1$. Hence

$$\int_{R_d}(1-y_1)dy_1 dy_2 \geq \int_{R_d} y_1 dy_1 dy_2.$$

Consequently

$$AI(\alpha,\beta) \geq AI(1/2,1/2) = AI(1/2) = \frac{J}{4}.$$

In the case (i), since for any $h(\mathbf{x})$ satisfying the assumptions of the proposition, $AI_T(h)$ can be written as $AI(\alpha,\beta)$ for some α and β, we have that the left hand side inequality in (2) is proven.

The proof of the right hand side inequality in (2) can be obtained from any expression of $AI(\alpha,\beta)$, for example (7). Taking into account that $y_1 \leq 1$ and $1-y_1 \leq 1$ in Δ, we have that

$$AI(\alpha,\beta) \leq 2J(\mathrm{v}(R_1)+\mathrm{v}(R_u)+\mathrm{v}(R_d)+\mathrm{v}(R_2)) = 2J\mathrm{v}(\Delta) = J.$$

(ii) In the second case we can assume that h is zero at A and C and is 1 at B.

For finding $L_T h(\tau(\mathbf{y}))$ it suffices to find the affine interpolant L_Δ such that $L_\Delta(P_1) = L_\Delta(P_3) = 0$ and $L_\Delta(P_2) = 1$. It is clear that $L_\Delta(\mathbf{y}) = Jy_1$. Therefore

$$L_T h(\tau(\mathbf{y})) = L_\Delta(\mathbf{y}) = Jy_1.$$

(5) can be written as

$$AI_T(h) = 2\int_\Delta |h(\tau(\mathbf{y})) - Jy_1| d\mathbf{y}.$$

The reasoning to prove that $AI_T(h) \geq J/4$ and $AI_T(h) \leq J$ is similar to that given in case (i). □

References

1. Alvarez, L., Lions, P.L., Morel, J.M.: Image selective smoothing and edge detection by nonlinear diffusion II. SIAM J. Numer. Anal. **29**, 845–866 (1992)
2. Archibald, R., Gelb, A., Gottlieb, S., Ryan, J.: One-sided post-processing for the discontinuous Galerkin method using ENO type stencil choosing and the local edge detection method. J. Sci. Comput. **28**, 167–190 (2006)
3. Archibald, R., Gelb, A., Yoon, J.: Polynomial fitting for edge detection in irregularly sampled signals and images. SIAM J. Numer. Anal. **43**, 259–279 (2005)
4. Bänsch, E., Mikula, K.: A coarsening finite element in image selective smoothing. Comput. Vis. Sci. **1**, 53–61 (1997)

5. Bänsch, E., Mikula, K.: Adaptivity in 3D image processing. Comput. Vis. Sci. **4**, 21–30 (2001)
6. Berntsen, J., Espelid, T.O.: Error estimation in automatic quadrature routines. ACM Trans. Math. Softw. **17**, 233–252 (1991)
7. Bliss, A., Su, F.E.: Lower bounds for simplicial covers and triangulations of cubes. Discrete Comput. Geom. **33**, 669–686 (2005)
8. Bosnjak, A., Montilla, G., Villegas, R., Jara, I.: 3D segmentation with an application of level set-method using MRI volumes for image guided surgery. In: Proceedings of the 29th Annual International Conference of the IEEE EMBS, pp. 5263–5266, Lyon, France, August 23–26 (2007)
9. Catté, F., Lions, P.L., Morel, J.M., Coll, T.: Image selective smoothing and edge detection by nonlinear diffusion. SIAM J. Numer. Anal. **29**, 182–193 (1992)
10. Chen, M.-H., Chin, R.T.: Partial smoothing splines for noisy + boundaries with corners. IEEE Trans. Pattern Anal. Mach. Intell. **15**, 1208–1216 (1993)
11. Chen, G., Yang, Y.H.: Edge detection by regularized cubic B-spline fitting. IEEE Trans. Syst. Man Cybern. **25**, 636–643 (1995)
12. Fujioka, H., Kano, H.: Extrema detection of optimal smoothing splines with application to edge detection problem. In: Proceedings of the 32nd Annual Conference on Industrial Electronics (IECON 2006), pp. 3294–3299. IEEE, New York (2006)
13. Gamelin, Th.W., Greene, R.E.: Introduction to Topology. Dover, New York, Mineola (1999)
14. Gelb, A., Tadmor, E.: Detection of edges in spectral data. Appl. Comput. Harmon. Anal. **7**, 101–135 (1999)
15. Gelb, A., Tadmor, E.: Detection of edges in spectral data II. Nonlinear enhancement. SIAM J. Numer. Anal. **38**, 1389–1408 (2000)
16. Gelb, A., Tadmor, E.: Adaptive edge detectors for piecewise smooth data based on the *minmod* limiter. J. Sci. Comput. **28**, 279–306 (2006)
17. Gelb, A., Cates, D.: Detection of edges in spectral data III-refinement of the concentration method. J. Sci. Comput. **36**, 1–43 (2008)
18. Gelb, A., Cates, D.: Segmentation of images from Fourier spectral data. Commun. Comput. Phys. **5**, 326–34 (2009)
19. Gevers, T., Smeulders, A.W.M.: Combining region splitting and edge detection through guided Delaunay image subdivisions. In: IEEE Proceedings of CVPR, pp. 1021–1026 (1997)
20. González, R.C., Woods, R.E., Eddins, S.L.: Digital Image Processing using MATLAB. Prentice Hall, New York (2004)
21. Gottlieb, D., Shu, Ch.-W.: On the Gibbs phenomenon and its resolution. SIAM Rev. **39**, 644–668 (1997)
22. Grundmann, A., Möller, H.M.: Invariant integration formulas for the n-simplex by combinatorial methods. SIAM J. Numer. Anal. **15**, 282–290 (1978)
23. Guan, X., Guan, Z.: Edge detection of high resolution remote sensing imaginary using wavelet. In: Proceedings of the International Conference on Info-tech and Info-net (ICII 2001), vol. 1, pp. 302–307. IEEE, New York (2001)
24. Horowitz, S.L., Pavlidis, T.: Picture segmentation by a tree traversal algorithm. J. ACM **23**, 368–388 (1975)
25. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. Int. J. Comput. Vis. **1**, 321–331 (1988)
26. Krylov, V.I.: Approximate Calculation of Integrals. Dover, New York (2005)
27. Lizier, M.A.S., Martins, D.C. Jr., Cuadros-Vargas, A.J., Cesar, R.M. Jr., Nonato, L.G.: Generated segmented meshes from textured color images. J. Vis. Commun. Image R. **20**, 190–203 (2009)
28. Llanas, B., Lantarón, S., Sáinz, F.J.: Constructive approximation of discontinuous functions by neural networks. Neural Process. Lett. **27**, 209–226 (2008)
29. Llanas, B., Sáinz, F.J.: Fast training of neural trees by adaptive splitting based on cubature. Neurocomputing **71**, 3387–3408 (2008)
30. Meinhardt, E., Zacur, E., Frangi, A.F., Caselles, V.: 3D edge detection by selection of level surface patches. J. Math. Imaging Vis. **34**, 1–16 (2009)
31. Orden, D., Santos, F.: Asymptotically efficient triangulations of the d-cube. Discrete Comput. Geom. **30**, 509–528 (2003)
32. Paragios, N., Chen, Y., Faugeras, O. (eds.): Handbook of Mathematical Models in Computer Vision. Springer, Berlin (2006)
33. Pavlidis, T., Liow, Y.-T.: Integrating region growing and edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **12**, 225–233 (1990)
34. Perona, P., Malik, J.: Scale space and edge detection using anisotropic diffusion. In: Proceedings of the IEEE Workshop on Computer Vision, Miami, pp. 16–22 (1987)
35. Preußer, T., Rumpf, M.: An adaptive finite element method for large scale image processing. J. Vis. Commun. Image R. **11**, 183–195 (2000)

36. Rumpf, M., Voigt, A., Berkels, B., Rätz, A.: Extracting grain boundaries and macroscopic deformations from images on atomic scale. J. Sci. Comput. **35**, 1–23 (2008)
37. Subbotin, Y.N.: Error of the approximation by interpolation polynomials of small degrees on n-simplices. Math. Notes **48**, 1030–1037 (1990)
38. Tang, Y.Y., Yang, L., Liu, J.: Quadratic spline wavelet approach to automatic extraction of baselines from document images. In: Proceedings of the Fourth International Conference on Document Analysis and Recognition, vol. 2, pp. 693–696. IEEE, New York (1997)
39. Wu, X.: Adaptive split-and-merge segmentation based on piecewise least-square approximation. IEEE Trans. Pattern Anal. Mach. Intell. **15**, 808–815 (1993)