

Simplification of Jacobi Sets

Suthambhara N¹ and Vijay Natarajan^{1,2}

¹ Department of Computer Science and Automation

² Supercomputer Education and Research Centre
Indian Institute of Science, Bangalore, India
email: {suthambhara, vijayn}@csa.iisc.ernet.in

Abstract. The Jacobi set of two Morse functions defined on a 2-manifold is the collection of points where the gradients of the functions align with each other or where one of the gradients vanish. It describes the relationship between functions defined on the same domain, and hence plays an important role in multi-field visualization. The Jacobi set of two piecewise linear functions may contain several components indicative of noisy or a feature-rich dataset. We pose the problem of simplification as the extraction of level sets and offset contours and describe an algorithm to compute and simplify Jacobi sets in a robust manner.

1 Introduction

Motivation. The Jacobi set extends the notion of critical points to multiple functions and helps describe the relationship between multiple scalar functions. Edelsbrunner et al. [1] have shown that the Jacobi sets can be used to compute a comparison measure between two scalar functions. Bennett et al. [2] have used the Jacobi set to represent tunnels and the silhouette of a mesh, both of which are subsequently used to compute a cross parameterization. Jacobi sets have also been used to track features of time-varying events such as molecular interactions, combustion simulation, etc. [3]. All the above applications face a common challenge, namely the presence of degenerate regions and noise in the data. The number of components of the Jacobi set is often more than what can be visually comprehended. So, it is necessary to simplify the Jacobi set. The simplification can be accomplished either using the notion of persistence [4], or otherwise.

Prior work and our approach. In their paper, Bremer et al. [3] have described a method to remove noise in the Jacobi set for time varying data. The persistence of a component of the Jacobi set is the time interval between its birth and death. This measure has been used to remove components that are either noise in the data or unimportant features. Extending this for general functions is nontrivial and hence a more complete approach with guaranteed error bounds is required. We pose the problem of computing Jacobi sets as the computation of a level set of a function defined on the input manifold. Jacobi set simplification now simplifies the level set. We also ensure that the change in relationship between the functions due to simplification does not exceed a given input threshold.

2 Background

Morse Theory. Morse theory studies the relationship between functions and domains. Let \mathbb{M} be a smooth Riemannian 2-manifold. Let f be a smooth function defined on \mathbb{M} and (x_1, x_2) be a local coordinate system such that the unit tangent vectors $(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2})$ form an orthonormal basis with respect to a Riemannian metric. The gradient of f at x is defined as the vector $\nabla f(x) = (\frac{\partial f}{\partial x_1}(x), \frac{\partial f}{\partial x_2}(x))$

A point x is a critical point of f if $\nabla f(x)$ is the zero vector. The function f is called a *Morse function* if the Hessian

$$\mathcal{H}_f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \frac{\partial^2 f}{\partial x_2 \partial x_1}(x) \\ \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) & \frac{\partial^2 f}{\partial x_2^2}(x) \end{bmatrix}$$

is non-singular at all critical points.

Level sets and Reeb graphs. The *level set* at c is defined as the set of all points where f attains the value c : $f^{-1}(c) = \{x \in \mathbb{M} \mid f(x) = c\}$. The *Reeb graph* of f is obtained by contracting connected level set components to points. Nodes in a Reeb graph correspond to critical points of f , see Fig.1. The level sets *sweep* the domain as we increase c over the range of the function f . During a sweep over the domain, the topology of the level set changes at critical points of f . If the sweep is in the direction of increasing function value, level set components are created at minima, they merge or split at saddles, and are destroyed at maxima. Given a sweep direction, saddles may be classified as split or merge saddles depending on the change in the topology of level sets at these points.

Jacobi Sets. The *Jacobi set* of two Morse functions f and g defined on a 2-manifold \mathbb{M} is the collection of points where the gradients of the functions align with each other or one of the gradients vanish. Alternately, the Jacobi set can be described as the collection of critical points of the family of functions $f + \lambda g, \lambda \in \mathbb{R}$:

$$\mathbb{J} = \{x \in \mathbb{M} \mid x \text{ is a critical point of } f + \lambda g \text{ or of } \lambda f + g\}$$

Note that the Jacobi set contains critical points of f and g . Edelsbrunner and Harer [5] used this alternate description to compute Jacobi sets of piecewise linear functions. They also showed that the Jacobi set of two Morse functions is a smoothly embedded 1-manifold in \mathbb{M} .

3 Simplification

We prefer to use the description of the Jacobi set as the level set of a gradient-based comparison measure [1] because it leads us to a natural algorithm for computing Jacobi sets. Let \mathbb{M} be a 2-manifold smoothly embedded in \mathbb{R}^3 . The *comparison measure* at a point $x \in \mathbb{M}$ for two Morse functions f and g is defined as $\kappa_x = \|\nabla f(x) \times \nabla g(x)\|$. Assuming \mathbb{M} is orientable, we define the

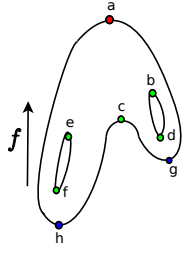


Fig. 1. Left: A two-holed 2-manifold and the height function defined on it. Points in blue, green, and red correspond to minima, saddle, and maxima of the function, respectively. **Right:** The Reeb graph of the height function. Loops in the Reeb graph correspond to holes in the manifold.

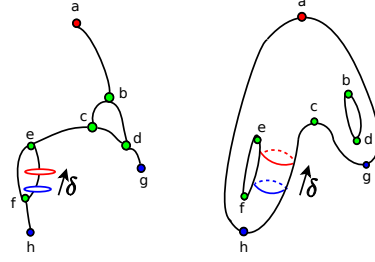


Fig. 2. Offsetting a level set component. **Left:** Level set components on the manifold. **Right:** Offsetting a level set component (blue) to another component (red) along an edge of the Reeb graph.

sign extended comparison measure, κ_x^S , at the point x with unit normal \hat{n} as $\kappa_x^S(f, g) = (\nabla f \times \nabla g) \cdot \hat{n}$. The sign extended comparison measure is a function defined on the manifold \mathbb{M} and the Jacobi set can be described as the set of points where κ_x^S equals zero, i.e. the zero level set of κ_x^S , $\mathbb{J} = \kappa_x^{-1}(0) = \kappa_x^{S^{-1}}(0)$.

The Jacobi set contains spurious loops because of noise and degeneracies in the data. Simplification of the Jacobi set refers to the reduction in number of components of \mathbb{J} with minimal change to the relationship between the two input functions.

The relationship between the functions is quantified by the *global comparison measure* κ , which is equal to the comparison measure integrated over the manifold and normalized by the total area [1].

$$\kappa = \frac{1}{\text{Area}(\mathbb{M})} \int_{x \in \mathbb{M}} \kappa_x dA_x,$$

where dA_x is the area element at x .

Offsetting components. The Jacobi set components are altered by computing offset level set components. Let p and p' be two level set components such that their corresponding points on the Reeb graph are connected by a monotone path. The level set component p is said to be offset to p' if it is replaced by the component p' . The cost of an offset operation is given by the hypervolume, which is computed as an integral over the swept region R of the domain:

$$H = \frac{1}{\text{Area}(\mathbb{M})} \int_{x \in R} \kappa_x dA_x. \quad (1)$$

Figure 2 shows a level set component offset upwards by a hypervolume δ . The direction of offset is upward if the function value increases and downward otherwise. We simplify the Jacobi set by computing offsets in an appropriate direction.

The following basic offset operations are used in the simplification process.

Merge : Two components whose edges share a common saddle are offset to the saddle so that they merge. The merged component is further offset by a small value resulting in a single component.

Split : A component is offset to a saddle and is further offset by a small value resulting in a split.

Purge : A component is offset to a local maximum or minimum. A further offset by a small value removes the component.

Create : A component is created at a local maximum or minimum and offset by a small value.

Figure 3 illustrates the basic offset operations, using the Reeb graph. The Reeb graph is naturally suited to represent the offsets because it traces the connected components of the level sets. Only two operations result in a reduction in the number of components. Temporary splits may be required to obtain a small number of components. We ensure that the number of splitting operations is lower than the number of component merging operations. We show in the next section that twice the total hypervolume swept during the operations is an upper bound over the total change in relationship between the functions.

The first step in the simplification procedure is the computation of the Reeb graph for κ_x^S . Arcs in the Reeb graph that contain the zero level set are also identified.

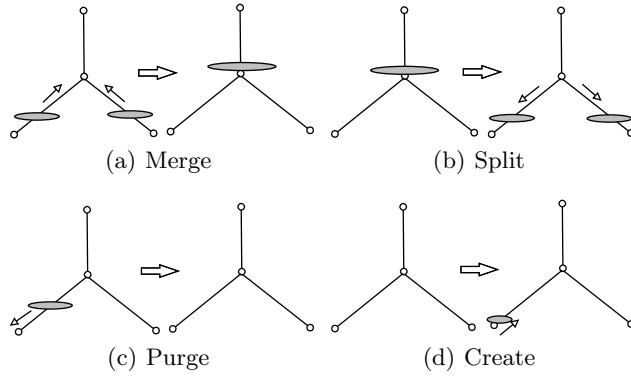


Fig. 3. Different offset operations used during simplification. All offsets are shown against the Reeb graph of κ_x^S .

Algorithm. The required simplification is specified as a percentage of the global comparison measure. The corresponding hypervolume threshold, i.e., the total

hypervolume allowed for the operations is calculated next. Since each simplification operation involves exactly one critical point, we can represent an offset by a critical point. We first augment the Reeb graph by adding dummy nodes at level zero. This augmented graph is transformed into a directed graph by replacing each arc uv with a directed arc uv (arc from u towards v), if $|\kappa_v^S| > |\kappa_u^S|$, see Fig. 4. Each vertex is then assigned a profit $P(v)$ given by

$$P(v) = \begin{cases} 1 & \text{if } v \text{ is a dummy vertex} \\ in(v) - out(v) & \text{otherwise,} \end{cases}$$

where $in(v)$ and $out(v)$ represent the indegree and outdegree of v in the directed graph. The profit for a non dummy node signifies the reduction in number of Jacobi set components if the operation corresponding to the node is chosen. The optimal simplification can now be formulated as an integer linear program (ILP) that maximizes profit. The variables in the ILP correspond to nodes of the directed Reeb graph.

$$\max \sum P(v)x_v$$

subject to constraints

$$\begin{aligned} \sum C(v)x_v &\leq T \\ x_v - x_u &\leq 0 \text{ for a directed arc } uv \\ x_u + x_v &\leq 1 \text{ } u,v \text{ adjacent to a common dummy node} \\ x_u, x_v &\in \{0, 1\} \end{aligned}$$

The cost $C(v)$ for each simplification operation is the sum of hypervolumes of the incoming arcs. T is the threshold given as input. A simplification operation is performed on a node if the corresponding variable in the ILP is set to one. The first constraint bounds the total hypervolume for the simplification. The second constraint enforces a dependency between variables corresponding to a directed arc uv . This dependency captures the fact that a simplification operation at v can be performed only after a level set component has been offset through the node u . At dummy vertices, there is a choice to perform an offset in either of the directions but not both. This choice is modeled in the third constraint. The ILP is a variant of the knapsack problem with dependencies among objects. Though a solution to the above ILP corresponds to the optimal simplification, the computation is slow in practice. So, we resort to a greedy strategy that chooses the least cost offset operation at every step until the threshold is reached. The greedy strategy has an additional advantage-it enables the creation of a multi-resolution representation of the Jacobi set.

The greedy algorithm requires all nodes to be stored in a priority queue. The priority queue is initialized with all possible simplification operations and updated with new operations that may become valid after an offset is performed. We define a node of the directed Reeb graph as *unreachable* if it cannot be reached by a path from a dummy node and *reachable* otherwise. Unreachable

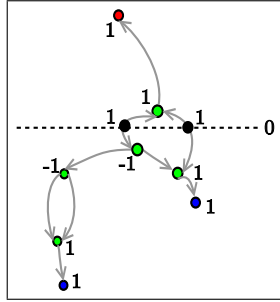


Fig. 4. Directed Reeb graph. The dotted line in the figure shows level 0. The dummy vertices are shown in black on the zero line. The profit for each node is also shown.

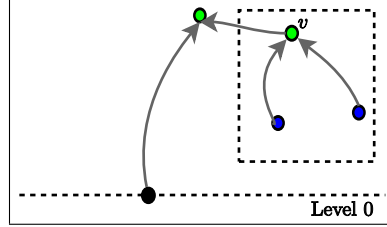


Fig. 5. A Section of a Reeb graph with unreachable vertices shown in the boxed rectangle. The unreachable component prevents the algorithm to proceed beyond the merge saddle v .

nodes may become obstacles that prevent offset operations. For example, a saddle with an incoming arc from an unreachable node prevents a merge operation, see Fig.5. Let G denote the directed Reeb graph and H denote the subgraph of G containing all unreachable vertices. A component J of H is a connected component in the undirected version of H . The cost of removing J is the sum of the cost of all edges of G that have at least one end point in J . If the algorithm is not able to proceed due to some obstacles, then least cost components of unreachable vertices are removed from G until a valid operation is identified. Finally, we extract offset components using seed sets stored in the Reeb graph [6]. We also ensure that the number of simplification operations with negative profits is smaller than a constant fraction of the operations with positive profits. This ensures that the number of components decreases as a result of simplification.

4 Analysis

In this section we show that twice the hypervolume swept during a simplification operation is an upper bound over the change in the relationship between the input functions.

Simplifying the input function. We do not change the function values in our experiments. However, we now compute changes to the function f caused by a small offset in order to obtain the upper bound. Figures 6a and 6b depict the changes to the function f after offsets in the up and down directions respectively. An upward offset introduces critical points at E and F of f restricted to level set I of g . To accomplish this, the function values at E and F can be interchanged to become $f(F)$ and $f(E)$ respectively. Within level set II of g , the critical points of f move from B and C to A and D respectively. The function f restricted to level set II between A and D is made monotone to achieve this movement of

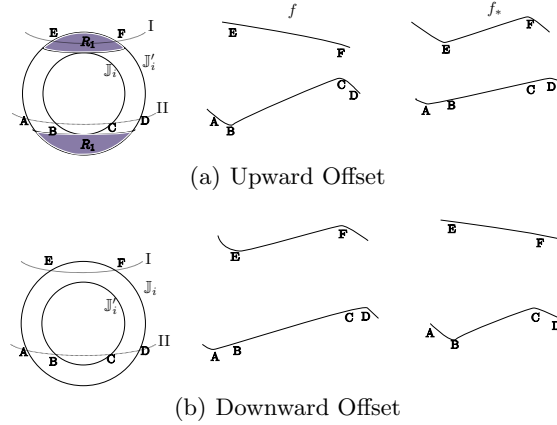


Fig. 6. Simplifying the input function. The left column shows a Jacobi set component \mathbb{J}_i and its offset version \mathbb{J}'_i . The dashed lines are level sets of the function g . The center column shows f restricted to the level sets I and II. The right column shows the simplified function f_* that corresponds to the offset Jacobi set component \mathbb{J}'_i .

critical points. The function values at A and D do not change and therefore the new pair have a reduced persistence. Downward offset destroys the critical point pair E and F and the restricted function f between E and F is made monotone. The function values at E and F are interchanged to become $f(F)$ and $f(E)$ respectively. Within level set II of g , critical points move from A and D to B and C respectively.

Effect on global comparison measure. As shown by Edelsbrunner et al. [1], the global comparison measure is given by

$$\kappa = \frac{2}{Area(\mathbb{M})} \int_{v \in \mathbb{J}} sign(v) f(v) dg,$$

where $sign(v)$ is defined as

$$sign(v) = \begin{cases} +1 & \text{if } v \text{ is a maximum of } f|_{g^{-1}(g(v))} \\ -1 & \text{otherwise.} \end{cases}$$

Let \mathbb{J}_i denote the i^{th} component of the Jacobi set. Define

$$\kappa_i = \frac{2}{Area(\mathbb{M})} \int_{v \in \mathbb{J}_i} sign(v) f(v) dg.$$

κ_i can be interpreted as the contribution of \mathbb{J}_i to the global comparison measure, $\kappa = \sum_i \kappa_i$.

Since the change to the function f corresponding to an offset is local to the region of the component, we will now compute the change in κ_i corresponding to an upward offset. If f_* is the modified function, the change in κ_i is given by

$$|\delta\kappa_i| = \frac{2}{Area(\mathbb{M})} \left| \int_{v \in \mathbb{J}'_i} sign(v) f_*(v) dg - \int_{v \in \mathbb{J}_i} sign(v) f(v) dg \right|.$$

Let R be the region of \mathbb{M} swept during the offset and R_1 be the region where the level sets of g do not intersect \mathbb{J}_i (shaded region in Fig.6a). The integral over \mathbb{J}'_i can be rewritten as a sum of integrals over two regions:

$$|\delta\kappa_i| = \frac{2}{Area(\mathbb{M})} \left| \int_{v \in \mathbb{J}'_i \cap R_1} sign(v) f_*(v) dg - \int_{v \in \mathbb{J}_i} sign(v) f(v) dg + \int_{v \in \mathbb{J}'_i \cap (R - R_1)} sign(v) f_*(v) dg \right|. \quad (2)$$

Consider the level sets I in Fig.6a. The difference between function values at E and F can be written as

$$f_*(F) - f_*(E) = f(E) - f(F) = \int_F^E \|\nabla f_t(x)\| dl.$$

Here, $\nabla f_t(x)$ represents the tangential component of $\nabla f(x)$ along the level sets and dl is the length element along the level set. The integral of $sign(v) f_*(v)$ over $\mathbb{J}'_i \cap R_1$ can be rewritten as an integral over R_1 using the above expression,

$$\int_{v \in \mathbb{J}'_i \cap R_1} sign(v) f_*(v) dg = \iint_{x \in R_1} \|\nabla f_t(x)\| dl dg.$$

Let du be the length element orthogonal to the level set. The area element is given by $dl du$. Using the fact that $dg = \|\nabla g(x)\| du$,

$$\begin{aligned} \int_{v \in \mathbb{J}'_i \cap R_1} sign(v) f_*(v) dg &= \iint_{x \in R_1} \|\nabla f_t(x)\| \|\nabla g(x)\| dl du = \int_{x \in R_1} \|\nabla f(x) \times \nabla g(x)\| dA_x \\ &= \int_{x \in R_1} \kappa_x dA_x. \end{aligned} \quad (3)$$

Consider the level set II of g in Fig.6a:

$$f_*(A) = f(A) = f(B) + \int_B^A \|\nabla f_t(x)\| dl$$

and

$$f_*(D) = f(D) = f(C) - \int_D^C \|\nabla f_t(x)\| dl.$$

Combining the above two equations,

$$\begin{aligned} (f(C) - f(B)) - (f_*(D) - f_*(A)) &= (f_*(A) - f(B)) + (f(C) - f_*(D)) \\ &= \int_B^A \|\nabla f_t(x)\| dl + \int_D^C \|\nabla f_t(x)\| dl. \end{aligned}$$

All pairs of points $A, D \in \mathbb{J}'_i \cap (R - R_1)$ have a corresponding pair $B, C \in \mathbb{J}_i$. So, we have

$$\begin{aligned} \left| \int_{v \in \mathbb{J}'_i \cap (R - R_1)} \text{sign}(v) f_*(v) dg - \int_{v \in \mathbb{J}_i} \text{sign}(v) f(v) dg \right| \\ &= \iint_{x \in (R - R_1)} \|\nabla f_t(x)\| \|\nabla g(x)\| dl du \\ &= \int_{x \in (R - R_1)} \kappa_x dA_x. \end{aligned} \quad (4)$$

Substituting (4) and (3) in (2) and using the triangle inequality,

$$|\delta k_i| \leq \frac{2}{\text{Area}(\mathbb{M})} \int_R \kappa_x dA_x = 2H.$$

The above inequality can be similarly derived for the downward offset. Thus, the hypervolume is a conservative estimate of the change in relationship between f and g caused by an offset.

5 Implementation for Piecewise Linear Functions

Scalar scientific data is typically represented by piecewise linear functions on triangle meshes, where the gradient and hence κ_x^S is not defined at vertices of the mesh. Given a vertex v of the triangle mesh, its neighborhood is the Voronoi region as shown in Fig. 7. Meyer et al. [7] used the Voronoi region to define discrete differential operators with minimal numerical error for triangulated surfaces. Let T_1, T_2, \dots, T_t be triangles that intersect the neighborhood R of v . The sign extended comparison measure κ^S is constant within each of the regions $T_i \cap R$. We follow Meyer et al. to define κ_v^S as the average value of the sign extended measure over R ;

$$\kappa_v^S = \frac{1}{\text{Area}(R)} \sum_{i=1}^t \kappa_i^S \text{Area}(T_i \cap R),$$

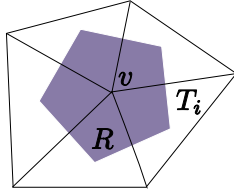


Fig. 7. Consider the vertex v and its adjacent vertices as a point set. The neighborhood R of a vertex v on a piecewise linear surface is represented by the Voronoi region of v .

where κ_i^S is the value of the sign extended comparison measure at a point that lies in the interior of T_i . Note that the gradients of f and g are constant in the interior of a triangle and hence κ_x^S is also constant within a triangle. The sign extended comparison measure is stored at vertices and a linear approximation is used within the edges and triangles. This approximation does not introduce significant artifacts in practice. The zero level set can be extracted using a marching triangles algorithm or from seed sets computed using a Reeb graph of κ_x^S .

6 Applications

We demonstrate the usefulness of the simplified Jacobi set using two different applications. Our approach to the definition and simplification of Jacobi sets is particularly useful when studying the relationship between two functions using their gradients.

Visualizing Silhouettes. Given a view direction d in \mathbb{R}^3 and a 2-manifold \mathbb{M} embedded smoothly in \mathbb{R}^3 , the silhouette is the set of points in \mathbb{M} where the tangent plane is parallel to d . Consider a Cartesian coordinate system with the z -axis along the view direction d . The Jacobi set of the two scalar fields $f(x, y, z) = x$ and $g(x, y, z) = y$ is the required silhouette. The silhouette of a model of the hand is shown in Fig.8(c). The model is shown in the original orientation in Fig.8(a). The view direction is perpendicular to the plane of paper. The orientation of the model has been changed for a better view of the computed silhouette in Figs.8(b) and 8(c). As seen from the figure, the silhouette has many components that are unimportant and the silhouette itself appears to contain noise. The simplification process removes small components because their removal does not adversely affect the relationship between the fields f and g used to compute the silhouette. We found that simplification using 2% threshold removed all noise. The Jacobi set was simplified using the greedy algorithm.

Combustion. We apply our algorithm to study a time varying dataset from the simulation of a combustion process. This application demonstrates the use of simplification when handling degenerate data. Degeneracies occur when κ_x is zero within a region, resulting in the Jacobi set containing higher dimensional

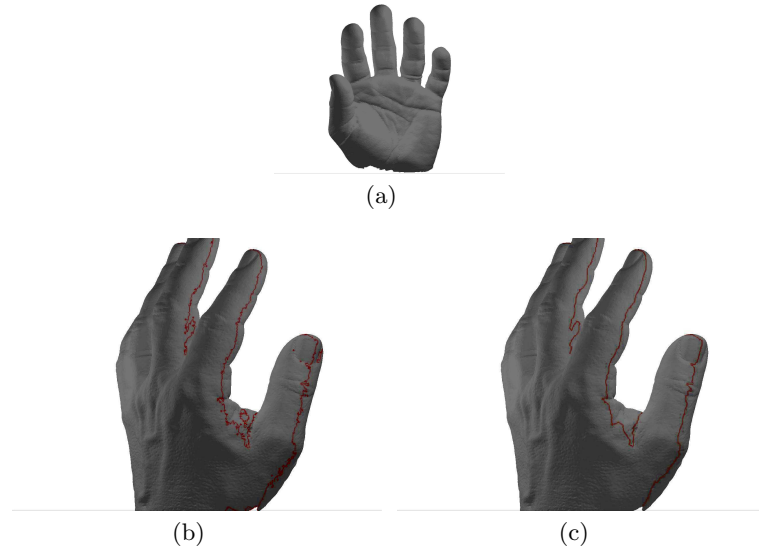


Fig. 8. Silhouettes. (a) Model of a hand in its original orientation. (b) Silhouette when viewed from a different angle. (c) Simplified silhouette.

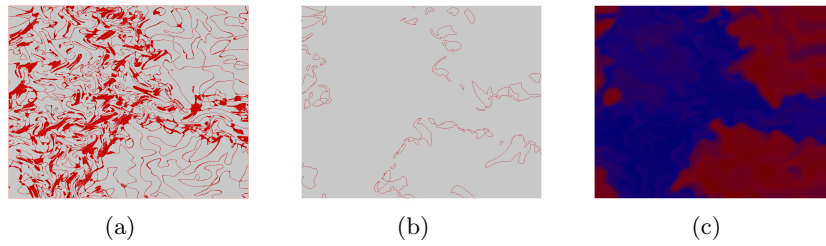


Fig. 9. Combustion. (a) Jacobi set of H₂ and O₂ in the 64th time step. (b) Simplified Jacobi set. (c) Concentration of O₂.

parts. During simplification, Jacobi set components within degenerate regions are automatically removed because they do not contribute to κ .

The dataset consists of the concentrations of H₂(fuel) and O₂(air) defined on a 600x600 grid for 67 time steps. We compute and simplify the Jacobi set for H₂ and O₂ at different time steps to identify the front of combustion. Combustion begins at regions where the fuel-air mixture is appropriate for ignition. The data is degenerate away from the front, thereby introducing noise in the Jacobi set. Figure 9 shows the results for the 64th time step when the combustion is in its final stage. The simplified Jacobi set again appears at the front. Figure 9(c) shows the O₂ concentration. Blue signifies a low function value and red signifies a high function value. The front consists of the boundary of red regions, which is also traced by the simplified Jacobi set.

7 Conclusions

We have described a robust algorithm for simplifying the Jacobi set of two Morse functions. Our algorithm ensures minimal change to the relationship between the two functions. Future work includes extending the algorithm to multiple functions and higher dimensions.

Acknowledgments

This work was supported by the Department of Science and Technology, India under grant SR/S3/EECE/048/2007. The combustion data was provided by Jackie Chen and Valerio Pascucci. We also thank the anonymous reviewers for their feedback.

References

- [1] H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Local and global comparison of continuous functions. In *Proceedings of the conference on Visualization '04*, pages 275–280, Washington, DC, USA, 2004. IEEE Computer Society.
- [2] J. Bennett, V. Pascucci, and K. Joy. Genus oblivious cross parameterization: Robust topological management of inter-surface maps. In *Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, pages 238–247, Washington, DC, USA, 2007. IEEE Computer Society.
- [3] P-T Bremer, E. M. Bringa, M. A. Duchaineau, A. G. Gyulassy, D. Laney, A. Mascarenhas, and V. Pascucci. Topological feature extraction and tracking. *Journal of Physics: Conference Series*, 78:012007 (5pp), 2007.
- [4] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 454, Washington, DC, USA, 2000. IEEE Computer Society.
- [5] H. Edelsbrunner and J. Harer. Jacobi set of multiple morse funtions. In *Foundations of Computational Mathematics, Minneapolis, 2002*, pages 37–57. Cambridge Univ. Press, 2004.
- [6] M. Kreveld, R. Oostrum, C. Bajaj, V. Pascucci, and D. Schikore. Contour trees and small seed sets for isosurface traversal. In *Proceedings of the thirteenth annual symposium on Computational geometry*, pages 212–220, New York, NY, USA, 1997. ACM.
- [7] M. Meyer, M. Desbrun, P. Schroder, and A. Barr. Discrete differential geometry operators for triangulated 2-manifolds. *VisMath.*, 2002.
- [8] T. Echehki and J. H. Chen. Direct numerical simulation of auto-ignition in inhomogeneous hydrogen-air mixtures. In *Proceedings of the 2nd Joint Meeting U.S. Sections Combustion Institute*, 2001.
- [9] Y. Matsumoto. *Introduction to Morse Theory*. Translated from Japanese by K. Hudson and M. Saito. Amer. Math. Soc., 2002.
- [10] G. Reeb. Sur les points singuliers d’une forme de pfaff complètement intégrable ou d’une fonction numérique. *Comptes Rendus de L’Académie des Séances*, 222:847–849, 1946.