J Vis (2013) 16:85–94
DOI 10.1007/s12650-012-0148-8

REGULAR PAPER

Estela A. Gavosto · James R. Miller

Visualization of data on unfolded hypercubes

Received: 3 April 2011 / Revised: 21 August 2012 / Accepted: 20 September 2012 / Published online: 8 December 2012

Abstract We present a new method to visualize data of the form $y = f(x_1, x_2, x_3, x_4)$ that can be extended to n-dimensions. The essence of the method consists of slicing the data until the dimensions of the slices are reduced to $n = 4$ and then "unfold" the data into 3D space. Specifically, we map a visual representation of y onto the boundaries of a hypercube or other polytope and unfold those boundaries into 3D space. The display of the data provides a structural representation of 2D slices which can be adjusted and manipulated, allowing geometric properties like connectivity to be easily studied.

Keywords Multivariate visualization · Visualization techniques methodologies

1 Introduction

In this paper, we present a new method for displaying a function $y = f(x_1, x_2, x_3, x_4)$, where the independent variables x_i, $i = 1, \ldots, 4$, vary over an interval of real numbers. Our setup is a particular case of multidimensional multivariate visualization that refers to the visualization of k dependent variables (or attributes) as a function of their position in n-dimensional space (de Oliveira et al. 2003; Love et al. 2005). The data considered could correspond to discretely sampled physical data or to numerically generated data. The method borrows on the intuition of traditional engineering drawing in which three orthographic views are used to represent and reconstruct a 3D object. The main idea and challenge is to extend the method to "carvings" of continuous data and higher dimensions.

Many different techniques have been considered to study this type of data. We summarize several of them in the next section. All methods to visualize higher dimensional data reduce the number of independent dimensions to two or three by projecting or slicing data to lower dimensional spaces, applying numerical transformations and/or mapping the data set to different variables. The resulting lower dimensional data is displayed on the screen in a variety of ways, for example: by rearranging the axes to be non-orthogonal and displaying data along all axes simultaneously; by embedding combinations of data dimensions to form composite spatial dimensions such as stacking; or by mapping each data value from large datasets to a unique pixel of the screen. Each method has advantages and limitations. One major trade-off is always between maintaining the global geometric features and observing details only possible in lower dimensional slices.

E. A. Gavosto (✉)
Department of Mathematics, University of Kansas, Lawrence, KS 66045, USA
E-mail: gavosto@math.ku.edu

J. R. Miller
Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS 66045, USA
E-mail: jrmiller@ku.edu

We define an n-dimensional polytope as a finite region in n-dimensional space bounded by a finite number of hyperplanes. That is an n-dimensional polytope will be bounded by $(n - 1)$-dimensional polytopes. The majority of our examples involve visualizing a hypercube (4D polytope) by examining its eight unfolded cube faces. Our method operates by "carving" the data with an n-dimensional polytope containing a region of the space to be studied. When $n > 4$, our method begins by slicing the data until the dimensions of the slices are reduced to $n = 4$ and then "unfold" the data into 3D space. The final display is an engaging arrangement of slices representing faces of the polytope that preserves 4D geometric structure of the set in higher dimensions and at the same time reveals details of higher dimensional slices resulting in a 4D visualization.

User interface issues are very important to the success of this technique. A recent study has shown that an unfolded cube display with six projections provided an enhanced display with improved time efficiency and higher surface visibility for computer tomography colonoscopy (Vos et al. 2003). Our method retains the cube unfolding advantages in an intuitive and exploratory interface. The display preserves the higher dimensional geometry by allowing only movements of the boundary faces that result in another unfolding. This dynamic manipulation of the boundary faces gives a mechanism for examining the relationships in the data by linking and arranging the views for the user. The end result is a tool that is useful to users with strong mathematical training.

We discuss related work in the next section and the rest of the paper is organized as follows. First, to develop an intuition about our technique, we will begin with the simplest case, $n = 3$, and a cube. Second, we explain our unfolding method in the 4D case and outline the n-dimensional. Third, we present several examples: a mathematical one (the complex Henon map), a visualization of the electrostatic potential of a protein, and a visualization of some data from the Hurricane Isabel data set. Finally, we describe the implementation and its user evaluation results, and present our conclusions and future directions.

2 Previous work

Many approaches have been introduced to visualize higher dimensional data. Space does not permit an exhaustive survey here. We describe some key contributions and refer the reader to Bajaj et al (1998) and dos Santos and Brodlie (2001) for more extensive background material. Asimov (1985) introduced the concept of the grand tour consisting of a path of orthogonal projections of higher dimensional data given by a sequence of 2D subspaces. The sequence of subspaces is selected in such a way that it is dense in the space of all subspaces. The suitability of a grand tour for a specific application is based on statistical tests.

Inselberg and Dimsdale (1990) introduced a tool based on a system of parallel coordinates inducing a mapping from an n-dimensional set to a 2D set. Points in n-dimensions are represented by a polyline that intersects n parallel axes at the value of the coordinates of the point. The expectation is that patterns emerge from which statistically significant conclusions can be drawn.

Feiner and Beshers (1990) developed the concept of "worlds within worlds" consisting of nested coordinate systems. The method allows the user to represent how hyperplane slices through high dimensional spaces are use to obtain a 3D image. Mihalisin et al (1991) described an approach based on plotting a dependent variable on hierarchical axes corresponding to the n-dimensional independent variable lattice.

Hanson and Heng (1992a, b) introduced a method to study a 3D scalar field by viewing the mapping as an elevation mapping in 4D: three independent variables used to define a fourth dependent variable. The authors extended their previous work for displaying 3D manifolds in 4D space to scalar functions. They display 4D rotated pseudocolor volume grids and use 4D lighting and shading rules.

Van Wijk and van Liere (1993) introduced the HyperSlice technique. This method gives a representation of a multidimensional function as a matrix of orthogonal 2D slices. The tool includes several interaction approaches for navigation. Dos Santos and Brodlie (2002) extended this method to consider 3D cells with the addition of other exploratory features.

A completely different approach was presented by Bajaj et al (1998). The Hypervolume visualization technique is a generalization of direct parallel projection methods in volume rendering. In contrast with some of the previous methods described that give detailed lower dimensional slices, Hypervolume provides global views of scalar fields.

3 Unfolding data in a cube

We first introduce our basic approach using a 3D cube. Our method is grounded in the classical engineering method of orthographic views to reconstruct a 3D object. We use it to build the intuition of the reader for the ideas that we will present in higher dimensions. Let us consider data of the form $y = f(x_1, x_2, x_3)$ where x_1, x_2 and x_3 are variables in a region of the space. We visualize the dependent variable y by mapping its range of values to a color and displaying it as a texture mapped onto faces representing subsets of the domain. The main concept is to study and understand this scalar function by looking at the data on the unfolded faces of the cube. We slice this data with a box of dimensions $[c_1, d_1] \times [c_2, d_2] \times [c_3, d_3]$ where c_i and d_i, $i = 1, 2, 3$ are real numbers. Notice that we are taking a structural collection of 2D slices of 3D data.

In our first example we define the dependent variable as the distance to the torus defined by the equation $\left(\sqrt{x_1^2 + x_2^2} - 2\right)^2 + x_3^2 = \frac{1}{4}$. Fig. 1 show how the six faces of the cube with the data form the unfolding of a cube and the cube superimposed in place on the unfolding.

Information about the data in the cube can be inferred from this unfolding in a fashion analogous to the way that we read views in an orthographic projection. A simple characteristic that we could have missed with a 2D slice of 3D data is connectivity. A typical example is two regions that appeared to be disconnected (see the two green areas in the front face of the cube in Fig. 1). By taking slices in a given direction, the connectivity with the regions cannot be observed. The green region is clearly connected in space as the top and central square slices in the unfolded cube show. This characteristic can also be observed in unfolded cubes of Fig. 2. It is easy to verify that there are 11 different unfoldings of the cube. Interactively switching among these unfoldings allows us to observe various characteristics of the data (see Fig. 2). For example, Unfolding I shows how the red values in faces 1 and 6 are connected through the region 2. This is not evident in unfoldings I and II. Unfolding III shows how the face 6 is connected to the green values "ring" that is part of faces 2, 3, and 4.

4 Unfolding data in hypercubes and other polytopes

For simplicity of exposition, we will concentrate on 4D polytopes of the form of a product of 4 intervals. We denote the coordinate dimensions as (x_1, x_2, x_3, x_4). Recalling the dependent variable is y, we unfold the 4D polytope to 3D. We will use the fact that a hypercube has eight boundary faces and we will define its "canonical unfolding" in the next section.

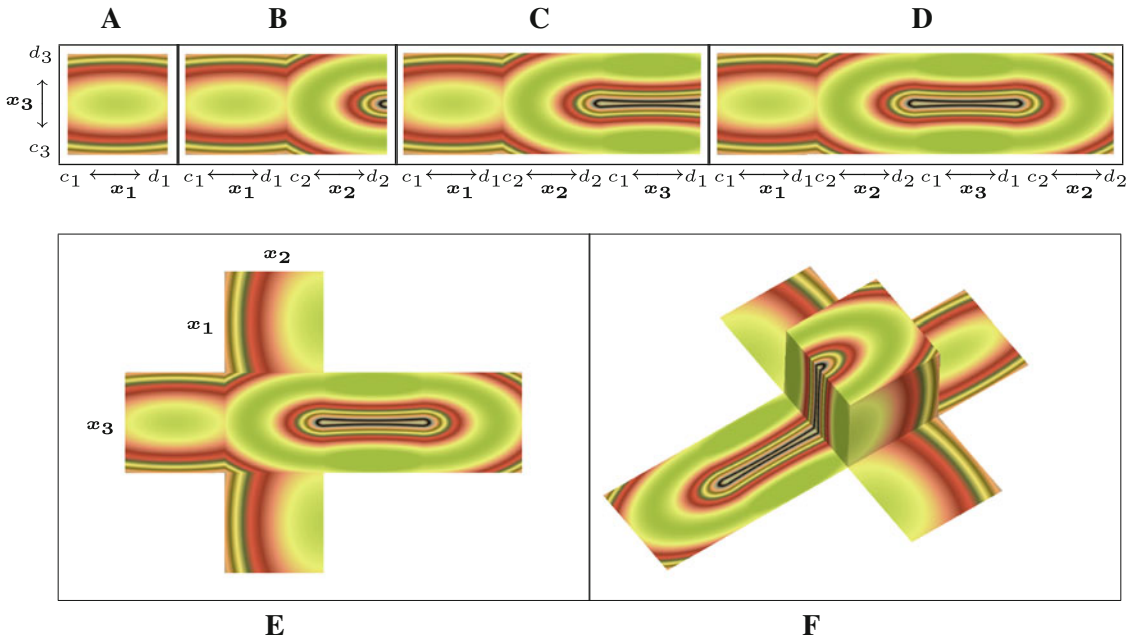

Fig. 1 Steps in the unfolding of a *cube* (**a**–**e**), *cube* superimposed on one of the unfoldings (**f**)

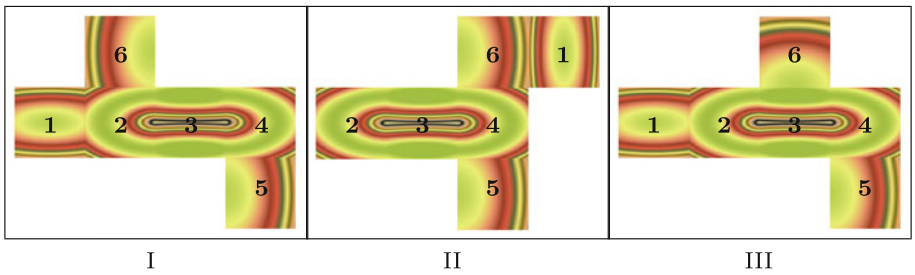

Fig. 2 Unfoldings **I**, **II**, and **III** of the same cube

4.1 Canonical unfolding procedure

We can imagine the canonical unfolding as proceeding in two stages. In the first, the x_1 and x_3 faces are unfolded along the x_1. In the second, the remaining four boundary faces are laid out in pairs perpendicular to the x_1 axis. Figure 3 below illustrates this two-stage procedure step by step for the unfolding of a 4D hypercube into 3D space. The data corresponds to a 4D version of our example from Figs. 1 and 2. In this case, it corresponds to the rotation of a sphere in 4D. That is, our dependent variable is the distance to the manifold defined by the equation $\left(\sqrt{x_1^2 + x_2^2} - 2\right)^2 + x_3^2 + x_4^2 = \frac{1}{4}$. The hypercube is of dimensions $[c_1, d_1] \times [c_2, d_2] \times [c_3, d_3] \times [c_4, d_4]$.

It can be shown that there are 256 different unfoldings of a hypercube, see Banchoff (1990), Turney (1994) and Aguilera Ramírez and Pérez Aguila (2002) (Our interactive manipulations only allow valid repositionings of cube faces.). The interactive snapping of cubes to generate alternate unfoldings can now be used to explore this function. Notice that in the first unfolding of Fig. 4, the top face has two disconnected green areas (corresponding to two disconnected spherical regions in the subspace (x_1, x_2, x_3)). Changing the location of some of the boundary faces shows in the second unfolding how different "handles" connect the two "spheres" (the equivalent of the two circles in a 3D torus which are connected by handles in the perpendicular direction). The third unfolding in Fig. 4 illustrates how the different handles fit from another point of view in 4D space.

The system is designed to work for n-dimensional hypercubes for arbitrary values of n. The unfolding of an nD hypercube generates a structured collection of $(n-1)$D hypercubes that comprise the boundary of the original hypercube. The idea described for $n = 4$ above is generalized and encoded in a pair of methods in the implementation. The first step generates a single boundary $(n-1)$D hypercube in the canonical unfolding of an nD hypercube. A given request is for the $(n-1)$D hypercube face i where the coordinate dimension, $c = \frac{i}{2}$; we generate the minimum boundary in coordinate direction c if i is even; we generate the maximum boundary $(n-1)$D hypercube face in that direction otherwise. If the initial hypercube has dimension greater than 4 (say, $k+4$), this method is called k times so that we can begin with a 4D

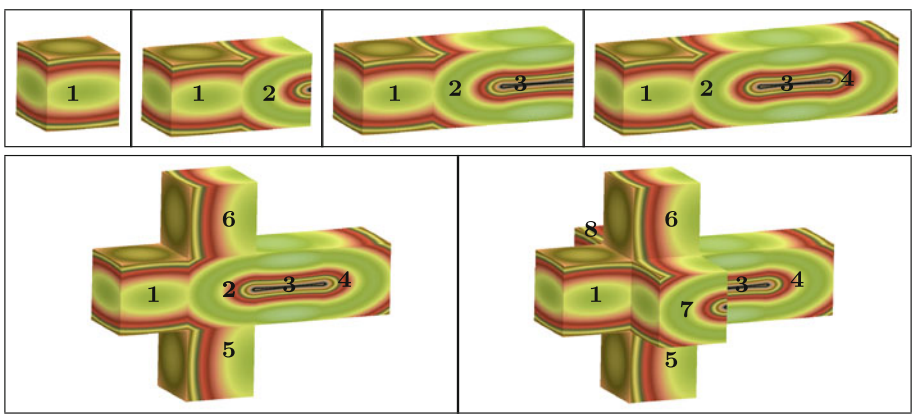

Fig. 3 Step by step canonical unfolding of a hypercube with mapped data where (*1*): (x_1, c_2, x_3, x_4) – slice, (*2*):(c_1, x_2, x_3, x_4) – slice, (*3*): (x_1, d_2, x_3, x_4) – slice (*4*): (d_1, x_2, x_3, x_4) – slice, (*5*): (x_1, x_2, c_3, x_4) – slice , (*6*): (x_1, x_2, d_3, x_4) – slice , (*7*): (x_1, x_2, x_3, c_4) – slice, (*8*): (x_1, x_2, x_3, d_4) – slice

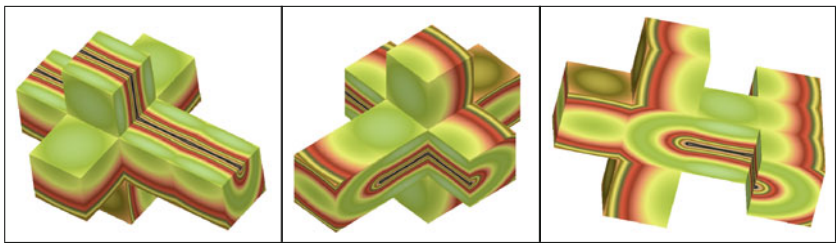

Fig. 4 Different unfoldings of a hypercube with mapped data

hypercube to visualize. This method is then called $2n$ times (where n is the minimum of the original dimension and 4) to generate the boundaries of the canonical unfolding. A simpler way of conceptualizing this process is to think in a 2D visualization of a 4D hypercube. Following the previously described process, the final output will be a collection of eight unfolded cubes, each one to them corresponding to the 2D unfolding of each of the 8 faces of the hypercube.

Given the bare bones idea of how unfoldings are created and manipulated, we turn our attention to the details of how we define and use the "texture" on the unfolded faces to visualize our dependent variable y. We do this in the context of several applications described in the next sections.

5 Applications

5.1 Mathematical visualization: the Henon map

The so-called Julia set for the Henon map is a fractal set in 4D that poses many challenges to visualize. See Miller and Gavosto (2004) for some recent work and history of the visualization of this set. The dependent variable is an "escape rate" computed as a function of four real coordinates (corresponding to two complex numbers: $x_1 + ix_2$ and $x_3 + ix_4$). The escape rate is determined using the forward iterates of the mapping $H(x_1, x_2, x_3, x_4) = ((x_1 + ix_2)^2 + c + d\ (x_3 + ix_4), d(x_1 + ix_2))$ where $i^2 = -1$ and c and d are two complex parameters.

Many characteristics of the set can be studied using our techniques. The image in Fig. 5 corresponds to fixing the parameter c and changing the parameter d. Notice that for $d = 0$ the Henon map is just $H(x_1, x_2, x_3, x_4) = ((x_1 + ix_2)^2 + c, 0)$, so the iterates do not change in the third and fourth variables. The first hypercube corresponds to $d = 0$ and in the other ones the values of d increase. We can notice how the structure changes from a "cylindrical structure" for $d = 0$ to branches that open up in all the directions for larger ds. This branching structure is much harder to understand with 2D slices without the structure of the unfolding.

Manipulation of one unfolding in the screen by rotating the hypercube allows us to explore the object and to search, for example, for symmetries along an axis in 4D space. See Fig. 6 for a series of rotations of one unfolding of the hypercube. For fixed values of the parameters and fixed region in the data space, changing the unfolding allows us to see how the structured 3D slices fit in 4D space. Observe in Fig. 7 below how the boundary cubes match in the different unfoldings giving different "views" of the 4D set. The unfoldings here allow to see how the branching occurs in much more complicated manner for these values of the parameters and to identify the directions where the "three" like structured is preserved.

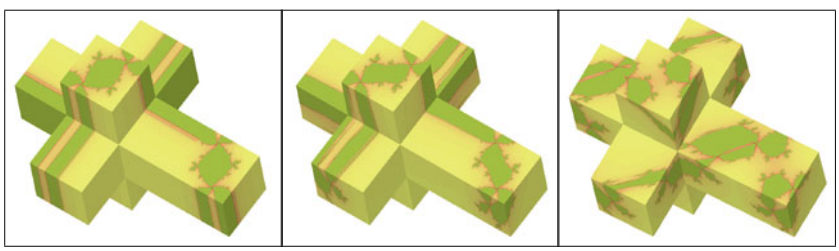

Fig. 5 Changing the d parameter for the Henon map data

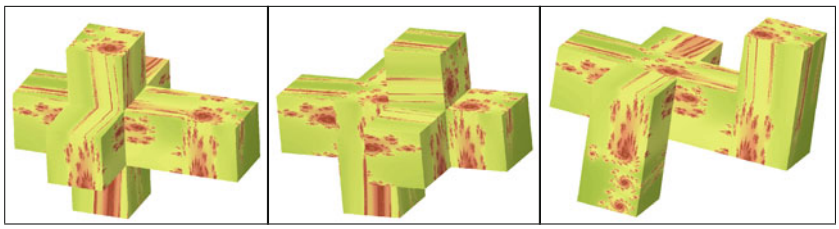

Fig. 6 Rotation of one hypercube to look for symmetries

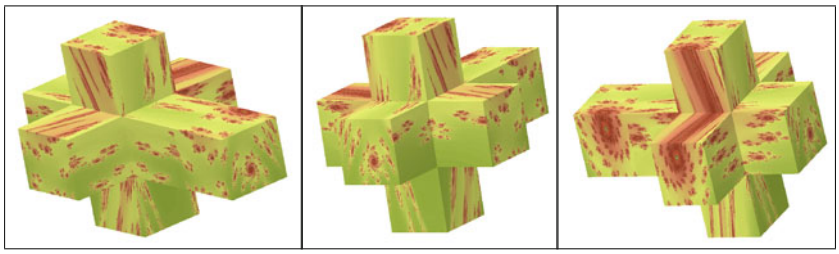

Fig. 7 Different unfoldings of a hypercube with Henon map data

5.2 Electrostatic potential for proteins

The example of Fig. 8 shows a hypercube display of the needle apparatus protein assembly. It consists of multiple copies of a MixH needle protein from *Shigella Flexneri*. The needle is a part of the type III secretion apparatus through which many Gram-negative bacteria inject the virulence factors into the host cell. We wish to study the electrostatic potential as a function of the concentration of salt in the medium. The potential can be computed using the Poisson–Boltzmann equation, and we are interested in visualizing it on and outside of the molecular surface. For purposes of our hypercube visualization, (x_1, x_2, x_3) represent the three spatial coordinates, and x_4 is the salt concentration. We compute the electrostatic potential (the dependent variable, y) as a function of these four independent variables and present the results in an unfolded hypercube. The potential is negative in some areas and positive in others. As the legend in Fig. 8 indicates, we use a diverging color scheme to map the negative half of the range to colors varying from dark green to yellow; the positive half of the range is then mapped from yellow to dark red.

Consider the two faces in Fig. 8 with the red axes on them. The vertical axes on the two faces correspond to x_2, the y coordinate direction in the protein assembly. The two horizontal red axes (which point in opposite directions due to the way the hypercube boundaries unfold) correspond to x_1, the x coordinate direction of the protein assembly. The two fixed independent variables on those two faces are x_3 and x_4, the z coordinate and the salt concentration, respectively. The x_3 coordinate is the same on both; the salt concentration is at its minimum on the left face, and it is at its maximum on the right face. In spite of the fact that the hypercube structure results in these two faces having a mirrored orientation, it is easy to see how

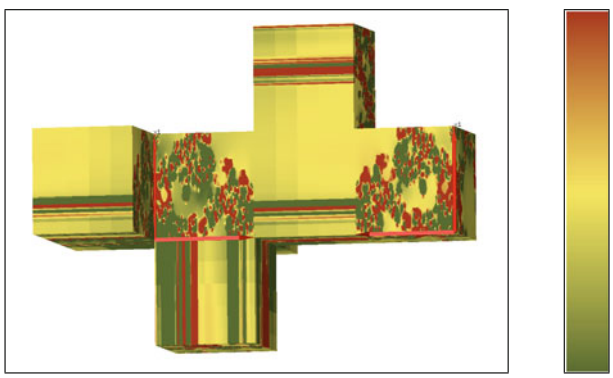

Fig. 8 Needle apparatus protein potential data

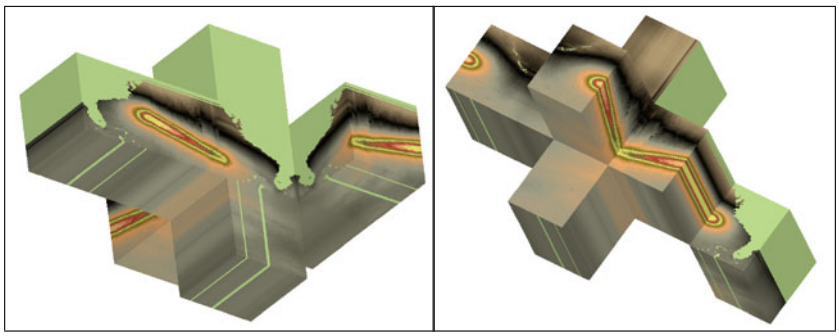

Fig. 9 Hurricane Isabel data mapped in a hypercube

increasing the salt concentration has affected the potential in the area displayed. Overall, the potential increases or remains unchanged. For example, in the U-shaped area about one-third of the way across the positive x_1 and x_2 axes (and also the area near the top of the x_2 axis), the potential has gone from being negative to being close to zero. There is an island just inside the face about half way up the x_2 axis that has gone from near zero to a positive value. By contrast, it is easy to locate other areas (e.g., most of the area around the origin of the face) that remain mostly unchanged.

Note also that the face connecting these two faces shows how this potential varies while transitioning from the minimum to the maximum salt concentration. Specifically, the intermediate face is a face of constant x_1 ($x_1 = max$; recall again that the two x_1 axes on the original two faces are pointing in opposite directions, hence each of those faces meet this intermediate one with $x_1 = max$). The vertical axis on the intermediate face like that of the two adjacent ones corresponds to the x_2 axis, hence reading this intermediate face from left to right allows us to see how the potential changes for each point along the vertical edge of the face as the salt concentration changes.

5.3 Hurricane Isabel

Hurricane Isabel was a hurricane that developed in the west Atlantic in September 2003. It eventually hit the North Carolina shore and came inland. The actual data set used here was simulated using the Weather Research and Forecast (WRF) model. The data set consists of 13 time-varying attributes, each considered a dependent variable captured as a function of longitude (x_1) latitude (x_2), elevation (x_3), and time (x_4). The example in Fig. 9 shows the pressure attribute mapped onto the hypercube boundaries.

Every face of each cube has two fixed and two varying independent variables. On faces where the shape of Florida is evident (again noting its mirrored orientation on some faces), clearly it is latitude and longitude that are varying; time and elevation are fixed. In the unfolded hypercube on the left of Fig. 9, the shape of Florida is clear on the top face on the left cube. Adjacent to that cube is another whose top face has longitude and height fixed, leaving latitude and time to vary. (Note, for example, how the piece of coastline at the top is smeared across that face. That emphasizes that we are seeing a fixed longitudinal slice.) The storm pressure shape we see on that face is showing how the eye of the hurricane passed through that longitudinal slice as a function of time and latitude. Similar effects can be seen in the alternate unfolding on the right of Fig. 9. Several of the faces show how the center of the hurricane passes through various longitudinal or latitudinal slices. The advantage of seeing the storm in 4D is that it is an instant in the space time domain and it allows to understand globally the hurricane immediately.

Note that the second "unfolding" is not a true unfolding since the cube in the bottom right corner shares only an edge with another cube. Cubes share a complete face boundary in true unfoldings. Our systems allows such pseudo-unfoldings because they frequently reveal structure not otherwise possible with true unfoldings.

6 Implementation details

The system is written in C++ and uses OpenGL for texture definition and image generation. When the program starts, a main graphics window and two GUI interface windows appear. Inside the main graphics

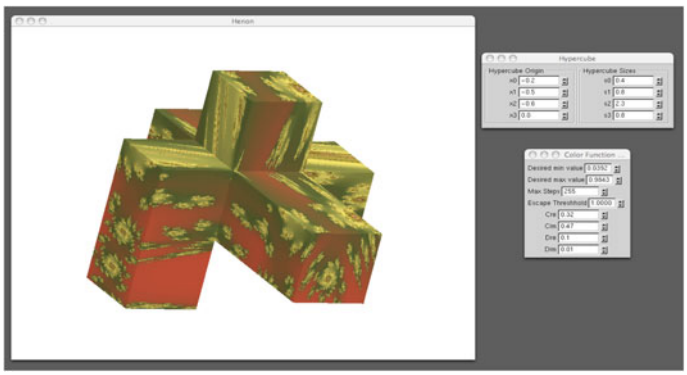

Fig. 10 The display window

window, the unfolded hypercube boundaries (along with the parent hypercube, if its dimension is 3 or less) are presented. This display can be dynamically rotated and zoomed. The zooming feature is useful to dynamically generate the texture at a resolution appropriate for the current zoom level. The unfolding gives a global view of the data and the zooming capability allows the user to closely examine local features. Several other interactive exploration techniques are available to the user (Fig. 10).

When the cursor passes over hypercube face boundaries, a pair of markers identifying corresponding boundaries appear. These serve as an indication to the user of what two boundaries will be snapped together if the user clicks on one of the two corresponding faces. If the cursor pauses for more than a few seconds on a face, the values of the two fixed hypercube coordinates are displayed.

There is a menu of options available from this window as well. Included are a host of display option toggles, saving and restoring state, and modifying the current parent hypercube dimensions.

In the first GUI window are controls to change individual coordinates of the origin of the parent hypercube as well as separate sizes of the hypercube in each coordinate direction. If the user wishes to change all (or even several) of these coordinates and sizes, it is more efficient to use the menu selection mentioned in the previous paragraph.

The second GUI window is tailored to the specific function being studied. For an iterated function system, for example, controls include the maximum number of iterations per point, the escape threshold, and other function-specific parameters.

Sometimes the region of a hypercube to be studied is much larger in 1D than another. By default, the hypercube display will present the actual aspect ratios. That frequently makes it difficult to see important detail information, so we also allow unequal scaling of the hypercube so that it can be drawn as an actual cube, even when its true aspect ratios are quite different. When this is needed, it is usually the case that the units of the independent coordinate directions are unrelated to one another (e.g., coordinates, time, pressure) hence this unequal scaling tends not to be misleading.

7 User evaluation

We perform a user evaluation where we compare our method with the array of 2D slices of the data corresponding to the faces of the hypercube organized in a similar to the display of a Scatter Plots Array (SPA). We choose SPA to study the difference that the structure display that our hypercube tool (HyperVis) provides. We considered data corresponding to a scalar field of four variables $y = f(x_1, x_2, x_3, x_4)$ exhibiting a critical point in a neighborhood of a point. A group of ten students (graduate students and 3 undergraduate students enrolled in a math graduate course) where asked to classify the point as a maximum, minimum or a saddle point in the 4D and in 3D subspaces. The students were asked to do this task with SCA with HyperVis and to describe how they selected the answers. The participants did not have much previous experience with either tool. They also were asked to rank statements about each tool according to the scale:

5= strongly agree, 4 = agree, 3 = neither agree nor disagree, 2= disagree and 1 = strongly agree.

8 Results

The summary of the ranking of the statements is the following.

Statement	SPA—mean (median)	HyperVis—mean (median)
The visualization tool was easy to use	3.6 (4.0)	4.1 (4.0)
The visualization tool helped me to understand the structure of 3D slides of the data	3.9 (4.0)	4.3 (4.0)
The visualization tool helped me to understand 4D structure of the data	3.4 (3.5)	4.4 (4.0)
It was interesting to interact with the data using this visualization tool	3.9 (4.0)	4.4 (4.5)

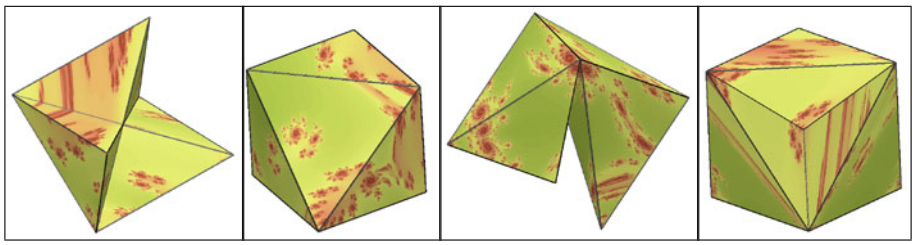

Fig. 11 Different unfoldings of a "hypercube corner" polytope with Henon map data

Most of the descriptions of how the students completed the task focused in the mathematical aspects of the data but a few of them gave useful suggestions regarding the labeling of the data in our tool. In response to their comments we have added new labeling capabilities to HyperVis. The number of students who gave correct answers for the task was eight for both tools. We noticed that the mean (or median) response to the questions regarding the 3D structure of the data do not defer much. In the questions regarding the 4D structure, the users appear to favor HyperVis over the SPA visualization.

9 Conclusions and future work

Our unfolding method gives an appealing environment for the study of scalar functions of many variables. The interactive ability to change the unfolding allows the user to interact directly with 4D geometry in an easy, dynamic, and intuitive display. The first time user gets familiarized very fast with the setup and the parallel with unfolding in 3D and orthographic projections helps to build the necessary intuition for the added dimension.

The foundation of the display is a structured collection of 2D slices. The geometry of all the different unfoldings gives fast access to the transitions and connections between the slices in different ways. The unfoldings are a synopsis of navigation through the data with lower dimensional slices. They give us detailed mapped images of the 4D space. The trade-off is that we display lower dimensional slices of the data in a structured interactive environment. See Miller and Gavosto (2004) for techniques of how the data in the "interior" of the cubes could be recovered. Another way of looking at the "interior" of the hypercube is to consider additional polytopes with faces crossing the data in additional directions. The pictures in Fig. 11 correspond to a "corner" of a hypercube with Henon map data.

Future work will center on selecting the optimal polytope and color map for a given scalar function. We are also planning to combine our previous work in Miller and Gavosto (2004) with the unfolding method creating an environment to keep track of changes from the higher dimensional space to the 4D space.

Acknowledgments We thank Wonpil Im from the Center for Bioinformatics from the University of Kansas for providing us with his data (Sect. 5.2) and for meeting us to discuss visualization needs. The Hurricane Isabel data were produced by the Weather Research and Forecast (WRF) model, courtesy of NCAR and the US National Science Foundation. We also wish to thank the anonymous reviewers for their suggestions which led to a much improved paper.

References

Aguilera Ramírez A., Pérez Aguila R (2002) Presenting methods for unraveling the first two regular 4D polytopes (4D simplex and the hypercube). In: Proceedings of the first international symposium on, Cyber Worlds, IEEE, pp 439–446

Asimov D (1985) The grand tour: a tool for viewing multidimensional data. SIAM J Sci Stat Comput 6(1):128–143

Bajaj CL, Pascucci V, Rabbiolo G, Schikore DR (1998) Hypervolume visualization: a challenge in simplicity. In: Proceedings of the 1998 IEEE symposium on volume visualization, pp 95–102

Banchoff T (1990) Beyond the third dimension: geometry, computer graphics and higher dimensions. Scientific American Library, New York

de Oliveira MCF, Levkowitz H (2003) From visual data exploration to visual data mining: a survey. IEEE Trans Visual Comput Graph 9(3):378–394

dos Santos SR, Brodlie KW (2002) Visualizing and investigating multidimensional functions. In: Proceedings of the symposium on data visualisation 2002 (VISSYM '02). Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp 173–182

Feiner S, Beshers C (1990) Worlds within worlds: metaphors for exploring *n*-dimensional virtual worlds. In: Symposium on user interface software and technology, Proceedings of the 3rd annual ACM SIGGRAPH symposium on User interface software and technology, Snowbird, Utah, United States, pp 76–83

Hanson A, Heng PA (1992a) Illuminating the fourth dimension. IEEE Comput Graph Appl 12:54–62

Hanson A, Heng PA (1992b) Four-dimensional views of 3D scalar fields. In: IEEE Visualization, Proceedings of the 3rd conference on Visualization '92, pp 84–91

Inselberg A, Dimsdale B (1990) Parallel coordinates: a tool for visualizing multi-dimensional geometry. In: IEEE visualization proceedings of the 1st conference on visualization '90, pp 361–370

Love AL, Pang A, Kao DL (2005) Visualizing spatial multivalue data. IEEE Comput Graphics Appl 25(3):69–79

Mihalisin T, Timlin J, Schwegler J (1991) Visualizing multivariate functions, data, and distributions. IEEE Comput Graph Appl 11(3):28–35

Miller J, Gavosto EA (2004) The immersive visualization probe for exploring n-dimensional spaces. IEEE Comput Graph Appl :2–11

Turney P (1984) Unfolding the tesseract. J Recreat Math 17(1):1–16

van Wijk J.J., van Liere R. (1993) HyperSlice—visualization of scalar function of many variables. In: Proceeding of the IEEE conference on visualization (Visualization 93), pp 119–125

Vos FM, van Gelder RE, Serlie IWO et al (2003) Three-dimensional display modes for CT colonography: Conventional 3D virtual colonoscopy versus unfolded cube projection.. Radiology 228(3):878–885