

Visualization of Multi-Dimensional Scalar Functions using HyperSlice

Robert van Liere
Centrum voor Wiskunde en Informatica
P.O. Box 4097, 1009 AB Amsterdam, The Netherlands
e-mail: robert1@cwi.nl
Jarke J. van Wijk
Netherlands Energy Research Foundation ECN
P.O. Box 1, 1755 ZG Petten, The Netherlands
e-mail: vanwijk@ecn.nl

HyperSlice is a new method for the visualization of scalar functions of many variables. With this method the multi-dimensional function is presented in a simple and easy to understand way in which all dimensions are treated identically. The central concept is the representation of a multi-dimensional function as a matrix of orthogonal two-dimensional slices. These two-dimensional slices lend themselves very well to interaction via direct manipulation, due to a one to one relation between screen space and data space. Several interaction techniques, for navigation, the location of maxima, and the use of user-defined paths, are provided.

In this article we present two extensions to the HyperSlice and show how the method can be used in practice.

1 INTRODUCTION

1.1 Problem

Scalar functions of several variables are often used in science and engineering. These functions can be denoted as $f(\mathbf{x}) = f(x_1, x_2, \dots, x_{n-1}, x_N)$, where \mathbf{x} is a point in N -dimensional space, and x_i is a variable of the i -th dimension. Scalar functions can be analytically defined, or can be the result of a simulation or measurements.

Visualization is an important tool for their analysis. Two types of use can be discerned. First, the function can be precomputed at a set of discrete points. The visualization then boils down to a visual inspection of a data set in which calculation of new function values is limited to interpolation of the values in the data set. Second, the function can be computed during the visualization. Here the user specifies what he is interested in, and a separate computation module generates the data. This approach is an example of *computational steering*: the

simulation runs continuously, while the user simultaneously views the results and changes input parameters. This is highly efficient for multi-dimensional functions, because when the number of dimensions is large, the precomputation of data on a fine grid is prohibitively expensive in terms of processing power and memory requirements. However, computational steering obviously assumes that the function can be evaluated fast enough for interactive use.

The complexity of the visual presentation of multi-dimensional functions depends heavily on N . For $N = 1$ a simple graph suffices, for $N = 2$ two-dimensional color images or three-dimensional mountain plots are routinely used. The visualization of scalar functions of three variables is known as volume rendering, and is an important and active area of research. Many techniques have been proposed for their visualization [?].

The direct visualization of scalar functions of more than three variables is more complex, because the human mind is not able to imagine high-dimensional objects. With some effort, four-dimensional functions can be imagined as time-varying three-dimensional functions, but if $N > 4$ hardly anybody can produce mental images of such functions.

One solution to the presentation of functions with $N > 4$ is therefore to fix the value of a number of variables so that the number of free variables is lower than four, and then to use a standard visualization technique. In other words, a slice of the data is selected and visualized.

1.2 Previous work

Several researchers have proposed methods for the visual representation of multi-dimensional data and interaction on these representations. Although much progress has been made recently, most of the proposed solutions still do not seem to be satisfactory. All solutions compromise on the dimensionality, granularity and legibility of the representation. We make a crude classification of existing multi-dimensional data representations, by distinguishing between techniques using hierarchy, using iconic representations, and using scatterplot matrices.

The central idea of representation techniques involving hierarchy is to select a small number of dimensions and display these within a space of higher dimension. Young, Kent and Kufeld [?] have developed the HyperSpace method for visualizing and interacting with multivariate data sets. First, this method uses interpolation to dynamically calculate and display a smoothly changing sequence of interpolations between two three-dimensional spaces. In effect, this is moving a three-dimensional object through a six-dimensional space. Second, this method uses residualization to redefine two three dimensional spaces as a linear combination of six or more variables. Residualization allows the user to move the three dimensional space into any N -dimensional space, with $N > 6$. Other authors have suggested variants and enhancements to this hierarchical representation technique [?, ?].

In the Exvis project [?] icons with settable attributes are used to represent data. The original Exvis icon is a five limbed stick figure with controllable limb-angle, size, thickness and color. The authors show how this representation can be used to represent over twenty different data parameters. Presenting multi-dimensional data as a very large collection of icons produces a texture. Many

other icons can be conceived to represent similar mappings. Other authors have also used icons and/or textures for representation [?, ?].

Scatterplot matrices [?] have been used extensively by the statistics community. Assuming an N -dimensional data set, a scatterplot matrix is an arrangement of $(N^2 - N)/2$ pairs of two dimensional plots in which rows and columns of the matrix share common scales. Dependencies between variables can be obtained by scanning a row (or column) and visualizing how one variable is plotted against all others. Various interaction techniques have been proposed on the scatterplot matrix representation. For example, brushing is a simple but effective techniques that enables users to select groups of data points which are then highlighted in other projections. Cleveland [?] argues that scatterplot matrix representations augmented with highly interactive techniques provide more information than a simple sequence of scatterplot matrices themselves.

Both the hierarchical methods and the icon based methods provide sophisticated representations of continuous data. However, most of these representations are primary intended for a single static display, or a sequence of displays with limited interaction. The Worlds within Worlds concept of Feiner and Beshers [?] is an important exception.

Scatterplot matrices provide simple representations of discrete data. An advantage is that all dimensions are treated identically, no more or less arbitrary decision is expected from the user how the data must be structured for presentation purposes. Furthermore, interaction techniques on this representation can be added relatively easily.

1.3 Overview

Our basic conjecture is that in scientific visualization, representation and interaction are equally important and that they are closely related. The visual representation should be such that the user can understand the behavior of the function, as well as easily interact on this representation.

The first choice to be made is on the dimensionality of the basic visual representation. The use of sophisticated three-dimensional techniques, possibly enhanced with animation and color, seems natural, because as many as possible dimensions are shown simultaneously. This solution is optimal if the function or data is three-dimensional. However, if more dimensions have to be visualized, only a selection can be shown, and hence navigation (e.g. modification of the values of variables that are fixed for a single representation) becomes essential. Here we run into problems. First, although significant progress has been made, current techniques for volume rendering are too slow for direct manipulation. Second, such volume renderings are more difficult to interpret than simpler representation forms, and often tuning of the settings of thresholds, opacity mappings, etc. is required. Finally, three-dimensional interaction is not trivial.

We therefore use two-dimensional slices as the basic visual representation. The geometric coordinates denote two variables, a gray or color value denotes the value of the function. Rendering is fast, visual interpretation is easy, and interaction is direct, because of the one to one relation between the screen space and data space.

However, a single slice only shows a very limited subset of the multi-dimensional space. We therefore developed *HyperSlice*, a new method for the visualization of multi-dimensional functions. With this method the function is presented in a simple and easy to understand way, all dimensions are treated identically, and interaction via direct manipulation of the representation is easy and effective.

The format of this paper is: first, we review the underlying concepts of the HyperSlice representation and basic interaction methods with this representation. Second, we provide various examples of interactive visualization techniques that augment the direct interaction with higher level information. Finally, we present two applications.

A more elaborate description of HyperSlice can be found in [?]. This paper introduces two extensions. In particular, we present interaction techniques based on rotation and contouring.

2 HYPERSLICE

The central concept is the representation of a multi-dimensional function as a matrix of orthogonal two-dimensional slices. These two-dimensional slices lend themselves very well to interaction via direct manipulation, due to a one to one relation between screen space and data space. For example: users can translate and rotate through the data space by simply pointing the mouse and dragging the two-dimensional slices. Furthermore, higher level interaction tools are integrated into HyperSlice, resulting in a powerful environment for the analysis of multi-dimensional scalar functions.

In this section we introduce the definition of the current point and of a slice, then we use these definitions to discuss the HyperSlice representation. Finally, we show how interaction is expressed.

2.1 Definitions

We assume that the focus of the user is on a single N -dimensional point of interest, $\mathbf{c} = (c_1, c_2, \dots, c_N)$, which is called the current point. The width of the focus is a set of scalar values w_i , with $i = 1, \dots, N$. The range of values of interest for dimension i is the interval $R_i = [c_i - w_i/2, c_i + w_i/2]$.

A *two-dimensional slice* $S_{k,l}$, with $k \neq l$, is a visual representation of $f(\mathbf{x})$, where $x_k \in R_k$ and $x_l \in R_l$ vary and the other x_i are equal to c_i . The horizontal axis of the slice is aligned with x_k , and the vertical axis with x_l .

A *one-dimensional graph* G_k is a graph of $f(\mathbf{x})$ where $x_k \in R_k$ and all other x_i are equal to c_i . In this case the horizontal axis is aligned with x_k , and the vertical axis is aligned with $f(\mathbf{x})$.

2.2 HyperSlice Representation

A HyperSlice representation is a $N \times N$ matrix of panels $\langle i, j \rangle$, with $1 \leq i, j \leq N$. Panels on the diagonal of the matrix contain graphs G_i , panels at off-diagonal positions contain slices $S_{i,j}$. The ranges R_i are enumerated along the horizontal and vertical axes of each of the panels. As a result, the current point will always be centered in the middle of the panel.