An Experimental Study of Old and New Depth Measures*

John Hugg† Eynat Rafalin† Kathryn Seyboth† Diane Souvaine†‡

Abstract

Data depth is a statistical analysis method that assigns a numeric value to a point based on its centrality relative to a data set. Examples include the half-space depth (also known as Tukey depth), convex-hull peeling depth and L_1 depth. Data depth has significant potential as a data analysis tool. The lack of efficient computational tools for depth based analysis of large high-dimensional data sets, however, prevents it from being in widespread use.

We provide an experimental evaluation of several existing depth measures on different types of data sets, recognize problems with the existing measures and suggest modifications. Specifically, we show how the L_1 depth contours are not indicative of shape and suggest a PCA-based scaling that handles this problem; we demonstrate how most existing depth measures are unable to cope with multimodal data sets and how the newly suggested proximity graph depth addresses this issue; and we explore how depth measures perform when the underlying distribution is not elliptic.

Our experimental tool is of independent interest: it is an interactive software tool for the generation of data sets and visualization of the performance of multiple depth measures. The tool uses a hierarchical render-pipeline to allow for diverse data sets and fine control of the visual result. With this tool, new ideas in the field of data depth can be evaluated visually and quickly, allowing researchers to assess and adjust current depth functions.

1 Introduction

Over the last decade, statisticians have developed the concept of *data depth* as a method of multivariate data analysis that is an attractive alternative to classical statistics [47, 36, 35]. In this era, massive data sets are the norm, whether the discipline is financial markets, human biology, molecular biology or sociology. *Data depth* provides the ability to analyze, quantify and visualize these data sets without making prior assumptions about the probability distribution from which they come.

Proposed *data depth* metrics are inherently geometric, with a numeric value assigned to each data point that represents its *centrality* within the given data set. The *depth median*, the point of maximal depth, is the depth based estimator for the center of the data set. *Depth contours* can be used to visualize and quantify the data (see, e.g. [36]).

Data depth remains a relatively new field. A number of *data depth* measures have been proposed, analyzed, and, in some cases, coded, and new *data depth* measures continue to be proposed. Examples include *convex-hull peeling depth* [13, 4], *half-space depth* [21, 56], *simplicial depth* [34], *regression depth* [44, 48] and L_1 *depth* [57]. Despite all of the theoretical analysis and individual experiments, there is no conclusive evidence as to which depth measure should be applied to which data sets. One key problem is that several *data depth* measures which perform beautifully in two dimensions quickly become impractical as the dimensionality increases. Others that can be effectively computed in higher dimensions are not statistically significant and produce output that can be misleading.

The goal of this work has been to develop **Depth Explorer** as an experimental platform for analysis and visualization both of random data sets and of pre-existing data sets using multiple different *data depth* measures. In particular, **Depth Explorer** includes implementations of standard *data depth* measures such as *half-space depth*, *convex hull peeling depth*, and L_1 *depth*. It also includes implementations of the newer class of *proximity graph data depth* measures [42, 43] such as *Delaunay depth*, *Gabriel Graph depth*, and β-*skeleton depth*. **Depth Explorer** allows the user to analyze and visualize a data set using each of these measures. More importantly, however, the **Depth Explorer** experimental platform allows the comparison of *depth measures*. In particular, our testing of L_1 *depth* demonstrated the poor results that L_1 *depth* generates on certain types of data set. And yet L_1 *depth* has been one of the few *data depth* metrics known to be computable in time that is linear in dimension, rather than exponential. We have developed an enhanced version of L_1 *depth* that we call L_1 *scaling depth* that retains the computational efficiency of L_1 *depth* but produces much better output.

The paper reports not only on the development, availability, and features of the **Depth Explorer**

*This material is based upon work supported by the National Science Foundation under Grant No. CCF-0431027.

†Department of Computer Science, Tufts University, Medford, MA 02155. {jhugg,erafalin,kseyboth,dls}@cs.tufts.edu

‡2005-2006 MIT Visiting Scientist & Radcliffe Institute Fellow.

Sandbox but also on the results of the experiments we have performed using this platform. Section 2 provides background information on the *data depth concept*, including applications and examples of depth functions. Section 3 provides technical information about **Depth Explorer**. Sections 4, 5 and 6 present analysis of data sets and depth measures using **Depth Explorer** and Section 7 describes future work.

2 Data Depth

A *data depth* measures how deep (or central) a given point $x \in \mathbb{R}^d$ is relative to F, a probability distribution in $\mathbb{R}^d$, or relative to a given data cloud.

Sections 2.1 and 2.2) introduce general principles of data depth irrespective of the particular data depth functions chosen . Definitions of four types of data depth functions, with different features and computational complexities, are presented in Section 2.3).

2.1 General Concepts

The following concepts apply to the data depth methodolgy and distinguish it from other statistical methods.

- **Non-parametric methodology:** Scientific measurements can be viewed as sample points drawn from some unknown probability distribution, where the analysis of the measurements involves computation of quantitative characteristics of the probability distribution (*estimators*), based on the data set. If the underlying distribution is known (for example normal distribution, log-normal distribution, Cauchy, etc.), the characteristics of the data can be computed using methods from classical statistics. However, in most real life experiments the underlying distribution is not known. The concept of data depth requires *no assumption* about the underlying distribution and data is analyzed according to the relative position of the data points.

- **Center-outward ordering of points:** The data depth concept allows the creation of a multivariate analog to the univariate statistical analysis tool of *rank statistics*. *Rank statistics* is based on the ordering of one-dimensional observations, where the order *reflects extremeness, contiguity, variability or the effect of external contamination and provides a parameter estimation method* [4]. If $\mathcal{S} = \{X_1, \ldots X_n\}$ is a sample of observations in $\mathbb{R}^1$ then the order statistics is defined as $\{X_{[1]}, \ldots X_{[n]}\}$ where $X_{[1]} \leq X_{[2]} \ldots \leq X_{[n]}$.[1] In higher dimensions the order of multivariate data is not well defined, and several ordering methods were suggested (e.g. [4]). The data depth concept provides a method of extending order statistics to any dimension by ordering the points according to their depth values.

- **Application to multivariate (high-dimensional) data sets:** The concept of data depth is defined with respect to points in Euclidean space in *any* dimension, thus enabling the derivation of multivariate distributional characteristics of a data set. The methodology enables the exploration of high dimensional data sets using simple two-dimensional graphs that are easy to visualize and interpret, and using quantitative estimators.

- **Robustness:** In the statistical analysis of datasets, observations that deviate from the main part of the data (*outliers*) can have an undesirable influence on the analysis of the data. Many depth functions are "robust against the possibility of one or several unannounced outliers that may occur in the data and yield reasonable results even if several unannounced outliers occur in the data" [46]. For example, "by adding k bad data points to a data-set one can corrupt at most the k-outermost (half-space) depth contours while the ones inside must still reflect the shape of the good data" [12].

2.2 Depth Contours, Median and Mode

The **median** is a depth based estimator for the center of a data set. The median of a set S under some depth measure $D : S \to \mathbb{R}$ is the set of points M such that $\forall p \in M,\ D(p) \geq D(q)\ \forall q \in S$. This definition supports the possibility that several points will tie for the deepest depth. The use of a single point or group of points as the median relies on the assumption of unimodality that is common in depth measures. **Depth Explorer** highlights the median points, visualizing the center of a data set.

The **mode** of a set is the most common value [59]. We use the term mode flexibly and refer to a *bimodal* (or *multimodal*) distribution or point set as one having two (or more) local maxima. The multiple maxima can be created, for example, from overlaying two different unimodal distributions. Often clustering algorithms are used to detect the points associated with each mode of the data set. This association, however, does not necessarily attribute a data point to the center of the distribution where it originated. For example, points located between two centers and far from each could be assigned to either of the two clusters.

Depth contours [55] are nested regions of increas-

[1] An alternative ranking order is from the outside inward, where the deepest point equals the median.

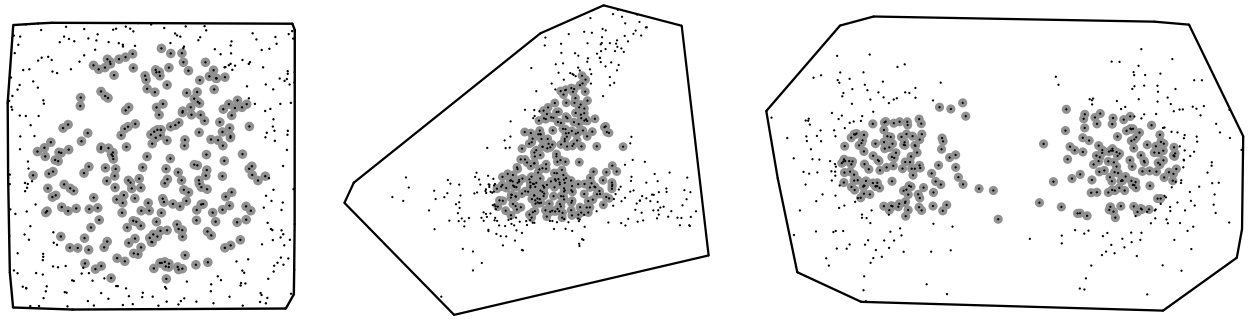


Figure 1: 50% deepest points highlighted for different distributions calculated using Half-space Depth. All distributions contain 500 points.

ing depth and serve as a topological map of the data. Let $D_F(x)$, $x \in \mathbb{R}^d$, be the value of a given depth function for point x with respect to a probability distribution F. The **region enclosed by the contour of depth** t is the set $R_F(t) = \{x \in \mathbb{R}^d : D_F(x) \geq t\}$. The α **central region,** C_α $(0 \leq \alpha \leq 1)$ is, for well behaved distributions, the region enclosed by the contour of depth t_α $C_\alpha = R_F(t_\alpha)$, where $P\{x \in \mathbb{R}^d : D_F(x) \leq t_\alpha\} = \alpha$ [36]. Well-behaved depth functions produce depth contours that are affinely equivariant, nested, connected and compact [61]. Contours have applications in visualization and quantification of data sets.

In **Depth Explorer** highlighting the $100\alpha\%$ deepest points visualizes the α central region. See, e.g., Figure 1.

If the underlying distribution is elliptic then the *convex hull* containing the $\alpha\%$ deepest points is a simplified sample estimate of the boundary of the contour [35]. However, in such a case the resulting contour may also contain shallower data points, that are not the $\alpha\%$ deepest points. In real life data sets usually the underlying probability distributions is not known, but if the data set is unimodal and convex then it is reasonable to assume that the underlying probability distribution is elliptic.

As a method of comparing the performance of a variety of depth measures, **Depth Explorer** can display the convex hulls enclosing the $20\%, \ldots 100\%$ deepest points under each of these measures, visualizing the type of contour produced by each measure. See Figure 2.

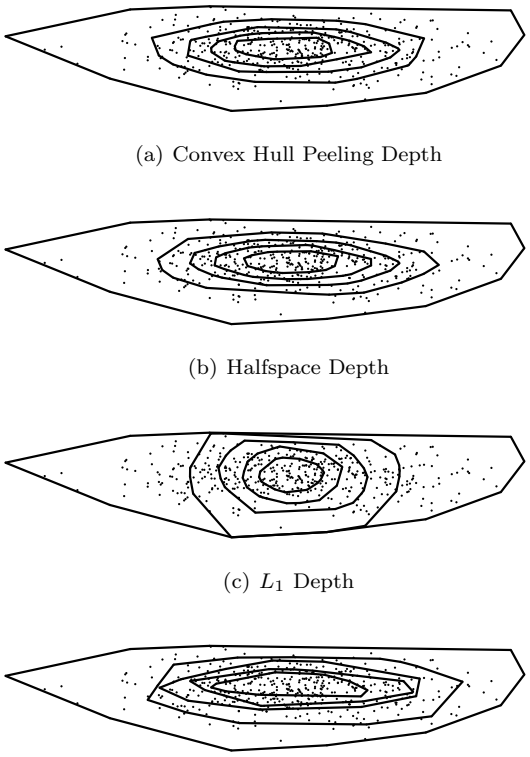
2.3 Depth Measures

We present four types of depth functions, with different features and computational complexities, that are all implemented in the **Depth Explorer Sandbox**. The *convex-hull peeling depth* is one of the early depth measures studied by the computational geometry community, but it lacks many statistical properties. *Half-space depth* is probably the best-known depth measures in the computational literature and has attractive statistical properties. The L_1 *depth* does not possess many desirable statistical properties, but the simplicity of its computation makes it useful for certain application of data depth. The newly suggested *proximity depth* was developed as a depth measure that is efficient to compute in high dimensions.

2.4 Convex-Hull Peeling Depth

DEFINITION 2.1. *The* **convex-hull peeling depth** *[13, 4] of a point X_k with respect to a data set $\mathcal{S} = \{X_1, \cdots X_n\}$ in $\mathbb{R}^d$ is the level of the convex layer to which X_k belongs. The level of the convex layer is defined as follows: the points on the outer convex hull of $\mathcal{S}$ are designated level one and the points on the kth level are the points on the convex hull of the set $\mathcal{S}$ after the points on all previous levels were removed (see Figure 2(a)).*

Convex-hull peeling depth is appealing because of the relative simplicity of its computation. However it lacks distributional properties and is not robust in the presence of outliers. The convex-hull peeling layers can be computed in the plane in optimal $\Theta(n \log n)$ time using Chazelle's deletions-only dynamic convex-hull algorithm [8] and in higher dimensions using iterative applications of any convex-hull algorithm (the complexity of computing the convex hull for a set of n points in $\mathbb{R}^d$ *once* is $O(n \log n + n^{\lfloor \frac{d+1}{2} \rfloor})$ for even d [41] and $O(n \log n + n^{\lfloor \frac{d}{2} \rfloor})$ for odd d [9] and it can be applied as many as $O(n)$ times to compute the entire set of peeling layers). The vertices of all convex layers can be computed in $O(n^{2-\gamma})$ time for any constant $\gamma < 2/(\lfloor d/2 \rfloor^2 + 1)$ [6].



(a) Convex Hull Peeling Depth

(b) Halfspace Depth

(c) L_1 Depth

(d) Delaunay-Based Proximity Depth

Figure 2: 20%, 40%, 60%, 80% and 100% contours for a data set consisting of 500 points, normally distributed, width scaled by 5. Note that the L_1 depth contours appear more round that is warranted by the distribution, while the Delaunay-Based Proximity Depth contours are the most elongated.

2.5 Half-space Depth

DEFINITION 2.2. *The* **half-space depth** *[21, 56] (in the literature sometimes called location depth or Tukey depth) of a point x relative to a set of points $\mathcal{S} = \{X_1, ..., X_n\}$ is the minimum number of points of $\mathcal{S}$ lying in any closed half-space passing through x (see Figure 2(b)).*

The half-space depth has many attractive statistical properties [61, 12]. However, its computation is exponential in dimension. The half-space depth of a *single point* in $\mathbb{R}^2$ can be computed in $O(n \log n)$ time [52], matching the lower bound [2]. The set of half-space *depth contours* can be computed in the plane in optimal $\Theta(n^2)$ time [38] (expanding upon ideas in [11]), or using other methods [52, 25, 29], including computation of the depth contours in $\mathbb{R}^d$ using parallel arrangement construction [17]. Theoretical and practical algorithms for computing or approximating the deepest contour relative to the data set (also known as the *Tukey median*) in two and higher dimensions exist for some time [52, 49, 51, 53, 58, 37, 30, 1], culminating in an $O(n \log n)$ expected time randomized optimal algorithm, which can be extended to any fixed higher dimension d, yielding an $O(n^{d-1})$ expected time algorithm [7]. The problem of computing the half-space depth is NP-hard for unbounded dimension [3].

2.6 The L_1 Depth

DEFINITION 2.3. *The L_1* **depth** *(L_1D) [57] of a point x with respect to a data set $\mathcal{S} = \{X_1, \cdots X_n\}$ in $\mathbb{R}^d$ is one minus the average of the unit vectors from x to all observations in $\mathcal{S}$:*
$L_1D(S, x) = 1 - \|\overline{e}(x)\|$, *where* $e_i(x) = \frac{x - X_i}{\|x - X_i\|}$, $\overline{e}(x) = \frac{\sum_{i=1}^{n} \eta_i e_i(x)}{\sum_j \eta_j}$. *$\eta_i$ is a weight assigned to observation X_i (and is 1 if all observations are unique), and $\|x - X_i\|$ is the Euclidean distance between x and X_i (see Figure 2(c)).*

Intuitively, the L_1 median of a cloud of points in $\mathbb{R}^n$ is the point that minimizes the sum of the Euclidean distances to all points in the cloud. The L_1 depth of a point can be summarized by the question, "How much does the cloud need to be skewed before this point becomes the L_1 median?". The L_1 depth ranges between 0 and 1. It is fast and easy to compute in any dimension, contributing to its appeal for study of large high-dimensional data sets. The L_1 depth is non-zero outside the convex hull of the data set and therefore can be used to measure within-cluster and between-cluster distance (see Section 7.1). However, it lacks many statistical properties.

2.7 Proximity Depth

For any proximity graph the proximity graph depth is defined using a point's minimum path length along graph edges to the convex hull of the data set $\mathcal{S}$.

DEFINITION 2.4. *The* ***[proximity graph] depth*** *of a point x relative to a set $\mathcal{S} = \{X_1 \dots X_n\}$ is the minimum number of edges in the [proximity graph] of $\mathcal{S}$ that must be traversed in order to travel from x to any point on the convex hull of $\mathcal{S}$ (see Figure 3).*

Proximity graphs are graphs in which points close to each other by some definition of closeness are connected [24]. We concentrate our analysis on the *Delaunay triangulation* [16] and *β-skeletons* [28] which are a parameterized family of proximity graphs, which include as a special case the *Gabriel graph* and the *relative neighborhood graph*. We denote by $\delta(p, q)$ the Euclidean distance between points p and q.

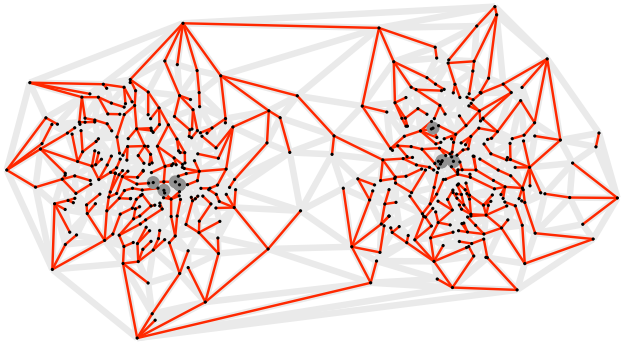


Figure 3: Exploring a proximity depth calculation. The edge-distance from each data point to a convex-hull point is illustrated using highlighted trees. Two normal clouds with 250 points separated horizontally by 6 standard deviations.

DEFINITION 2.5. *The **Delaunay triangulation** (DT) of a d-dimensional point set S is the simplicial decomposition of the convex hull of S such that the d-sphere defined by the points of every simplex in the decomposition contains no point $r \in S$ [16].*

This decomposition is the dual of the *Voronoi diagram* and is unique for every set of points [14].

DEFINITION 2.6. *The β **skeleton** of a point set S in $\mathbb{R}^d$ is the set of edges joining β-neighbors.*
*Points p and q are **lune-based β-neighbors** for $\beta \geq 1$, iff the lune defined by the* intersection *of the spheres centered at $(1-\frac{\beta}{2})p+\frac{\beta}{2}q$ and $(1-\frac{\beta}{2})q+\frac{\beta}{2}p$, each with radius $\frac{\beta}{2}\delta(p,q)$, contains no point $r \in S$.*
*Points p and q are **circle-based β-neighbors** for $\beta \geq 1$, iff the lune defined by the* union *of the two spheres of radius $\frac{\beta}{2}\delta(p,q)$ contains no point $r \in S$.*
*Points p and q are **β-neighbors** for $\beta < 1$, iff the lune defined by the intersection of the two sphere of radius $\frac{\beta}{2}\delta(pq)$ which contain p and q in their boundary contains no point $r \in S$ (for $\beta < 1$ the lune-based and circle-based neighbors are identical).*

For $\beta > 1$ the **lune-based β-skeletons** are planar and monotonic with respect to β: $G_{\beta_1}(S) \subset G_{\beta_2}(S)$, for $\beta_1 < \beta_2$. The *Gabriel graph* (GG) [18] is the lune-based 1-skeleton while the *relative neighborhood graph* (RNG) [54] is the lune-based 2-skeleton. The **circle-based β-skeletons** for $\beta > 1$, are not necessarily planar and have a reverse monotonic relation with respect to β: $G_{\beta_1}(S) \subset G_{\beta_2}(S)$, for $\beta_1 > \beta_2$.

For $\beta < 1$, as β becomes smaller, the β skeleton tends towards the complete graph.

Overall Complexity The depths of all points in a proximity graph can be determined in linear time in the number of edges in the graph by using a breadth-first search (BFS) of the graph, beginning at every point on the convex hull of $\mathcal{S}$ (Figure 3). Assignment of all depths of a point set is accomplished by (1) Computation of the proximity graph; (2) Location of all convex hull points; and (3) Breadth-first search of the proximity graph.

In two dimensions there are optimal $O(n \log n)$ time algorithms to compute the DT [16], the circle-based β-skeletons for $\beta \geq 1$, and the lune-based β-skeleton for $1 \leq \beta \leq 2$ [28, 33, 23]. The lune-based β-skeletons for $\beta > 2$ and the β-skeletons for $\beta < 1$ can be computed in optimal $O(n^2)$ time [28, 23]. Points on the convex hull can be determined in $O(n \log n)$ time [40], for an overall time requirement of $O(n \log n)$ for the DT and $O(n^2)$ or $O(n \log n)$ for the β-skeleton.

In dimensions higher than 2, the DT can be calculated in $O(n^{\lceil \frac{d}{2} \rceil})$ time [15]. The β-skeletons require checking n points for interiority on n^2 lunes, which requires a distance calculation for a total of $O(dn^3)$ time. More efficient algorithms for specific graphs like the GG or RNG or for 3-dimensional space are known [24]. The set of points on the convex hull of the set can be found in $O(mn)$ time, where m is the number of extreme points [39]. Breadth-first search then requires linear time in the size of the proximity graph. Clearly, the time complexity in higher dimensions is dominated by the computation of the proximity graph itself. Assignment of all depths, then, has a total complexity of $O(n^{\lceil \frac{d}{2} \rceil})$ time for Delaunay depth and $O(dn^3)$ time for the β-skeleton depths. The exponential dependence on dimension for calculating Delaunay depth makes it impractical for use in high dimensions.

3 The Depth Explorer Statistical Sandbox

Depth Explorer is an interactive software tool for the generation of data sets and visualization of the performance of multiple depth measures. The tool is aimed for use by statisticians, to visually and quickly evaluate new ideas in the field of depth based statistics, allowing researchers to assess and adjust current depth functions. It was designed to be simple to use. The tool uses a hierarchical render-pipeline to allow for diverse data sets and fine control of the visual result. The Depth Explorer source, executable for Mac OS X and documentation is publicly available and can be found in [22].

3.1 Scene Generation and the Render Tree

Depth Explorer enables automatic generation of data sets and transformation or composition of existing data sets. For each data set, **Depth Explorer** can quickly visualize the behavior of the depth measure on the data. Data sets are defined using a hierarchical representation

of the scene (the *render-tree*) in an XML file. Clicking a toolbar button switches to "live" view and renders the scene to the screen. The user can then switch back into XML editor mode to adjust the scene and then re-render. The changes are almost instantly re-rendered into the window, allowing for interactive experimentation. Generated PDF files can be saved to the filesystem.

Data generation, transformations and visualizations are specified hierarchically using XML tags in the *render tree*, see Figure 4. The tree is rendered starting from the leaf nodes and moving up. Each node is a C++ module that, given the output of its children nodes, modifies or appends the data, then passes it to its parent. Ultimately, the root node, called the *canvas* node describes the dimensions and scale of the PDF output file and renders its childrens' data onto the PDF document. The leaf nodes are data sources, either CSV files with a data set, or a random cloud generator with a set of points. **Depth Explorer** can currently create Gaussian clouds with any number of points with standard deviation 1, as well as uniformly distributed clouds on the range $(-1,-1)$ to $(1,1)$.

Non-leaf nodes modify data or add visualizations. **Depth Explorer** supports affine transformation nodes that can scale, rotate or translate nodes below them. With affine transformations and any number of randomly generated clouds, a broad spectrum of data sets can be generated to test statistical methods.

Visualizations include the display of the $\alpha\%$ contour by highlighting the $\alpha\%$ deepest points. If the set is dense enough this serves as a good visual approximation to the $\alpha/100$ central region, see, e.g., Figure 1. **Depth Explorer** displays the convex hulls enclosing the $20\%, 40\%, 60\%, 80\%$ and 100% deepest points, to visualize a subset of the depth contours, under the assumption that the underlying distribution is elliptic and the *convex hull* containing the $\alpha\%$ deepest points is a simplified sample estimate of the boundary of the contour, see Figure 2. Note that the resulting contour may contain shallower data points, that are not the $\alpha\%$ deepest points.

As **Depth Explorer** uses a modular, tree-based renderer, it is trivial to combine visualizations, even at different points in the hierarchy. By viewing two visualizations on the same data, it is possible to compare depth measures or convey two concepts in one image. Figure 3 illustrates how **Depth Explorer** can help visualize a powerful concept, such as computation of the proximity depth.

3.2 Depth Explorer Construction

The **Depth Explorer** program is currently divided into two sections. The *libdepthengine* library contains all of the code to load a *render-tree* into memory and render it to PDF. This library is written entirely in portable C++. It uses PDFlib from GmbH Software [19] to generate PDF files and it uses Apple's implementation of BLAS [5] and LAPACK [31] to do matrix math and compute eigenvalues. PDFlib is open source and very portable, and thus does not introduce platform dependency. The BLAS and LAPACK routines used are standard and thus implemented in all implementations of BLAS and LAPACK, and do not introduce platform dependence as well.

The **Depth Explorer** GUI links against *libdepthengine* and is responsible for editing XML data, viewing rendered scenes, saving rendered scenes as PDF files and printing. The GUI was developed with Objective-C using the Apple's Cocoa [10] application frameworks. Although the implementation is platform dependent, the amount of code is small and the complexity is very low compared to *libdepthengine*. For example, while the Cocoa frameworks provide support for XML parsing, parsing in **Depth Explorer** is done inside *libdepthengine* in portable C++.

3.3 Performance and Interactivity

Depth Explorer does not render instantaneously even a relatively easy depth measure like L_1. Nonetheless, almost all scenes are rendered within seconds, not minutes, allowing for interactive feedback. On a 1.5 Gigahertz Powerbook G4, rendering most scenes commonly requires 2-5 seconds. Rendering time is heavily dependent on the processing that is done in each node. If no depth-based visualizations are included in the render tree, than even the most complicated scene with thousands of data points will render almost instantly. When depth calculations are involved, computation time is slower.

To render an L_1 50% depth contour on a cloud of 1000 points takes about two seconds. To render the same contour on 10,000 points requires about 5 seconds. While the scale-up should be linear according to the algorithm, the software is tuned to handle large amounts of data, thus the slowdown is only apparent in very large datasets or with very slow computations.

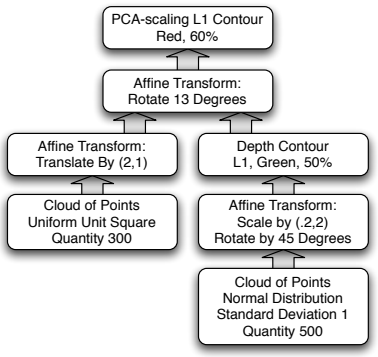
Thus **Depth Explorer** is not a "sit and wait" program: almost all scenes can be rendered in a matter of seconds, not minutes. Conversely, **Depth Explorer** is not a real-time program: it is not fast enough to give dynamic feedback as the user changes parameters to the render tree nodes. Thus a render button is required. As more (slower) depth measures[2] and complex visu-

[2]Depth Explorer's current implementation of Halfspace Depth

XML Representation of Render Tree

```
<canvas width='6' height='6' margins='.1' minrange='6'>
    <pcabag color='1,0,0' size='.6'>
        <transform angle='13'>
            <transform xtrans='2' ytrans='1'>
                <cloud type='uniform' points='300'/>
            </transform>
            <bag type='l1' color='0,1,0' size='.5'>
                <transform xscale='.2' yscale='2' angle='45'>
                    <cloud type='normal' points='500'/>
                </transform>
            </bag>
        </transform>
    </pcabag>
</canvas>
```

Internal Representation of Render Tree



Final Rendered Output

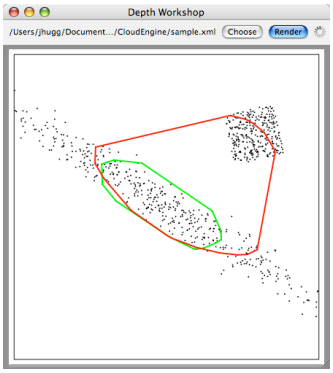


Figure 4: The Render tree form XML to Final Scene. This scene is defined by two source clouds, a uniform square with 300 points and a normal cloud with 500 points. The square is translated and rotated while the normal cloud is scaled and rotated. Both clouds are rotated 13 degrees together. Two depth contours are calculated, one L_1 50% contour on the normal cloud and one PCA-Based L_1 Scaling 60% contour on the entire scene.

alizations are supported, it seems unlikely that **Depth Explorer** will ever become fully realtime. See Section 7 for a discussion of speed improvements.

4 Analysis of the L_1 Depth Measure

The L_1 Depth measure has many statisticians very excited as the computation time is very low. With a fast depth measure available, suddenly the concept of data depth is applicable to many new and more complex problems. L_1 Depth is hoped to be "good enough", that is, its speed will make up for its sub-par results. Still, as L_1 Depth is a new and relatively untested depth measure, it is still unknown just how much quality of result is compromised for speed.

To answer this question, we analyzed the L_1 depth function on unimodal normally distributed data sets that were scaled or rotated. It is desirable that depth contours will be representative of the shape of the of data. However, as we demonstrated visually (see Figure 2) L_1 depth contours do not exhibit this desirable property, while all other tested depth measure do. Since the contour is supposed to be indicative of the shape of the cloud, the L_1 depth measure is not as useful as many other depth measures.

Using **Depth Explorer** to visualize how L_1 Depth performs on various data sets, we discovered that, under the L_1 depth, a point whose distance to the median point is small is more likely to have higher depth than the same point in other depth measures. Consequently, L_1 depth contours tend to be more circular than desired.

As L_1 depth is most successful on hyper-spherical data, by performing an affine transformations for non-spherical data we can create a shape that approaches a hypersphere and compute the L_1 depth on this scaled cloud. The depth values for the scaled cloud are then used for the corresponding points in the original cloud.

This works very well when data sets are scaled along an axis, but many times, a data set is elongated or compressed in directions not parallel to an axis. Thus, Principal Component Analysis (PCA) is used to determine the major directions of the data cloud. Scaling along these vectors produces an approximation of a hyperspherical cloud for many data sets (see Figure 5).

DEFINITION 4.1. *The k-PCA-based L_1 depth of a point x with respect to a data set $\mathcal{S} = \{X_1, ...X_n\}$ in $\mathbb{R}^d$ is the L_1 depth of $pca(x)$ with respect to a data set $pca(\mathcal{S}) = \{pca(X_1), \ldots pca(X_n)\}$. Let $v_1, v_2, \ldots v_n$ be the eigenvectors for $\mathcal{S}$, computed using PCA analysis. Then $pca : \mathbb{R}^d \to \mathbb{R}^d$ is defined as follows: (i) rotate $\mathcal{S}$ so that the first k primary vectors coincide with the axes and (ii) For each of the primary k axes, project the points of $\mathcal{S}$ on the axis and scale the projected set such that the .8 and .2 quantiles go to 1 and -1 respectively. (as a result of this transformation 60% of the data fits between -1 and 1 for each of the primary k axis).*

The *PCA-based L_1 depth* of a point x with respect to a data set $\mathcal{S}$ in $\mathbb{R}^d$ is the d-PCA-based L_1 depth. We assume that data sets are centrally symmetric relative to each of the axis and therefore always use the d-PCA-

and Beta-Based Proximity depth are naive and thus have $O(n^3)$ time complexity. This limits these particular depth calculations to about 500 points. This will be improved in a upcoming minor revision.

based L_1 depth.

PCA-scaling-L_1 depth removes much of the aforementioned bias towards points closer to the median when assigning depth values. We used **Depth Explorer** to confirm this visually, see, e.g. Figure 5. Note that the final contour generated approximates the shape of the cloud.

PCA-scaling does not, address the tendency for L_1 depth contours to be rounded in square data clouds and is limited to handling data clouds that are roughly convex (see Section 6).

5 Multimodality and Depth Measures

5.1 Multimodal Distribution

Depth Explorer was tested on multimodal data sets and the behavior of depth functions on these data sets was analyzed. The tested depth functions, convex-hull peeling depth, half-space depth and L_1 depth, were not well suited for analysis of multimodal data sets (see Figure 8). However, the newly suggested proximity depth was found to handle these data sets well, see Figure 8 and 9.

5.2 Median and Seeds

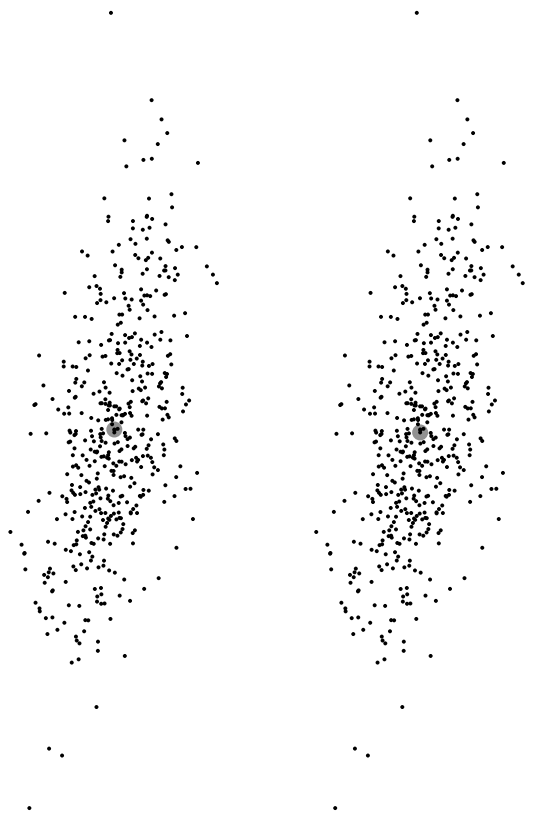
The median, as it was defined in Section 2, is not a good estimator in multimodal situations, because points that are not of maximum depth in the set may in fact be local maxima, local peaks in depth (see Figure 6). Estimating a set using a simple median ignored local maxima that represent several modes. The proximity graph depth allows the definition of an estimator that handles modes of a data set:

DEFINITION 5.1. *A **seed** of a point set S under some depth measure $D : S \to \mathbb{R}$ is a connected set of points $T \subset S$ st $\forall p, q \in T, D(p) = D(q)$ and $\forall r \in S, r \notin T$ adjacent to some $u \in T, D(r) < D(u)$.*

Unfortunately, points of different clusters are not necessarily distinguished by the proximity graph depth measures. Distributions that are too close behave as a single mode; those separated by large empty regions appear unimodal as the dearth of points between the clusters prevents paths from traveling inward quickly from the convex hull.

Quantitative analysis

Tests performed on Depth Explorer verify the abilitiy of the proximity graph seeds to discern unimodality and bimodality quantitatively. Analysis of both bimodal and unimodal planar point sets was performed using a separate program that computed the seeds. Bimodal sets in two dimensions (containing x and y coordinates) were created by combining two 200-point normal distribution sets, each with x and y standard deviations of 10. They began centered at the same x-coordinate, but were then separated in increments of 10. The unimodal sets began with an x-coordinate standard deviation of 10, which was gradually increased to stretch the unimodal set in the x-direction. We plotted the standard deviation of the x-values of the points of local maxima against the x-coordinate range of the set. When the bimodal sets are close, they behave very similarly to the unimodal sets, but as they pull apart their seeds separate, illustrating the bimodal behavior (Figures 7 and 9). This behavior is similar for each of the proximity graph depth measures.



(a) Convex Hull Peeling Depth (b) Halfspace Depth

Figure 6: Highlighted deepest point approximates the depth median. 500 points, Y-axis compressed 70%, rotated 10 degrees.

Algorithms Finding seeds requires a recursive search to locate all points associated with the seed and to locate all points in S that are connected to that seed. The basic process to compute the seeds is that of comparing the depth of point p to the depth of its neighbors in the proximity graph and checking

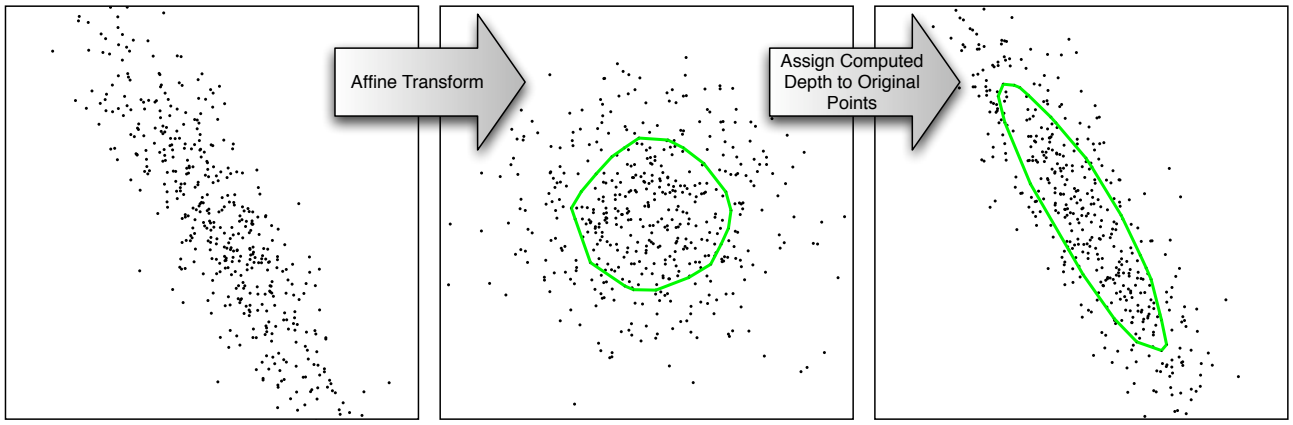


Figure 5: How Affine Scaling Improves the L_1 Depth Results

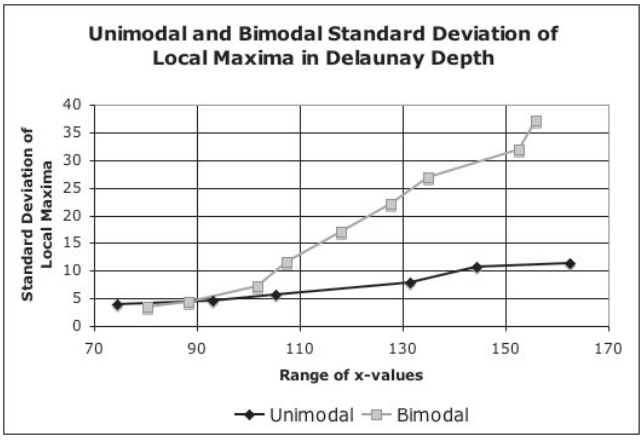


Figure 7: Computation of seeds using the Delaunay depth function for unimodal and bimodal point sets. For every x-value X the unimodal point set was constructed to have comparable width to the width of the bimodal point set, whose separation is X. The standard deviation of the x-values of the local maxima were computed. The results support our assumption, that in a bimodal point set we expect to find wider gaps between the points of local maxima and therefore larger standard deviation compared to unimodal point sets.

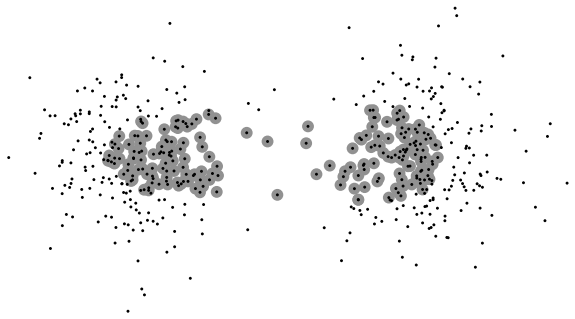
whether it is deeper or of the same depth as its neighbors (*compare(p)*).

One method of computation searches the list of points to locate those that are deeper or of the same depth as their neighbor. In this case, every connected grouping of points must be checked and there can be $O(n)$ such groupings. The *compare(p)* process is called recursively for each point p in a grouping exactly once and obtains a yes/no answer for that point before returning. Because *compare(p)* eliminates points of lower depth than p, it is called at most n times. Each call iterates through some portion of p's adjacency list. Each edge in the graph represents two entries in adjacency lists, and is considered exactly twice, producing an algorithm with a running time that is linear in the size of the proximity graph. The size of the graph is linear in two dimensions and up to quadratic in higher dimensions. The process does not add to the overall complexity, because construction of the graphs requires at least as much time.

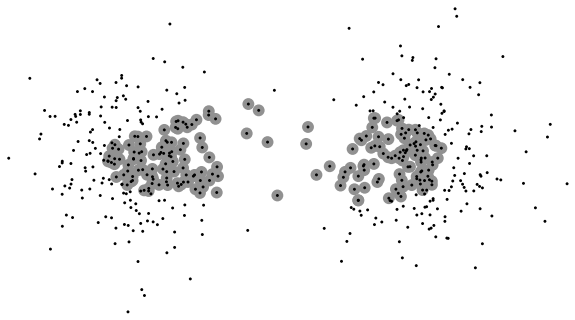
Additional improvements allow the number of seeds to be decreased further (see [42, 43]). For example, only those seeds with **CH-value** great than 2 and over half of the maximum (**CH-point seeds**). The *CH-value* of a seed T is the number of convex hull vertices that can be reached by a path originating at T for which the depths of points visited along the path is strictly decreasing.

5.3 Proximity Graph Depth Function Family

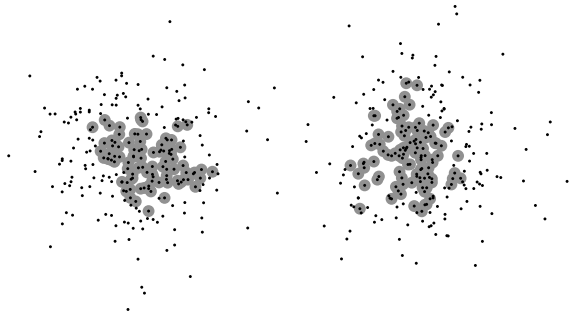
We compared the family β-skeleton depth functions with the DT depth visually, using the **Depth Explorer** (see Figure 9), and quantitatively, with a separate code written in C++, using the LEDA library [32]. For the quantitative analysis 300 normally distributed data sets with 400 points each were generated and the average number of seeds produced was calculated.



(a) Convex Hull Peeling Depth

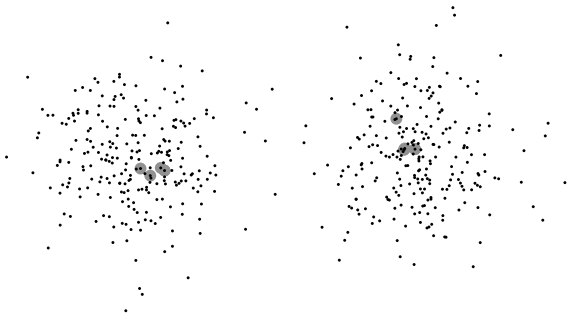


(b) Halfspace Depth

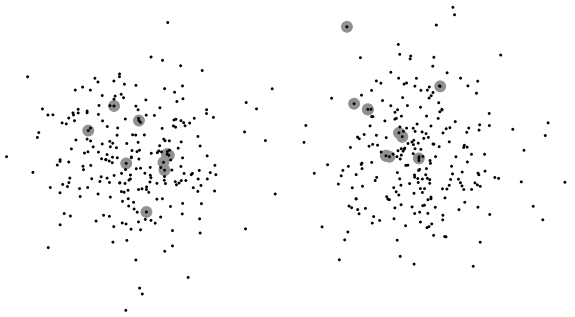


(c) Delaunay Based Proximity Depth

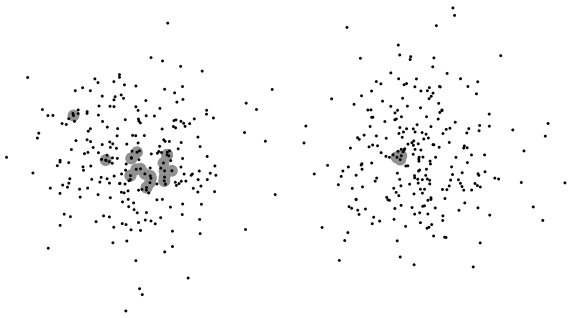
Figure 8: Deepest 30% of the points of a bimodal distribution highlighted with different depth measures. The traditional depth measures' depth is biased towards the space between distributions, while proximity depth measures recognize bimodality. The distribution is two normal clouds with 250 points separated horizontally by 6 standard deviations.



(a) Delaunay-Based Proximity Depth



(b) Gabriel-Based Proximity Depth



(c) β-Based Proximity Depth ($\beta = .970$)

Figure 9: Highlighted seeds for a bimodal data set computing with three proximity depth measures. The distribution is two normal clouds with 250 points separated horizontally by 6 standard deviations.

The average number of CH-point seeds was plotted against the average deepest depth attained by the point set (Figure 10). The locations of the points indicate that the values $\beta = .962$ and $\beta = .970$ best approximate the performance of Delaunay Depth for these two characteristics. The weakness of the GG (1-skeleton) is also visible, as it finds many more seeds in a unimodal data set than the other graphs.

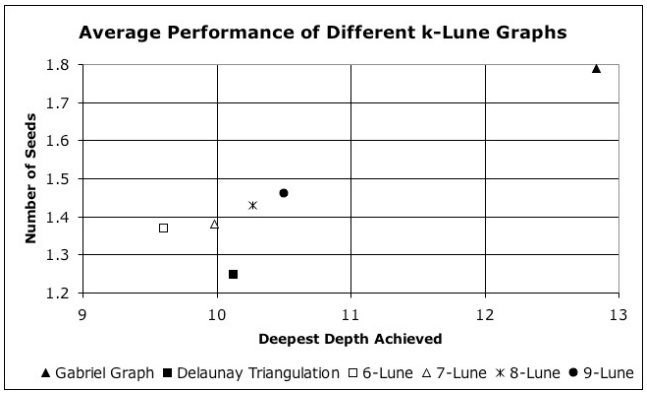


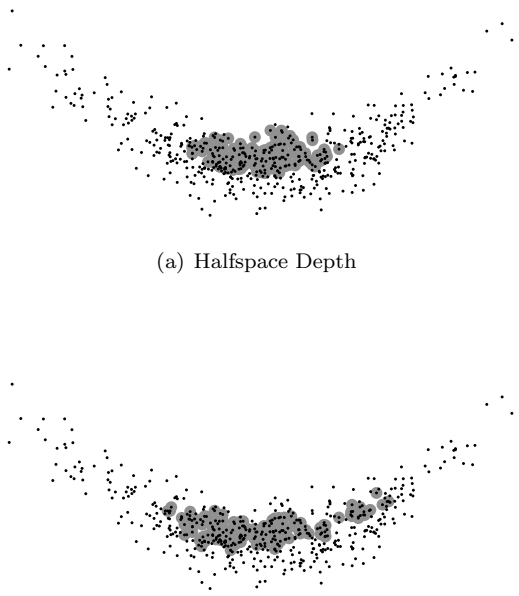
Figure 10: Performance comparison of proximity-depth schemes for 300 normally distributed unimodal point sets with 400 points. A low average number of seeds and a high average number of deepest depth values is desirable. The .962-skeleton and .970-skeleton perform most similarly to the DT The weakness of the GG (1-skeleton) is also visible here, as it finds many more seeds than the other graphs.

6 Non-Elliptic Underlying Distributions

We tested the behavior of depth functions on point sets that did not follow an elliptic distribution. It is desirable that the depth contours will be indicative of the shape of the data cloud. However, for most depth functions, this does not hold. In fact, the *four desirable properties of depth functions*, as suggested by statisticians as a tool to analyze and evaluate depth functions [34, 60, 61, 20] assume that the underlying distribution is unimodal and that the ordering of the points is center-outward.

When the cloud is not round, as in the square uniform plot from Figure 1(a), the depth contours are still round, when they should be more rectangular to match the shape of the data.

If the data points are not in a convex position, but follow a *banana-like* shape, as in Figures 11(a) and 11(b), none of the depth functions capture this shape, not even the proximity graph depth functions which are not convex. We are currently working on a version of the proximity graph depth that will handle this type of distribution.



(a) Halfspace Depth

(b) Delaunay-Based Proximity Depth

Figure 11: 30% deepest points highlighted where the underlying data set is distributed in non-convex position. Traditional depth measures bias towards the inside center of the curve, while proximity-based depth measures have much less bias, and can better represent shape.

7 Future Work

Depth Explorer has great potential as a tool for evaluation and analysis of depth functions, and the work presented here just scratches the surface. Our goal is to continue exploration of depth functions using this tool and to make it publicly available and more user-friendly, such that other researchers, especially in the statistics community, can make use of it.

7.1 Depth Explorer for Analysis of Real Life Data

Application of the data-depth concept to real-life data have been suggested by statisticians for a while. Many of them are two-dimensional graphs that visualize statistical properties of high-dimensional data sets. The flexible architecture of **Depth Explorer** can be augmented to compute and display these graphs

- The multivariate nature of data depth yields simple two dimensional graphs for high dimensional data sets, that can be easily visualized and interpreted [36]. Examples include *scale curves* as a measure of scale/dispression, tracking how the depth contours

expand; *shrinkage plots* and *fan plots* as a measure of *kurtosis*, the overall spread relative to the spread in the tail; and *depth vs. depth (DD) plot* to compare variation between two sample distributions.

- The *bagplot* [50] [3] is a visual tool based on the half-space depth function that is indicative of the shape of a two dimensional data set. It includes a *bag*, the contour containing the $n/2$ observations with largest depth. Magnifying the bag by a factor 3 yields the *fence*, where observations outside the fence are flagged as outliers. The bagplot is an affine invariant and robust visualization tools for the location, spread, correlation, skewness, and tails of a two dimensional data set.

- The robust nature of the data depth concept makes it appropriate to serve as a robust classification and cluster analysis tool. Rousseeuw *et al.* [52] suggest using the volumes of the depth contours that contain α of the points of each cluster (for a constant α) to classify a data point to a given cluster or to partition a given set into a number of clusters. Jörnsten, Vardi and Zhang [27] introduced the *Relative Data Depth, ReD*, based on the L_1 depth as a validation tool for selecting the number of clusters and identifying outliers, in conjunction with an exact K-median algorithm. The concept was tested on several real-life data sets [26].

7.2 Technical Enhancements

On the technical side, work will concentrate on the following directions:

- **Depth Explorer** is currently limited to two dimensional visualizations and data. Since many of the computations extend trivially to three or more dimensions, support for two dimensional visualizations on higher dimensional data is under development. Later extending the software to support three dimensional visualizations will be a priority, however, a three dimensional interface will require careful consideration and much development. This will be a major focus in the ongoing development of Depth Explorer.

- Massive improvements to the editor interface including direct integration with online help and automatic highlighting of XML errors. A live preview of the data distribution will be added, eliminating the current back and forth nature of constructing a scene.

- The process to add a new data generator, modifier or visualizer is currently very simple. Steps need to be taken to further simplify this process and fully document it. The goal is to make it nearly trivial to expand the functionality of **Depth Explorer** within the render tree framework.

- Additional visualization types, For example: coloring points for clustering visualizations according to membership; shading regions of depth rather than outlinining or highlighting them; points displaying their numerical depth value as they are moused over.

- The current tool is single-threaded and cannot take advantage of the recent push for dual core processors on the desktop. Making the render process parallelizable will decrease render time, especially as the size of the render increases.

- Support of additional platforms other than Mac OS X including Microsoft Windows. Depth explorer has been developed with portability in mind, however, as development resources are limited, development and testing on additional platforms may be a secondary priority for the immediate future.

[3] The *Sunburst plot* [36] is a similar visual tool.

References

[1] P. K. Agarwal, M. Sharir, and E. Welzl. Algorithms for center and Tverberg points. In *Proc. 20th Annu. ACM Sympos. Comput. Geom.*, pages 61–67, 2004.

[2] G. Aloupis, C. Cortes, F. Gomez, M. Soss, and G. Toussaint. Lower bounds for computing statistical depth. *Computational Statistics & Data Analysis*, 40(2):223–229, 2002.

[3] N. Amenta, M. Bern, D. Eppstein, and S.-H. Teng. Regression depth and center points. *Discrete & Computational Geometry*, 23(3):305–323, 2000.

[4] V. Barnett. The ordering of multivariate data. *J. Roy. Statist. Soc. Ser. A*, 139(3):318–355, 1976.

[5] BLAS. Basic linear algebra subprograms. `http://www.netlib.org/blas/`.

[6] T. M. Chan. Output-sensitive results on convex hulls, extreme points, and related problems. *Discrete Comput. Geom.*, 16(4):369–387, 1996. Eleventh Annual Symposium on Computational Geometry (Vancouver, BC, 1995).

[7] T. M. Chan. An optimal randomized algorithm for maximum Tukey depth. In *Proceedings of 15th ACM-SIAM Symposium on Discrete Algorithms (SODA04)*. ACM Press, 2004.

[8] B. Chazelle. On the convex layers of a planar set. *IEEE Trans. Inform. Theory*, 31(4):509–517, 1985.

[9] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete Comput. Geom.*, 10(4):377–409, 1993.

[10] COCOA. Apple cocoa. `http://developer.apple.com/cocoa/`.

[11] R. Cole, M. Sharir, and C. K. Yap. On k-hulls and related problems. *SIAM Journal on Computing*, 15(1):61–77, 1987.

[12] D. L. Donoho and M. Gasko. Breakdown properties of location estimates based on halfspace depth and projected outlyingness. *Ann. Statist.*, 20(4):1803–1827, 1992.

[13] W. Eddy. Convex hull peeling. In H. Caussinus, editor, *COMPSTAT*, pages 42–47. Physica-Verlag, Wien, 1982.

[14] H. Edelsbrunner. *Algorithms in Computational Geometry*. Springer-Verlag, 1978.

[15] H. Edelsbrunner and R. Seidel. Voronoi diagrams and arrangements. *Discrete and Computational Geometry*, 1:25–44, 1986.

[16] S. Fortune. Voronoi diagrams and Delaunay triangulations. In *Handbook of discrete and computational geometry*, CRC Press Ser. Discrete Math. Appl., pages 377–388. CRC Press, Inc., Boca Raton, FL, USA, 1997.

[17] K. Fukuda and V. Rosta. Exact parallel algorithms for the location depth and the maximum feasible subsystem problems. In *Frontiers in global optimization*, volume 74 of *Nonconvex Optim. Appl.*, pages 123–133. Kluwer Acad. Publ., Boston, MA, 2004.

[18] K. R. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.

[19] Pdflib gmbh - main page, 2005.

[20] X. He and G. Wang. Convergence of depth contours for multivariate datasets. *Ann. Statist.*, 25(2):495–504, 1997.

[21] J. Hodges. A bivariate sign test. *The Annals of Mathematical Statistics*, 26:523–527, 1955.

[22] J. Hugg. Depth explorer website. `www.cs.tufts.edu/r/geometry/depthexplorer/`.

[23] F. Hurtado, G. Liotta, and H. Meijer. Optimal and suboptimal robust algorithms for proximity graphs. *Comput. Geom.*, 25(1-2):35–49, 2003. Special issue on the European Workshop on Computational Geometry—CG01 (Berlin).

[24] J. W. Jaromczyk and G. T. Toussaint. Relative neighborhood graphs and their relatives. *Proc. IEEE*, 80(9):1502–1517, sep 1992.

[25] T. Johnson, I. Kwok, and R. Ng. Fast computation of 2-dimensional depth contours. In *Proc. 4th International Conference on Knowledge Discovery and Data Mining*, pages 224–228, 1998.

[26] R. Jörnsten. Clustering and classification based on the L_1 data depth. *J. Multivariate Anal.*, 90(1):67–89, 2004.

[27] R. Jörnsten, Y. Vardi, and C.-H. Zhang. A robust clustering method and visualization tool based on data depth. In *Statistical data analysis based on the L_1-norm and related methods (Neuchâtel, 2002)*, Stat. Ind. Technol., pages 353–366. Birkhäuser, Basel, 2002.

[28] D. G. Kirkpatrick and J. D. Radke. A framework for computational morphology. In G. Toussaint, editor, *Computational geometry*, pages 217–248. North-Holland, 1985.

[29] S. Krishnan, N. H. Mustafa, and S. Venkatasubramanian. Hardware-assisted computation of depth contours. In *13th ACM-SIAM Symposium on Discrete Algorithms*, 2002.

[30] S. Langerman and W. Steiger. Optimization in arrangements. In *Proceedings of the 20th International Symposium on Theoretical Aspects of Computer Science (STACS 2003)*, 2003.

[31] LAPACK. Linear algebra package. `http://www.netlib.org/lapack/`.

[32] LEDA. Library of efficient data structures and algorithms. `www.ag2.mpi-sb.mpg.de/LEDA`.

[33] A. Lingas. A linear-time construction of the relative neighborhood graph from the Delaunay triangulation. *Comput. Geom.*, 4(4):199–208, 1994.

[34] R. Liu. On a notion of data depth based on random simplices. *The Annals of Statistics*, 18:405–414, 1990.

[35] R. Liu. Data depth: center-outward ordering of multivariate data and nonparametric multivariate statistics. In M. Akritas and D. Politis, editors, *Recent Advances and Trends in Nonparametric Statistics*, pages 155–168. Elsevier Science, 2003.

[36] R. Liu, J. Parelius, and K. Singh. Multivariate analysis by data depth: descriptive statistics, graphics and inference. *The Annals of Statistics*, 27:783–858, 1999.

[37] J. Matoušek. Computing the center of planar point sets. *DIMACS Series in Disc. Math. and Theoretical Comp. Sci.*, 6:221–230, 1991.

[38] K. Miller, S. Ramaswami, P. Rousseeuw, T. Sellarés, D. Souvaine, I. Streinu, and A. Struyf. Efficient computation of location depth contours by methods of combinatorial geometry. *Statistics and Computing*, 13(2):153–162, 2003.

[39] T. Ottmann, S. Schuierer, and S. Soundaralakshmi. Enumerating extreme points in higher dimensions. In *Symposium on Theoretical Aspects of Computer Science*, pages 562–570, 1995.

[40] F. Preparata and S. Hong. Convex hulls of finite sets of points in two and three dimensions. *Commun. ACM*, 20(2):87–93, 1977.

[41] F. P. Preparata and M. I. Shamos. *Computational geometry*. Texts and Monographs in Computer Science. Springer-Verlag, New York, 1985. An introduction.

[42] E. Rafalin, K. Seyboth, and D. Souvaine. Path length in proximity graphs as a data depth measure. *Tufts CS Technical Report 2005-5*, Tufts University, Nov. 2005. Abstract appeared in *Proceedings of the 15th Annual Fall Workshop on Computational Geometry*, UPenn, 2005, pages 11-12.

[43] E. Rafalin, K. Seyboth, and D. Souvaine. Proximity graph depth, depth contours, and a new multimodal median, 2005. submitted for publication *22nd Annual*

ACM Symposium on Computational Geometry.

[44] P. Rousseeuw and M. Hubert. Depth in an arrangement of hyperplanes. *Discrete & Computational Geometry*, 22:167–176, 1999.

[45] P. Rousseeuw and I. Ruts. Bivariate location depth. *Applied Statistics-Journal of the Royal Statistical Society Series C*, 45(4):516–526, 1996.

[46] P. J. Rousseeuw. Introduction to positive-breakdown methods. In *Robust inference*, volume 15 of *Handbook of Statist.*, pages 101–121. North-Holland, Amsterdam, 1997.

[47] P. J. Rousseeuw. Introduction to positive-breakdown methods. In J. E. Goodman and J. O'Rourke, editors, *Handbook of discrete and computational geometry*, Discrete Mathematics and its Applications (Boca Raton), pages xviii+1539. Chapman & Hall/CRC, Boca Raton, FL, second edition, 2004.

[48] P. J. Rousseeuw and M. Hubert. Regression depth. *J. Amer. Statist. Assoc.*, 94:388–433 (with discussion), 1999.

[49] P. J. Rousseeuw and I. Ruts. Constructing the bivariate tukey median. *Statistica Sinica*, 8:827–839, 1998.

[50] P. J. Rousseeuw, I. Ruts, and J. W. Tukey. The bagplot: A bivariate boxplot. *The American Statistician*, 53:382–387, 1999.

[51] P. J. Rousseeuw and A. Struyf. Computing location depth and regression depth in higher dimensions. *Statistics and Computing*, 8:193–203, 1998.

[52] I. Ruts and P. J. Rousseeuw. Computing depth contours of bivariate point clouds. *Comp. Stat. and Data Analysis*, 23:153–168, 1996.

[53] A. Struyf and P. Rousseeuw. High-dimensional computation of the deepest location. Manuscript, Dept. of Mathematics and Computer Science, University of Antwerp, Belgium, 1999.

[54] G. Toussaint. The relative neighborhood graph of a finite planar set. *Pattern Recognition*, 12:261–268, 1980.

[55] J. Tukey. Mathematics and the picturing of data. In *Proceedings of the International Congress of Mathematics*, pages 523–531, 1974.

[56] J. W. Tukey. Mathematics and the picturing of data. In *Proc. of the Int. Cong. of Math. (Vancouver, B. C., 1974), Vol. 2*, pages 523–531. Canad. Math. Congress, Montreal, Que., 1975.

[57] Y. Vardi and C.-H. Zhang. The multivariate L_1-median and associated data depth. *Proc. Nat. Acad. Sci. USA.*, 97:1423–1426, 2000.

[58] K. Verbarg. Approximate center points in dense point sets. *Inform. Process. Lett.*, 61(5):271–278, 1997.

[59] E. W. Weisstein. Mode. From MathWorld, `http://mathworld.wolfram.com/Mode.html`.

[60] Y. Zuo and R. Serfling. General notions of statistical depth function. *Ann. Statist.*, 28(2):461–482, 2000.

[61] Y. Zuo and R. Serfling. Structural properties and convergence results for contours of sample statistical depth functions. *Ann. Statist.*, 28(2):483–499, 2000.