

# Depth Explorer — A Software Tool for Analysis of Depth Measures

J. Hugg\*

E. Rafalin\*

D. L. Souvaine\*

## 1 Introduction

Data depth is an analysis method measuring how central a point is relative to a cloud of data points. Given a cloud of data points in  $R^2$ , the most central point, the *median*, has the highest depth value. Points on the periphery have lowest depth values. Moving inward from the periphery to the median, the depth increases. Examples of data depth functions include convex-hull peeling depth<sup>1</sup> and halfspace depth<sup>2</sup>.

We present *Depth Explorer*, a software tool for the examination of depth measures, written in C++ and Objective-C. It uses a tree-based data structure to render 2-D statistical distributions and depth-based visualizations specified in XML files. We demonstrate the potential of the tool, presenting an experimental analysis of the  $L_1$  depth measure that exposes weaknesses of  $L_1$  depth and evaluates potential improvements.

### 1.1 Depth Contours

*Depth contours*, nested contours that enclose regions of increasing depth, provide powerful tools to visualize and compare data sets. The contour of the **sample  $\alpha$ th central region** is the Convex Hull (CH) containing the most central fraction of  $\alpha$  sample points [5]. The contour is constructed by sorting all points of the original set according to their depth. A contour that encloses, for example, 75% of the points is created by taking the CH around data points  $X_{[1]}, \dots, X_{[\lceil .75n \rceil]}$ .

Depth contours helps visualize the *shape* of the data and determine outliers, foreign influence on the data or errors, any of which can have an undesirable influence on the analysis of the data. The shape of a contour can help determine which data points may be outliers. One outlier detection method [6] grows the 50% contour by a factor of three and classifies points outside the expanded region as outliers. This assumes that the depth measure used generates a 50% contour that is indicative of the overall shape of the data. Depth contours as defined require monotonic depth measures: as a point  $x$  moves away from

the ‘deepest point’ along any fixed ray through the center, the depth at  $x$  should decrease monotonically [10].

### 1.2 Data Depth in High Dimensions

Currently, depth based statistics are proving very successful at analyzing bivariate data sets. While concepts of depth and depth contours extend well to higher dimensions, the computational complexity of measures like half-space depth is exponential in dimension. To use depth with high dimensional data, other depth measures or approximation schemes must be used. Rousseeauw developed an approximation for the half-space depth measure that scales linearly in the number of dimensions [7]. The  $L_1$  depth measure developed by Vardi and Zhang [9] is also scalable.

**Definition:** The  $L_1$  depth ( $L_1D$ ) [9] of point  $x$  with respect to set  $S = \{X_1, \dots, X_n\}$  in  $\mathbb{R}^d$  is one minus the average of the unit vectors from  $x$  to all observations in  $S$ :  $L_1D(S, x) = 1 - \|\bar{e}(x)\|$ , where  $e_i(x) = \frac{x - X_i}{\|x - X_i\|}$ ,  $\bar{e}(x) = \frac{\sum_{i=1}^n \eta_i e_i(x)}{\sum_j \eta_j}$ .  $\eta_i$  is a weight assigned to observation  $X_i$  (1 if all observations are unique), and  $\|x - X_i\|$  is the Euclidean distance between  $x$  and  $X_i$ .

The  $L_1$  median of a set  $S$  in  $\mathbb{R}^n$  is the point that minimizes the sum of the Euclidian distances to all points in  $S$ .

## 2 Depth Explorer

There is no definitive quantified way of comparing and evaluating depth measures. Most statistical tools, including depth measures, are evaluated by comparing results generated from data sets. Statistical software packages often focus on applying existing tools to fixed data sets, but ideally tools would be compared over the space of all data sets. As an approximation of this unachievable goal, we propose the Depth Explorer (DE) statistical sandbox. Rather than using static data and changing tools or their parameters, our software keeps the tool settings constant and perturbs the data. DE facilitates the automatic generation of data sets and the transformation or recomposition of existing data sets, and quickly visualizes the behavior of the statistical method on the data. The core of DE is written in C++, while the GUI is written in Objective-C. This separation of logic and presentation will facilitate porting to other systems.

DE does not render instantaneously even a relatively easy depth measure like  $L_1$ . Nonetheless, almost all scenes are rendered within seconds, not minutes, allowing interactive feedback. Furthermore, if no depth-based visualizations are embedded within the render tree, even a complicated scene with thousands of data points renders almost instantly; if depth calculations are required, computation time is slower. To render an  $L_1$  50% depth

\*Department of Computer Science, Tufts University, Medford, MA 02155. {jhugg,erafalin,dls}@cs.tufts.edu. Partially supported by NSF grant CCF-0431027. Full version in [4].

<sup>1</sup>The **convex-hull peeling depth** [2, 1] of a point  $X_k$  with respect to a data set  $S = \{X_1, \dots, X_n\}$  in  $\mathbb{R}^d$  is the level of the convex layer to which  $X_k$  belongs. The points on the outer convex hull of  $S$  are designated level one and the points on the  $k$ th level are the points on the convex hull of the set  $S$  after the points on all previous levels were removed.

<sup>2</sup>The **half-space depth** [3, 8] of a point or position  $x$  relative to a set of points  $S = \{X_1, \dots, X_n\}$  is the minimum number of points of  $S$  lying in any closed half-space determined by a line through  $x$ .

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, we believe that many implementation optimizations are not fully realized until the data set gets larger.

## 2.1 Representing Operations as a Tree

DE currently uses an XML markup to represent the render tree. XML is well suited due to its hierarchical nature. The root of the XML tree and the render tree is the canvas tag, which can describe a screen view, a page view or both. DE converts the XML data into an *internal* tree representation. The leaf nodes must be data sources, either CSV files with a data set, or a random cloud generator with a set of points. DE creates Gaussian clouds as well as uniformly distributed clouds.

Non-leaf nodes do not contain new data; rather, they modify data or add visualizations to data. DE supports affine transformation nodes that scale, rotate or translate nodes below them. With randomly generated clouds followed by affine transformations, a broad spectrum of data sets can be generated.

To visualize a depth measure, contours can be drawn along lines of equal depth. Points can be colored according to their clustering. As DE uses a modular tree-based renderer, it is trivial to cluster data and generate depth contours concurrently, even at different points in the hierarchy.

The tree is rendered starting from the leaf nodes up. Each node is a C++ module that, given the output of its children nodes, modifies or appends the data, then passes the data to its parent node. Ultimately, using the *canvas* node’s parameters, DE renders the result at the root of the tree data into a pdf document.

## 3 Analysis of the $L_1$ Depth

The  $L_1$  depth measure is attractive, as it is fast, simple to implement and easily parallelizable. The quality of results, however, is not stellar. Using DE, we compared the performance of  $L_1$  depth and of CH Peeling Depth on many sample data sets quickly. As DE currently supports only  $L_1$  Depth and CH Peeling Depth, peeling depth is used as a reference for performance. We are working on expanding DE to include additional depth measures.

We noted that, in  $L_1$  depth, a point whose distance to the median point is small is more likely to have greater depth than the same point in other depth measures. As a result,  $L_1$  depth contours tend toward circularity: a long and thin cloud of points has a much less thin 50% contour; similarly, a rectangular cloud has round contours. Since contours should approximate the shape of the data, the  $L_1$  depth measure is less useful than other depth measures.

Statisticians cite *affine invariance* as a key quality of a “good” depth measure [10]: the depth of a point should remain fixed under affine transformation of the point set.  $L_1$  depth is not affine invariant, but this presumed flaw actually helps: affine transformations on non-spherical data can produce a cloud that approaches a hypersphere on

which  $L_1$  performs well. The resulting depths are mapped back to the original pointset.

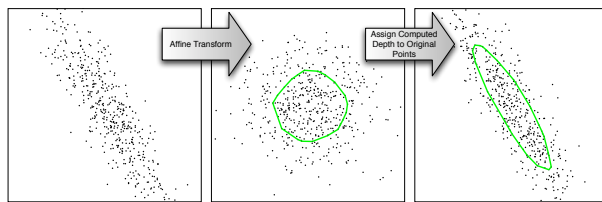


Figure 1: Affine Scaling Improves the  $L_1$  Depth Results

The fundamental difficulty in scaling  $L_1$  depth is determining the proper affine transformation to produce an approximate hyperspherical cloud. We use Principal Component Analysis (PCA). For  $d$  dimensional data, PCA generates a set of  $d$  perpendicular vectors describing the general directions of the data cloud. Scaling along these vectors produces an approximation of a hyperspherical cloud. Like the original  $L_1$  depth measure, PCA-based scaling  $L_1$  depth is affine-rotation invariant. However, PCA-based scaling  $L_1$  depth removes much of the aforementioned bias towards points closer to the median when assigning depth values (see Figure 1), allowing it to generate a contour that approximates well the shape of a long, thin cloud. Note, however, that it still produces rounded contours for square data clouds. Also this method is still limited to clouds that are roughly convex, but many depth measures share this limitation.

We used DE to confirm our hypothesis visually. See Figure 1 for an example of successful PCA-based scaling  $L_1$  depth. The final generated contour is a good approximation of the shape of the cloud. This new method of scaling does not, however, address the tendency for  $L_1$  depth contours to be rounded in square data clouds.

## References

- [1] V. Barnett. The ordering of multivariate data. *J. Roy. Statist. Soc. Ser. A*, 139(3):318–355, 1976.
- [2] W. Eddy. Convex hull peeling. In H. Caussinus, editor, *COMPSTAT*, pages 42–47. Physica-Verlag, Wien, 1982.
- [3] J. Hodges. A bivariate sign test. *The Annals of Mathematical Statistics*, 26:523–527, 1955.
- [4] J. hugg, E. Rafalin, and D. L. Souvaine. Depth explorer - a software tool for analysis of depth measures. TUFTS-CS technical report tr-2005-6, Tufts University, 2005.
- [5] 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.
- [6] P. J. Rousseeuw, I. Ruts, and J. W. Tukey. The bagplot: A bivariate boxplot. *The American Statistician*, 53(4), 1999.
- [7] P. J. Rousseeuw and A. Struyf. Computing location depth and regression depth in higher dimensions. *Statistics and Computing*, 8:193–203, 1998.
- [8] 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.
- [9] Y. Vardi and C.-H. Zhang. The multivariate  $L_1$ -median and associated data depth. *Proc. Nat. Acad. Sci. USA.*, 97:1423–1426, 2000.
- [10] Y. Zuo and R. Serfling. General notions of statistical depth function. *Ann. Statist.*, 28(2):461–482, 2000.