Adjusting the Generalized Barycentric Coordinates for More Comprehensive Layout

Shenghui Cheng, Bing Wang, Zhiyuan Zhang, Klaus Mueller

Visual Analytics and Imaging Lab, Computer Science Department, Stony Brook University, NY, USA and SUNY Korea, Songdo, Korea

ABSTRACT

The Generalized Barycentric Coordinates (GBC) plot is often used to visualize high-dimensional data in 2D. However, its mapping is not overly accurate in some cases. We propose an algorithm that adjusts the GBC layout as well as the data points inside the GBC. It consists of ordering the variables by a correlation-based method and an iterative error-reduction scheme. We provide two examples to evaluate our method, quantitatively and qualitatively.

1 INTRODUCTION

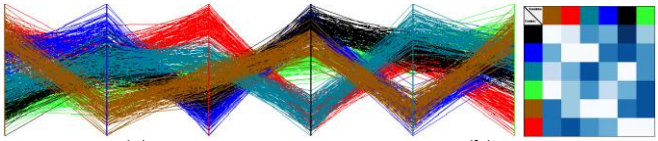
The visualization of high dimensional data has become a frequently studied topic. The objective typically is to find patterns in the data, such as clusters as well as outliers to these clusters. One method is the Generalized Barycentric Coordinates (GBC) plot [1]. Its attributes are the vertices of a regular sided polygon and the samples form patterns in its interior.

Our method is inspired by systems that arrange the nodes representing the attributes along a convex shape and lay out the data points in the interior of this shape. The most well-known framework of this kind is RadViz [2] which uniformly spaces the attributes as *dimensional anchors* along the circumference of a circle. The location of the data points is then determined by a weighting formula where data point attributes with higher values receive a higher weight and so increase the attraction of the point to the corresponding anchor points. However, this can lead to location ambiguities which can be reduced by re-ordering the anchor points either manually or algorithmically. Related to RadViz is Gravi++ [3] which uses a different weighting formula but also spaces the attributes at uniform distances onto an encompassing circle. GBC can be viewed as a generalization of all of these systems.

While the GBC is more flexible than other methods, in order to show more accurate relations between the data points and variables, it needs to adjust the data points. Our algorithm also lays out the variables on the general convex polygon, but the distance of two variables is determined by correlation similarity. Lastly, since the data points are also not comprehensive in terms of relative positions to the variables, we give an iterative error-reduction scheme.

2 THE GENERALIZED BARYCENTRIC COORDINATES (GBC)

To conduct our experiments, we generated a test dataset comprised of a set of 6 6-D Gaussian distributions. Fig. 1a visualizes this dataset using parallel coordinates, assigning each Gaussian a unique color. In addition, we also colored the axes (representing the 6 dimensions or variables) such that each axis color matches that of the cluster with the highest value for that dimension.



(a) (b)

Fig. 1: Visualization of the value matrix. (a) Parallel coordinate plot. (b) Value matrix.

Email:{shecheng, wang12, zyzhang, mueller}@cs.sunysb.edu

Fig. 1b shows what we call the *value matrix*. It visualizes the normalized attribute values of the center vectors using a single-hue sequential color scheme as a function of cluster ID (x-axis) and dimension (y-axis). Darker blue tones correspond to higher values.

2.1 The GBC Plot

The GBC plot is derived from GBC interpolation [1] which extends the method of barycentric interpolation from triangles to multi-vertex convex polygons. The task is to interpolate the value of an interior point p from the values stored at the polygon vertices v_i. Referring to Fig. 2, the interpolation weight w_i of vertex v_i for p is:

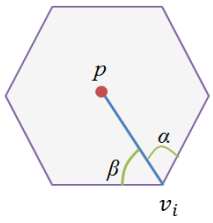


Fig.2 The GBC Plot

$$w_i = \frac{\cot(\alpha) + \cot(\beta)}{\|p - v_i\|^2}$$

Then, given N vertices, the interpolated value pv at p is:

$$pv = \sum_{i=1}^{N} v_i a_i \quad \text{where } a_i = w_i \Big/ \sum_{k=1}^{N} w_k \quad \text{and} \quad \sum_{i=1}^{N} a_i = 1$$

The GBC plot seeks to compute the position of a point p in a convex polygon in which each vertex is assigned to one of the attributes. This replaces pv by the 2D vector p, and the v_i by the 2D coordinates of the attribute vertices. We then set the weights to be the values of the N-dimensional vector, normalize them to compute the a_i, and finally use the 2D coordinates of the attribute vertices to interpolate the 2D coordinate of p. Fig. 4a shows a resulting layout, coloring the clusters and the polygon vertices in the same shade as in Fig. 1.

3 ALGORITHM

3.1 Correlation Ordered Variables

Some existing methods, such as star coordinates, have become better by modifying the arrangements of the variable nodes or coordinate systems. We can try to see if this also helps for the GBC plot. In this spirit, we might hypothesize that when the correlation of two variables is high, they are more similar and should be put closer, else they should be spread more apart. Since we are bound to preserve a linear ordering of the vertices on the polygonal hull, we need to generate some ordering that can maximize this goal. We can accomplish this by arranging the vertices through an approximate Traveling Salesman Problem (TSP) solver that operates on the matrix of pairwise correlations among the variables. TSP has been successfully employed to determine a good axis ordering for parallel coordinates [4].

We place all attribute vertices on a circle, ordered by the correlation-based TSP solver and spaced apart by the reverse pairwise correlations. In this way, the relative positions of variables to variables become more accurate and comprehensive. Fig. 4 shows an example.

3.2 Iterative Error Reduction

Next, we explore if an iterative error-reduction scheme can improve matters. It seeks to reconstruct an error polygon for each data point

and iteratively reduce size of this polygon. Our method is illustrated in Fig. 3 along with the algorithm.

The first assumption this algorithm makes is the existence of a set of distance contours that encode the importance of a variable to a given data point. Suppose we have the variables V_1, V_2, V_3, V_4, V_5 and a data point $X = (x_1, x_2, x_3, x_4, x_5)$. Fig. 3 examines the distance contours for variable V_4. Assuming that X has been normalized to a unit vector, the maximum importance a variable can have is 1.0. This would mean in the case examined that x_4=1.0 and so X would coincide with V_4's vertex in the plot. In contrast, if x_4=0.0 which is the minimum importance, then with the current vertex ordering X would need to fall on the edges V_5V_1 , V_1V_2 or V_2V_3. Any other value would lead to a placement of X onto some contour in between. Fig. 3 shows the contour $\overline{P_5P_1P_2P_3}$ for x_4=0.6. The contour is constructed by connecting V_4 with all vertices V_i and marking the points P_i where $(\overline{V_4P_i})/(\overline{V_4V_i}) = 1 - 0.6$. Connecting these points yields the desired contour.

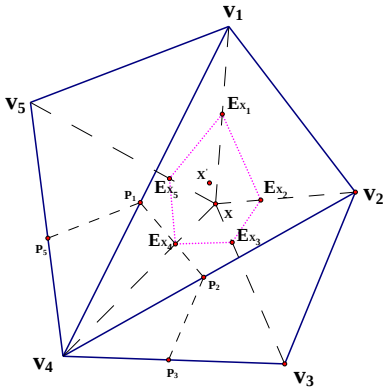


Fig.3: Illustration and algorithm.

Next we find V_4's vertex on the error polygon (marked as E_{X_4}) by intersecting the contour with the line that connects V_4 with X. Performing this procedure for all variables yields all vertices of the error polygon (marked as polygon $E_{X_1}E_{X_2}E_{X_3}E_{X_4}E_{X_5}$). The iterative step concludes by moving X into the center of the error polygon, and then a new iteration begins.

In practice we iterate about 50 times which completes in a couple of seconds and so does not cause a significant performance drop. The result of this algorithm for our test data set is shown in Fig. 4b. Overall, the GBC comprehensive layout algorithm is as follows.

Algorithm 1: GBC Comprehensive Layout Algorithm

1. Reorder the variables by TSP and space them.
2. Initial: set the error threshold and maximum iterations.

If error < threshold || iterations > maximum, return.

For each data point X_i

For each variable V_j

Compute distance contour

Compute error polygon vertex $E_{X_{ij}}$

Construct error polygon EP_i formed by $E_{X_{ij}}$ (j=1, 2,.., n)

Move X_i to the center of EP_i

Compute the overall error and iterations.

4 Evaluation

A simple approach to evaluate the GBC plot is the size of the error polygon. The better layout should have a smaller error polygon. Suppose a dataset with m data points and n dimension. The distance of the ith data point to the jth dimension point is l_{ij} with the error ε_{ij} in this dimension. Then the error ratio θ of the CBC plot in terms of this data is

$$\theta = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\varepsilon_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n}l_{ij}}$$

The error ratio of the test data reduces from 35.7% to 24.3%.

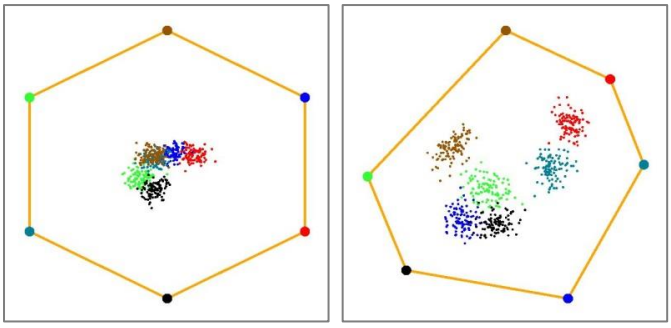


Fig. 4 (a) Original GBC (b) Modified GBC

5 Case Study

We obtained a collection of stock data online: 1,662 stocks with 21 metrics (Fig. 5). We cluster the stocks into 4 clusters with different colors. Fig. 5a is the original GBC plot and Fig. 5b is the modified one. We can see that in Fig. 5a, most points are congested in the corner and there is much overlap. The features of the clusters cannot be well appreciated. By contrast, Fig. 5b shows the stock better. The four clusters have broken away. In addition, the error ratio reduces from 11.2% to 2.4%.

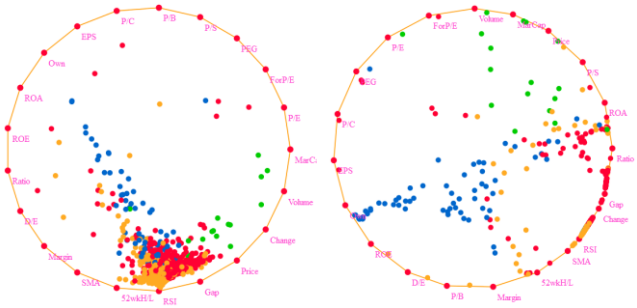


Fig.5. GBC with stock data. (a) original GBC (b) modified GBC

6 Conclusion

In this poster, we propose an algorithm to adjust the GBC layout as well as the data points located inside the GBC. First, we reorder the variables and space them according to the pairwise correlations. It can give a more accurate layout of variables. Second, we move the data points iteratively to reduce the error area size. We then define and show the error ratio.

7 Acknowledgment

This research was partially supported by NSF grant 1117132 and the Korean Ministry of Science, ICT and Future Planning Korea under the IT Consilience Creative Program supervised by NIPA.

References

[1] M. Meyer, A. Barr, H. Lee, M. Desbrun, "Generalized Barycentric Coordinates on Irregular Polygons," *J. Graphics Tools*, 7(1):13-22, 2002.

[2] P. Hoffman, G. Grinstein, K. Marx, I. Grosse, E. Stanley, "DNA Visual and Analytic Data Mining", *Proc. IEEE Vis*, pp. 437-441, 1997.

[3] K. Hinum, S. Miksch, W. Aigner, S. Ohmann, C. Popow, M. Pohl, M. Rester, "Gravi++: Interactive Information Visualization to Explore Highly Structured Temporal Data," *J. Universal Computer Science* 11(11):1792-1805, 2005.

[4] Z. Zhang, K. McDonnell, K. Mueller, "A Network-Based Interface for the Exploration of High-Dimensional Data Spaces," *IEEE Pacific Vis*, Songdo, Korea, pp. 17-24, March, 2012.