Balanced Layouts Using the Composite Data-Variable Matrix

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

Numerous methods have been described that allow the visualization of the data-variable matrix. But all suffer from a common problem-visualizing the data and variable points separately which is hard for people to catch the relations in data and variables together. We describe a method that allows data and variables balanced layouts. We achieve it by combining two distance matrices typically used in isolation – the distance matrix encoding the similarities of the variables and the distance matrix encoding the similarity of the data points. The remaining two submatrices are obtained by creating a fused distance matrix – one that measures the distance of data points with respect to the variables or vice versa. We then use MDS to simultaneously optimize the placement of data points and variable points, producing a display that allows users to appreciate all three types of relationships in a single display: (1) the patterns of the collection of data items, (2) the patterns of the collection of variables, and (3) the relationships of data items with the variables and vice versa.

1 INTRODUCTION

The data matrix is a common representation high-dimensional datasets. Let *N* be the number of samples (or data points) drawn from a given population and let *D* be the number of attributes (or variables) measured per sample – we then obtain an *N*×*D* data matrix. In this data matrix, the samples and attributes can change roles. For example, for a data matrix storing the results of a DNA microarray experiment for multiple specimens, one research objective might consider the genes expressed in the microarray to be the samples and the specimens to be the attributes, or vice versa. Switching from one objective to the other formally requires a transposition of the data matrix.

Numerous methods have been described that allow the visualization of the data matrix. Embedding the high-dimensional space onto a 2D canvas via a suitable optimization strategy is a common strategy. In a low-dimensional space embedding, such as multi-dimensional scaling (MDS) [1], linear discriminant analysis (LDA), and others the attributes are even completely suppressed and only clusters of samples can be visually appreciated.

While changing the roles of samples and attributes is easy – it requires a simple transpose of the data matrix – the unequal treatment of attributes and samples represents a significant problem. It makes it difficult to observe patterns formed by attributes and samples simultaneously, and it also makes it difficult to see the samples in the proper context of the attributes. The method we propose provides such a comprehensive display. It uses MDS to simultaneously optimize the placement of samples and attributes.

2 THE COMPOSITE DISTANCE MATRIX

Let $DM = [x_{ij}]_{m\times n}$ be the *data matrix* with m rows and n columns, where the rows denote the data points, the columns denote the variables and x_{ij} is the data value in the ith row and jth column. Without loss of generality, we assume DM is normalized to [0, 1]. Now let D be the *data space* with m data points:

$$D_i = [x_{i1}, x_{i2}, \dots, x_{in}] \qquad (i = 1,2, \dots, m)$$

Let V be the *variable space* with n variables:

$$V_j = [x_{1j}, x_{2j}, \dots, x_{mj}]^T \qquad (j = 1,2, \dots, n)$$

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

where T is the transpose function. Thus, we can look at DM in two ways. We can map it into variable space V in which D represents the points, or we can map it into data space D where V represents the points. An illustration of the inverse relationship is provided in Fig. 1.

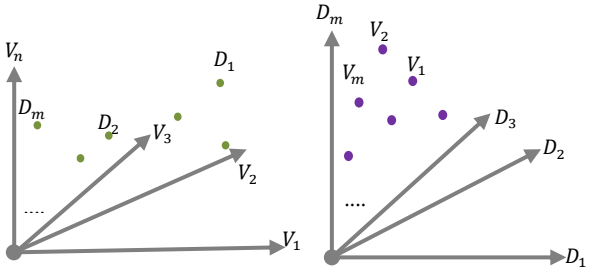

Fig. 1: The inverse relationships of *D* and *V* in the data matrix.

2.1 Extending the Data Matrix

As mentioned, current visualization methods tend to look at the two spaces – data space and variable space – in an imbalanced fashion. The usual resort is to either visualize the data matrix or its transpose with the algorithm at hand which then lowers the fidelity of one space at the cost of the other. But it can often be beneficial to see both spaces at the same time and do so in a balanced way where all relationships – data to data, data to variables, and variables to variables – are conveyed at equal fidelity.

Visualization of relationships in a data matrix can be made explicit by transforming it into a *distance matrix*. The notion of distance (also often called dissimilarity) can take many forms – Euclidian, cosine similarity, correlation, etc. But in all cases, the matrix stores the pairwise distances of two data matrix vectors, either *V* or *D*, but not both. So only one type of relation gets expressed, *V* or *D*.

Our solution is to create a distance matrix in which both types of relations are equally expressed. We call this matrix the *composite distance matrix* and the space the *composite space* (see Fig. 2). In this composite space, both data and variables can be located at the same time.

2.2 Creating the Composite Distance Matrix

We can derive the composite distance matrix C_{DV} as follows:

$$C_{DV} = \begin{bmatrix} DD & DV \\ VD & VV \end{bmatrix}$$

Here, DD stores the pairwise dissimilarities of the data points, VD and DV store the pairwise dissimilarities of the variables with the data points, and VV stores the pairwise dissimilarities of variables.

As mentioned, there are various measures suitable to express distance or dissimilarity. However, these measures have sometimes opposite meaning. Let F be the function of *Dissimilarity Metrics* where *F*=Euclidian Distance||1-Cosine Similarity||1-Correlation||…

2.1 The Data to Data Distance Matrix (*DD*)

The data points are vectors of equal length. The dissimilarity can be obtained using any of the functions in *F*. Then the *DD* matrix is an *m*×*m* matrix with elements:

$$DD_{ij} = F(D_i, D_j).$$

To demonstrate our method with a controlled experiment, we generated a test dataset comprised of a set of 6 6-D Gaussian distributions.

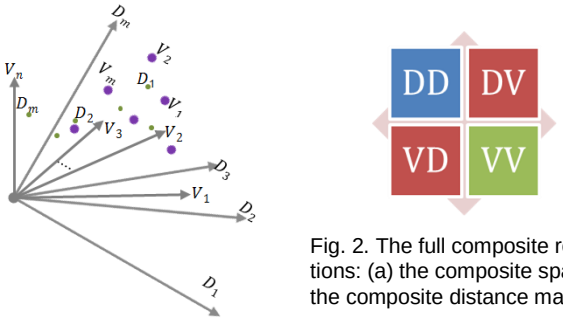

Fig. 2. The full composite representations: (a) the composite space, (b) the composite distance matrix.

The MDS layout of its *DD* distance matrix is shown in Fig. 3a.

2.2.2 The Variable to Variable Distance Matrix (*VV*)

Like *DD*, the dissimilarity between the variables can also be easily obtained by applying *F*. The matrix *VV* is of size $n \times n$ and its elements are given as:

$$VV_{ij} = F(V_i, V_j)$$

Fig. 3d shows an MDS layout of this *VV* distance matrix.

2.2.3 The Data to Variables Distance Matrix (*DV*, *VD*)

The *DD* and *VV* matrices are the well-known distance matrices that result when *DM* is either directly distance-transformed, or its transpose. The *DV* and *VD* matrices, on the other hand, are new types of matrices in this regard and we need to define them first. This effort is complicated by the fact that the lengths of the data and variable vectors are likely not equal, which makes it impossible to calculate their dissimilarity directly using *F*. In the following let us first consider *DV* – similar arguments also hold for *VD*.

We start by realizing that in the data space (Fig. 1a), spanned by the variable axes, these variable axes can also be considered vectors. Then each V_j can be rewritten as:

$$\acute{V}_j = [\overbrace{0,0,\dots,1}^{\text{j numbers}},0,\dots,0]^T$$

where the single 1– the value is at the position of the dimension the axis represents. It is a value of 1 since the axis vectors are unit vectors. With these vectors in place we can write the elements of the $n \times m$ matrix *DV* as follows:

$$DV_{ij} = F(D_i, \acute{V}_j)$$

In a similar fashion, just with reversed roles of *D* and *V* (Fig. 1b), we can also write the elements of the $m \times n$ matrix *VD* as follows:

$$VD_{ij} = F(V_i, \acute{D}_j)$$

To visualize these two matrices in isolation, we cannot use MDS since this requires a square matrix. We use a Generalized Barycentric Coordinate [2] layout to show these matrices in Fig. 3b and Fig. 3c.

2.2.4 The Full Distance Matrix – Putting It All Together

Once the *DD*, *DB*, *VD*, *and VV* sub-matrices have been constructed, they are assembled into the $(m+n)\times(m+n)$ composite distance matrix C_{DV}. We then use the MDS algorithm for its layout. But before running the algorithm it is important to notice that these four sub-matrices were not created equally. They have been calculated from vectors with different lengths – *n* or *m* – and they may also have used different dissimilarity metrics *F*. We have observed that this inequality can lead to cases in which data and variables may not mix well. That is, the points due to the data vectors and those due to the variables may clump together into separate and disjoint clusters. To overcome this problem, we use the following weight adjustment scheme:

$$W_{DD}:W_{DV}:W_{VD}:W_{VV} = \frac{M_{max}}{M_{DD}}:\frac{M_{max}}{M_{DV}}:\frac{M_{max}}{M_{VD}}:\frac{M_{max}}{M_{VV}}$$

where M_{IJ} is the mean of Matrix IJ, $I,J \in \{D,V\}$. Our experiments indicate that this scheme works quite well in practice, and Fig. 3e shows the end result for this example, combining *DD*, *VV*, *DV*, and *VD* into the final composite distance matrix C_{DV}. We note that this matrix is not the sum of the four individual matrices. Rather, it is found by the MDS algorithm that now operates on the entire matrix at once. In Fig. 3e the thick points are the variables. We have verified that their placement in all cases quite accurately corresponds to the dominant dimensions in the nearby clusters.

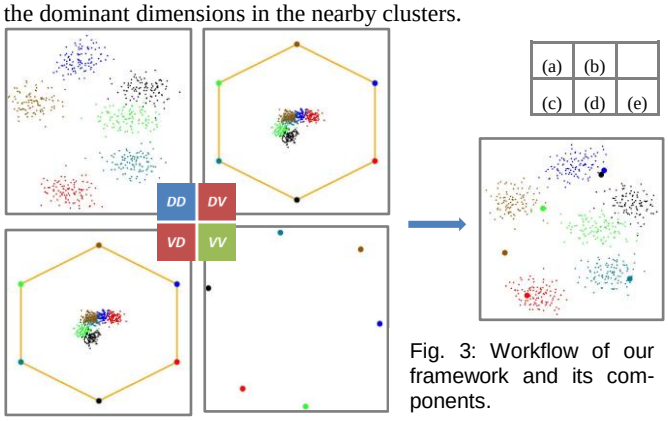

Fig. 3: Workflow of our framework and its components.

3 Case Study

We used our method to visualize the red wine quality dataset with 1,599 instances and 12 attributes (see Fig.4). The small blue dots are the wines and the larger red dots are the attributes. We can learn a few things from this layout. We see that the number of high quality wines is relatively sparse and that high quality wines typically have high alcohol, sugar, and fixed acidity. We can also see that most wines seem to have high PH, sulphate, and citric acid, but that certain tradeoff exist. For example, high PH has low acid or vice versa.

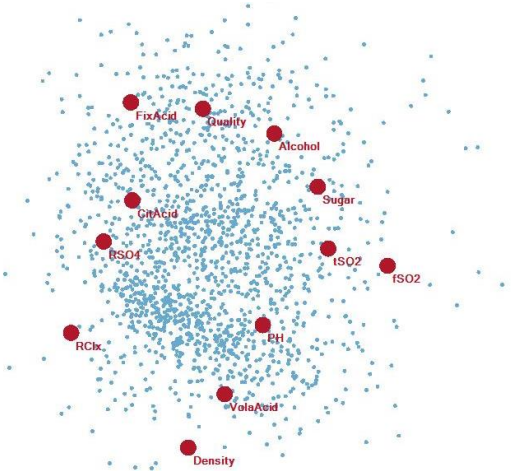

Fig. 4. The wine quality data balanced layout.

4 Conclusions

Our method allows a balanced layout of data and variables points. We believe one of the advantages is that it provides data layouts that are essentially self-labeling – we can cluster the data and then label the clusters by the name of the variables that coincide with them.

5 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] J. Kruskal. M. Wish, *Multidimensional Scaling*. Sage Publications, 1977.
[2] M. Meyer, A. Barr, H. Lee, M. Desbrun, "Generalized Barycentric Coordinates on Irregular Polygons," *Graphics Tools*, 7(1):13-22, 2002.