Vector Data Viewer for Distribution Glance

Ryoji Fukuda
rfukuda@oita-u.ac.jp
Faculty of Science and Technology,
Oita University
Japan

Yuki Toyosaka
toyosaka@ip.kyusan-u.ac.jp
Faculty of Infomation Science,
Kyushu Sangyo University
Japan

Abstract:
We developed a software to plot vector data for a rough understanding of their distribution. For higher dimensional data, we created selection and change functions for view axes. Moreover, we analyzed approximation methods for determining the density of a vector distribution by using local information of vector data. In addition, we created a density viewer for vector distribution.

1. Introduction

Researchers use various vector data for analyzing scientific or engineering problems, and sometimes these data are used to distinguish several targets. The discriminant and cluster analyses, for example, are most commonly used for distinguishing vector data. In both these analyses, stochastic behaviors of data are important targets in several situations. Consider a case in which the Mahalanobis distance is used; approximations of local distributions are not easy if the data follow a multi-modal distribution. In this paper, we explain how our system visualizes the distribution of vector data. The purpose of the proposed system is to understand the distribution of multidimensional data; this may help us understand, for example, the difficulty in distinguishing two categories and in selecting a delicate discriminant method.

We used various numerical data and low-dimensional vector data in our previous studies ([1],[2], [3]). In the current study, we considered multidimensional vector data. Assume that the dimension is up to ten, which is not considerably high. In the proposed system, the data are expressed using several two-dimensional (2-D) scatter plots. These are orthogonal projection images, that is, images are translated using orthogonal projections, change in axes scale, and parallel shifts. We must select two directions to create a projected image. First, the system creates a standard image, for which the axes are the primary and secondary components. Moreover, the system randomly selects an axis. If one direction is selected as a display axis, the other axis is randomly defined. The system also approximates the distribution density for each selected point. In this study, we considered two approximation methods for density approximation. One method is a simple count of points in a constant radius ball, and the other is the reciprocal value of average distance in a k -neighborhood. The k -neighborhood of point $\vec{p}$ is a point family consisting of k -nearest points. This method is a comparatively simple model like a distance-based model ([4], [5]) and has the merit of being easy to use.

The k - neighborhood (a neighborhood defined using *k*-distance) was originally defined to analyze local outliers ([6]), and the k -distance was defined in [6] to analyze the local outlier factor (LOF). The aim of our analysis is to find outliers by using the density degree of a data-set; this degree value is called LOF. Our study uses the concept of k -distance. The k - neighborhood is determined as the k -th nearest points group. Therefore, if there is an isolated small group, we cannot find other group elements through a neighborhood search. Then, we consider new neighborhood considering direction from a center. The details are described in Section 4.3.

We should grasp neighborhood distance information to approximate densities. Let n be the data size, then $n(n-1)/2$ is the distance calculation if the computation time cannot be reduced. We

considered a 2-D partition by using the primary and secondary components, and every point was placed into some partition block. Next, we calculated distances of points near blocks.

In engineering, it is important to analyze vector data to consider a problem, and for a university student, it is important for understanding multidimensional statistical concepts or properties, for example, to obtain a rough grasp of the distribution of raw data. In such situations, if the dimension is not less than two, we would not be able to grasp raw data distribution. Even in a three-dimensional (3-D) case, we often change our view points in viewing a 3-D scatter plot. We need certain functions to change such view points. Thus, we propose a standard way to select certain view points, and an easy method to change them. As such we would be able to grasp the distribution of vector data.

2. System Outline

The purpose of the proposed system is to help grasp the data distribution easily. We have two view modes: direction and separation. The direction view is a mode to grasp the data distribution with respect to one direction, whereas the separation view infers to the linear separation by a hyperplane. In this section, we explain how we control them by using our system.

2.1 Data Format

Here, we consider multidimensional vector data and assume that the dimensions range from three to ten. There is no merit in using our system for one-dimensional (1-D) or two-dimensional (2-D) data. In theory, there are no upper bounds; however, if the number of dimensions is very high, it may be too complex to understand using our system. Two types of data with the same dimension can be input in the system, and each data set is given by a text file written in our format. Figure 2.1 displays a part of the data file used.

```
// Sample data
dim=3
num=1000
-0.327759,3.725040,4.911717
2.394533,-0.603544,-1.137942
-0.931709,4.462387,1.214458
1.404485,0.010095,4.717296
1.502153,4.592601,-0.388975
```

Figure 2.1 Format of data file

The first line is a comment line, which starts with "//." The second line defines the data dimension, and the third line defines the maximum data size. The lines below these definitions describe one vector datum. If the number of lines exceeds the value of "num," the system inputs the first "num" data and ignores the rest. If the number is less than "num," "num" is replaced by the input number.

2.2 Configuration File

Some parameters can be set or reset by using the configuration file. The following table lists the parameters in the configuration file.

Table 2.1 Parameters in a configuration file

Parameter	Expression	Comment
dim	dim=*d*	*d* must be the same as dim in data file.
range	range[*n*]=(-2.0, 2.0)	$n=0,1,\ldots,d-1$
color	color[*n*]=(255,0,0,)	$n=0,1$
dot size	dot size=2.0	line width and dot size(pixel).

When a data file is loaded, all parameters are defined according to the data. We can change these values by using the configuration file if necessary. The value of "dim" must be the same as the number of dimensions in the data file. Based on the dimension range, data values are scaled and shifted for each coordinate. The data positions (dots), and density values are displayed in the view area. The dot size and line width are defined using the value of "dot size," and the dot and line color are defined using "color."

2.3 System Screenshot

Figure 2.2 is a screenshot of the system. There are three view areas plotting vector data. There is a text box below each view area. This text area is to set the view axes, a scale, and a center point.
The large text box at the bottom is for setting directions. When the vector data are stored, the primary and secondary components are listed in this box. Moreover, two sets of data are stored components of another data set and combined data set. Any directions can be set in this area, and these can be used in setting text boxes.

Figure 2.2 Screenshot of the system

2.4 Setting of Display Area

We can set target data set, axis directions, the scale, and the center using setting text area for each view area. At most, two data sets are stored in the system. The target is "0," "1," or "0,1." "direction" and "sub" are axes directions; the vertical axis is "direction." The standard scale and the center point are defined in the system for each data axis, and we can change them using a configuration file. The point "center" defined in this text area is a 2-D point expressed by using the given directions, and the scale is a real value product to all relative vectors from the center.

2.5 Definition of Vectors

Vectors used for view axes are defined in vectors' definition area. When data sets are stored, the primary and secondary components of covariance vectors are listed in the text area. We can set arbitrary vectors in this area. Moreover, we can select "randV[*]" to define the "sub" direction. This is not a constant vector. When the user pushes the "Redraw" button, a random vector "randV[ ]" is set randomly with the condition that this is orthogonal to the "direction."

2.6 Density View

A given area is divided into 20×20 rectangles. The vector data are separated using this partition. We calculate the average value for each divided block. When a user clicks the left mouse button, we obtain the density value in the corresponding divided area.

2.7 Outputs

For each view area, we can save the results. The scattered graph is saved as a BMP file, and other contents are saved as a text file. In the text file, the number of data, two axis directions, the scales, and the range are listed.

3. Selection of View Axes

For arbitrary vector data, the value range varies from coordinate to coordinate. Thus, we need some basic translation before we visualize the data. In this section we explain the details.

3.1 Normalization for Each Coordinate

Let $\{\vec{x}_k\}_{k\le N}$ be a sequence of d dimensional vectors, and we denote $\vec{x}_k=(x_k^{(j)})_{j=1}^{d}$. Set

$$m^{(j)}=\frac{1}{N}\sum_{k=1}^{N}x_k^{(j)},\ s^{(j)}=\frac{1}{N}\sum_{k=1}^{N}\left(x_k^{(j)}-m^{(j)}\right)^2$$

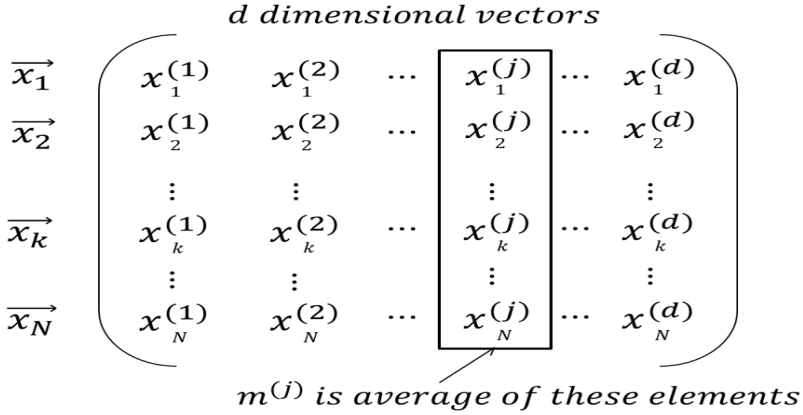

Figure 3.1 Coordinates of d dimensional vectors

We define the starting range for *j*-th coordinate $(r_{1,}^{(j)} r_2^{(j)})$ as follows.

$$r_1^{(j)} = m^{(j)} - C\sqrt{s^{(j)}}, \quad r_2^{(j)} = m^{(j)} + C\sqrt{s^{(j)}} \quad (\ C = 2.5 \text{ in the system}).$$

Next we define the normalized vector as follows.

$$\tilde{x}_k^{(j)} = \frac{x_k^{(j)} - r_1^{(j)}}{r_2^{(j)} - r_1^{(j)}} \quad , \quad \vec{x}_k{}' = (\tilde{x}_k^{(j)})_{j=1}^{d}$$

3.2 Setting of Directions

We use principal component analysis (PCA) for replacing direction vectors of components. Let *S* be the covariance matrix of $\{\vec{x}'_k\}_{k \le N}$. Eigenvectors $\vec{e}_1, \vec{e}_2$ correspond to the largest and second largest eigenvalues of *S*. We call them primary and secondary components of *S*. In a case where we have two data sets $\{\vec{x}'_k\}_{k \le N}$ and $\{\vec{y}'_k\}_{k \le N'}$, we calculate primary and secondary components for $\{\vec{x}'_k\}_{k \le N}$, $\{\vec{y}'_k\}_{k \le N'}$, and $\{\vec{x}'_k\}_{k \le N} \cup \{\vec{y}'_k\}_{k \le N}$. These vectors are listed in the text area "Definition Vector." We can also set an arbitrary vector by hand in this text area.

3.3 Setting of Random Vectors

Let $\vec{v}$ be a direction vector defined in the text area, and "sub" is defined as "randV[* v];" we set the other axis using a random number. Let *G* be a $d-1$ dimensional Gaussian random vector, and the distributions of all coordinates are *N(0,1)*, independent of each other. Thus, $\frac{G}{|G|}$ is a random vector uniformly distributed on the $d-2$ centered sphere with radius 1. Next, congruent transformation from $\mathbb{R}^{d-2}$ to hyperplane $\{\vec{x} : \vec{x} \cdot \vec{v} = 0\}$ realizes a uniformly distributed random vector orthogonal to $\vec{v}$ and $|\vec{x}| = 1$.

4. Density Approximation

Let N be the total number of vector data, and each vector is defined as $\vec{p}_k \in \mathbb{R}^{dim}$ ($1 \le k \le N$), and the approximated density values are defined as d_k. Next, the mean density value for the view point is calculated using the method in Section 2.6. The system has two quantification functions for local density. In this section, we explain these two calculation methods for the density value of point $\vec{p}_k$.

4.1 Dual Partition

For approximating densities, we use the Euclidean distance. Without having a concept for reducing the computation time, we calculated $N(N-1)/2$ distances to create a complete neighborhood system. We then prepared a 2-D partition and every point was set into the corresponding divided area. By using this partition, we reduced the calculation steps.

For given vector data $\{p_k\}_{k \le N}$, we set covariance matrix S. The primary and secondary components are the eigenvectors $\vec{e}_1$ and $\vec{e}_2$ corresponding to the largest and second largest eigen values of S, respectively. We then divided the projection $\{(\vec{e}_1 \cdot \vec{p}_k, \vec{e}_2 \cdot \vec{p}_k)\}_{k \le N}$ into $D \times D$ rectangular areas (we used $D = 10$ in the system). In addition, we defined the two sequences $\{x_i\}_{i \le D}$ and $\{y_j\}_{j \le D}$ satisfying

$$\#\{\vec{p}_k : \vec{p}_k \cdot \vec{e}_1 \in I_i^{(1)}\} = \frac{N}{D}, \quad (i = 1,2,\ldots,D, \quad I_i^{(1)} = [x_{i-1}, x_i])$$

$$\#\{\vec{p}_k : \vec{p}_k \cdot \vec{e}_2 \in I_j^{(2)}\} = \frac{N}{D}, \quad (j = 1,2,\ldots,D, \quad I_j^{(2)} = [y_{j-1}, y_j])$$

We set the values of $\{x_i\}_{i \le D}$ and $\{y_j\}_{j \le D}$ by using sorted sequences of $\{\vec{e}_1 \cdot \vec{p}_k\}_{k \le N}$ and $\{\vec{e}_2 \cdot \vec{p}_k\}_{k \le N}$, respectively. Thus, the number of set $\#\{\vec{p}_k : \vec{p}_k \cdot \vec{e}_* 2 \in I_j^{(*)}\}$ can slightly differ from $\frac{N}{D}$ if there are several $\vec{p}_k$ with the same $\vec{p}_k \cdot \vec{e}_*$ value.

We considerd dividing area $P_{i,j}$, for i, j ($i, j \le D$) as follows:

$$P_{i,j} = \{\vec{x} \in \mathbb{R}^{dim} : \vec{x} \cdot \vec{e}_1 \in I_i^{(1)}, \vec{x} \cdot \vec{e}_2 \in I_j^{(2)}\}$$

Thus, every $\vec{p}_k$ is set to some $P_{i,j}$. In the following two subsections, we explain the reduction methods for the construction of neighborhoods using this partition.

4.2 Standard Local Counts

Let r_0 be the neighborhood radius, which is defined in the configuration file (see Section 2.1). For each $k \le N$, we created a set. For constructing U_k, we defined the minimum distance $md(k, i', j')$ from $\vec{p}_k$ to $P_{i',j'}$ as follows:

$$md(k, i', j') = min\{|(x, y) - (e_1 \cdot x_k, e_2 \cdot x_k)| : x_{i'-1} \le x \le x_{i'}, y_{j'-1} \le y \le y_{j'}\} \quad .$$

We call area $P_{i',j'}$ "possible area" satisfying $md(x, i', j') < r_0$. We then constructed the neighborhood U_k as follows:

(1) Let $\tilde{P}_{i,j} = \{p_l : p_l \in P_{i,j}\}$, for each i, j ($0 \le i, j \le D$).

(2) For each $\vec{p}_k$, find (i, j) such that $p_k \in P_{i,j}$.

(3) Add all elements $\vec{p} \in P_{i,j}$ satisfying $|p_k - p| < r_0$ to U_k.

(4) Perform the same search, for $P_{i',j'}$ if $md(i', j') < r_0$.

(5) The search of $P_{i',j'}$ is conducted in close order while a possible area exists.

Consider the case in which $dim = 3$ and $P_{i,j}$ is an infinitely long square pole. Thus, two vectors in $P_{i,j}$ can be very far from each other. However, no neighborhood elements exist if the area is not a possible area. In other words, we can reduce the number of calculation steps through this partition.

4.3 Density of Four Directions k -neighborhood

In this section, we explain the density of four directions k -neighborhood(4DKN). First, we explain 4DKN. We assume that k is a multiple of four. Figure 4.1 shows the schematic of 4DKN.

The origin of the coordinate graphics (Figure 4.1) is the target data point. The projected image of data points is separated by axes from first to fourth quadrant. We constructed a neighborhood of the target data point using four quadrants as following:

- We selected $\frac{k}{4}$ nearest points from the center for each quadrant.
- The point number in a quadrant can be less than $\frac{k}{4}$ if there are not enough points in this quadrant.
- We used dual partition defined in Section 4.1 to reduce the calculation time.

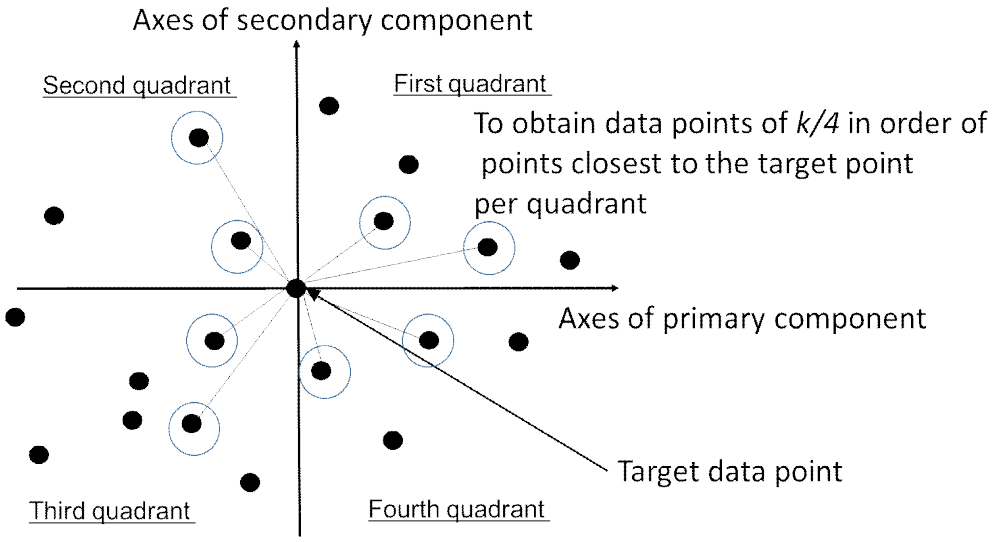

Figure 4.1 Making of k -neighborhood($k=8$)

We remark that the distances between the center points and other points are calculated as higher dimensional Euclidian space, and not the projected 2-D space.

4.4 Comparisons

In this section, we show the comparison of the proposed density and standard local counts using random numbers. Consider a 3-D Gaussian distribution with the mean vector $\vec{m}=(2,1,5)$ and the co-variance matrix $S=\begin{pmatrix}2&0&0\\0&3&0\\0&0&1\end{pmatrix}$. Next, by using random numbers, we prepared $N=1000$ vectors. By using $\vec{m}$ and S, we can obtain the correct likelihood value for each point. We compared these values with 4DKN values and standard local counts. We selected parameters $k=20$ (4DKN) and $r_0=0.6$ (standard local counts) by considering the size of both neighborhoods.

Figures 4.2, 4.3 and Table 4.1 show the relation and correlation between calculated variables. Figure 4.2 shows the correlation between density and likelihood of each data point, and Figure 4.3 shows the correlation between standard local counts and likelihood of each data point. As can be seen, both of correlations are highly positive. Table 4.1 shows that the correlation coefficient of 4DKN density at more than 0.8 is higher than the correlation coefficient of standard local counts. Therefore, we concluded that the precision of 4DKN density is good.

Table 4.1 A correlation between variables and likelihood

	Density using 4DKN	Number of data in r_0
Correlation coefficient	0.8	0.76

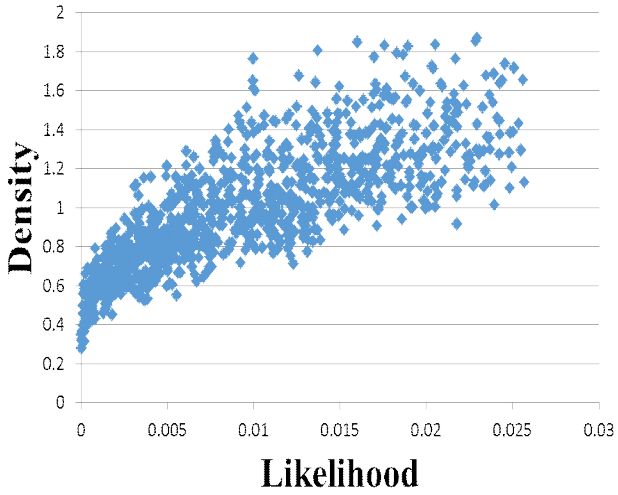

Figure 4.2 Correlation between density and likelihood of each data point(k=20)

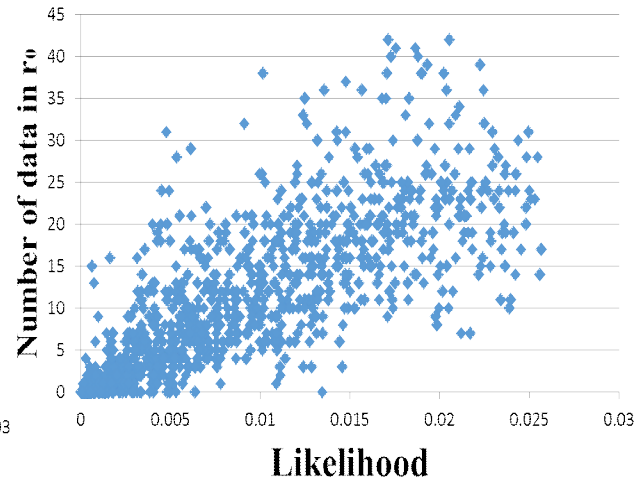

Figure 4.3 Correlation between standard local counts and likelihood of each data point(r_0=0.6)

5. Conclusion

We have created a system. We developed a software to plot vector data for a rough understanding of their distribution. We have some functions to control view angles in the system. We analyzed approximation methods for determining the density of a vector distribution by using local information of vector data. In addition, we created a density viewer for vector distribution. Our study use standard Euclidean distance, we will seek a more effective way to calculate the distance (image-based distance, Mahalanobis distance, for example). Furthermore, we will try some incorporation of our system into matrix-based systems such as Matlab. These are our future problems.

References

[1] Toyosaka, Y., Ogino, Y., and Fukuda, R., Evaluation of Mathematical Information Created by a Text-Based Communication Tool for Visually Handicapped, Proceedings of 15th International Conference on Computers Helping People with Special Needs, Universal Learning Design, 2016, pp. 45-52.

[2] Fukuda, R., Toyosaka, Y., Text-Based Communication Tool for Mathematical Documents for Visually Handicapped, Proceedings of the 20th Asian Technology Conference in Mathematics, 2015.

[3] Fukuda, R., Kojo, M., Graphic Input System in Elementary Geometry for Nonvisual Communication Proceedings of the 19th Asian Technology Conference in Mathematics, 2014.

[4] Knorr E. M., Ng R. T., Finding Intensional Knowledge of Distance-based Outliers, Proc. 25th Int. Conf. on Very Large Data Bases, Edinburgh, Scotland, 1999, pp. 211-222.

[5] Knorr E. M., Ng R. T., Algorithms for Mining Distance Based Outliers in Large Datasets, Proc. 24th Int. Conf. on Very Large Data Bases, New York, NY, 1998, pp. 392-403.

[6] Breunig, M. M. Kriegel, H. P. Ng, R. T. Sander, J., LOF: Identifying Density-based Local Outliers , ACM SIGMOD Record 29: 93, 2000.