# Reference Point Transformation for Visualisation

**Cheng G. Weng** **Josiah Poon**

School of Information Technologies,
J12, University of Sydney,
Sydney, NSW, Australia 2006,
Email: {cheng, josiah}@it.usyd.edu.au

**Abstract**

Visualisation of multi-dimensional dataset can be very useful for data mining purposes. This paper describes a simple visualisation technique to reduce a high dimensional dataset into a 3D space. Our aim is to design a method that is simple, easy, computationally cost-effective and able to give a reasonable visualisation of the dataset.

The original dataset is projected to a lower dimensional space via geometric metrics, while the proximity of the original data points is approximately preserved. The idea behind our data transformation is the concept of triangulation, which is applied through the use of reference points. In our study, we compared our method with the Principal Component Analysis (PCA) and Random Projection (RP). The results suggest that: when compared with PCA, our method can deliver a comparable visualisation of the dataset at a lower cost; when compared with RP, our method yields better visualisations at a similar cost.

*Keywords:* Data visualisation, Dimensionality reduction, Random projection

## 1 Introduction

"A picture is worth a thousand words", a proverb that well describes why visualisation can be a powerful tool for gaining deeper insight into difficult problems. A good visualisation should be able to convey a story of what is being visualised. Generally, different domains require different types of visualisation techniques, for instance, we use stock charts to show stock market data and maps to display physical locations. It seems that the human brain is accustomed to handling information in graphical forms. A good example which shows that graphs are better analytical tools would be to compare reading experimental results from a 100x100 table, as oppose to reading them from a scatter plot. For any non-trivial sets of experimental results, it is generally easier to spot trends in the scatter plot than in the table. So a good visualisation should help to organise complex information into easy to understand structures. In this paper, the domain that we will try to visualise is the vector datasets, i.e. a collection of well-structured and stationary dataset, where each example in the dataset is represented by $n$ number of features.

In general, there are some questions that need to be considered in order to construct a good visualisation for any vector dataset. Below are some of the basic questions that will help us shape our method:

1. Meaningfulness: Can the user *interpret* the visualisation in ways that would help them *learn* more about the dataset?
2. Interactive: Does the visualisation need to provide a *feedback* function to interact with the user and give real-time responses?
3. Computation: How long will it take to generate the visualisation? Does it scale up well with more data, i.e. more examples, higher dimensionality, or both?
4. Limitations: What would be the scope of the visualisation? Can it work with different types of attributes, or maybe an arbitrary number of dimensions?

Based on these questions, we have developed a data transformation technique to allow users to visualise datasets in an interactive environment. This technique uses different reference points to position examples in the dataset, therefore, we refer to it as the *kRef* method. While it is not meant to be an accurate approach as compared with other dimensionality reduction methods, we recommend it as a good visualisation alternative for practical reasons: it scales up well to large datasets, it is cheap to compute, it is memory efficient, it can be computed in parallel, it has no dimensionality restrictions, it is quick to implement and, most importantly, it is able to preserve a reasonable proximity of the original data points in order to deliver a meaningful visualisation of datasets.

In the next section, we will provide some related works, this will then be followed by Section 3, which is the main content of the paper and it describes our technique in detail. In Section 4, we will demonstrate the results of comparing our technique with PCA and RP, and in Section 5 we will provide some discussions and future works.

## 2 Related Works

Visualisation is one of the hot topics in computer science and there are many techniques that can be used to visualise multi-dimensional datasets. One of the early works for visualising multi-dimensional datasets was the parallel coordinates method (Inselberg, 1996). Parallel coordinates provides a 2D representation of any given dataset. A simple illustration is provided in Figure 1. The advantage of this method is that users can visualise the pairwise relationship between different dimensions in a fairly pleasant manner, but the drawback is that it can be very difficult to visualise when the dataset has a high dimensionality. There is also a novel approach of using human faces to represent different examples in the dataset (Morris

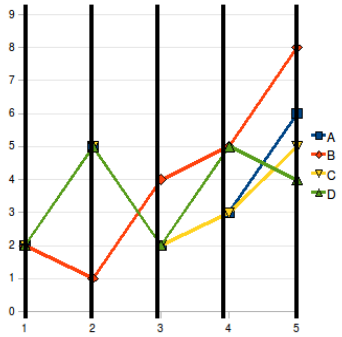



Figure 1: Parallel coordinates
An illustration of parallel coordinates. The $x$-axis is the dimensions/attributes and the $y$-axis shows the different values of the attributes. Examples are shown as lines connecting each dimension. In this figure, all four examples have the same value for Attribute 1, but they split into 2 groups on Attributes 2. In the last attribute, all four examples have a different value.

et al. (2000)). In this approach, multiple faces are presented, one face per example and each face have a number of adjustable facial features to represent different dimensions. For example, if one chooses to use the mouth to represent the first attribute, then the sizes of the mouth could be adjusted according to the value of the first attribute. In essence, this method makes use of the human's ability to spot similar and dissimilar faces, which translates to similar and dissimilar examples based on a combination of attributes. This method seems to be more scalable to dimensionality than the parallel coordinates method, but it is feasible only when dealing with a handful of examples.

Other related works in the data mining community are the dimensionality reduction techniques, such as PCA (Jolliffe (2002)), self-organising maps (Kohonen (2000)), random projection (Bingham and Mannila (2001); Lin and Gunopulos (2003); Blum (2006)) and multi-dimensional scaling (Borg and Groenen (1997)). For a good survey paper on dimensional reduction techniques, Fodor (2002) has provided a comprehensive coverage. Essentially, our technique also belongs to the category of dimensionality reduction, and the main challenge of dimensionality reduction is to perform feature selection with minimum information loss. The reduced dataset can then be utilised for other purposes, such as visualisation, data compression and efficient learning.

From an appropriate perspective, random projection (RP) is similar to our work because of the similar dataset transformation process. Random projection is based on Johnson-Lindenstrauss lemma (Johnson and Lindenstrauss (1984)), which states that if we project a set of $m$ data points from $n$-dimensional space onto a random k-dimensional space, such that $k \geq O(\varepsilon^{-2}log(m))$, then the pairwise distances are preserved within $\varepsilon$. This minimum bound suggests that $k$ could potentially be much smaller than $n$. To apply random projection, the original $m$ by $n$ data matrix is reduced, by multiplying a random $n$ by $k$ matrix, with the constraint that the column vectors in the random matrix are unit vectors. After the description of our work, in the discussion section, we will point out the differences between our method and random projection.

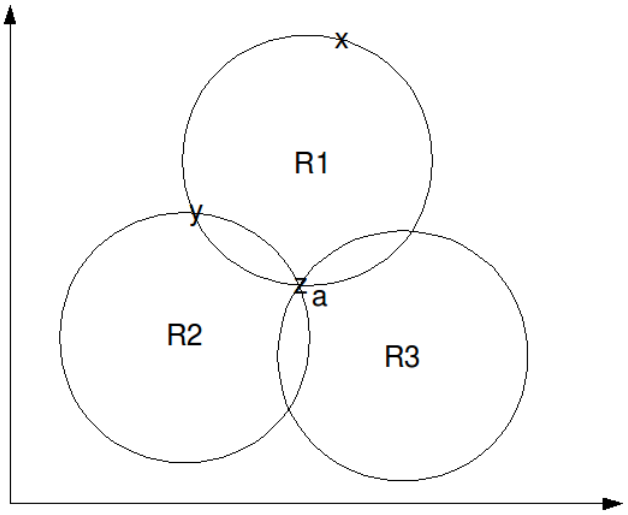



Figure 2: Reference Points
In this 2D example, we have 3 reference points (R1, R2 and R3) and 4 data points (x, y, z and a). The circles represent equal Euclidean distances from the centre reference points, i.e. the distances between R1 to x, y and z are the same.

## 3 Reference Point Method

As a motivational example of our proposed technique, we use Figure 2 to show that, using reference point $R1$ alone is not possible to tell apart $x$, $y$ and $z$ by the distance measurement, as they have the same distance. However, the situation can be improved with an additional reference point, $R2$, because $x$ will have a different distance to $R2$ than $y$ and $z$. Lastly, in order to separate $y$ and $z$, we need another reference point, $R3$. Thus, with 3 reference points, we can uniquely identify $x$, $y$ and $z$ with their distance information to each reference point. Therefore, the basic requirement for the method to work is to have at least 3 reference points, and because we can select any $k$ number of reference points, provided that $k \geq 3$, hence we called this method *kRef*.

Furthermore, we have put a data point, $a$, to show that, in the *3Ref* space, $a$ and $z$ should have similar distance measurements to each reference point because they are located at a close proximity; this is why *kRef* transformation is able to preserve the approximate pairwise distances in a lower dimensional space. One can see this transformation as projecting a dataset from the view of each reference point and combining different views to approximate the original image. However, the distances in the *kRef* space will be distorted due to the projection based on a distance metric.

### 3.1 Algorithm and Run-time

The *kRef* method will transform the original dataset, $X$, into a new space, $kRefTransform(X)$:

$$X = \begin{bmatrix} x_{00} & \dots & x_{0n} \\ \vdots & \ddots & \vdots \\ x_{m0} & \dots & x_{mn} \end{bmatrix}$$

$$kRefTransform(X) = \begin{bmatrix} kRef(x_0)_{00} & \dots & kRef(x_0)_{0k} \\ \vdots & \ddots & \vdots \\ kRef(x_m)_{m0} & \dots & kRef(x_m)_{mk} \end{bmatrix}$$

**Algorithm 1** Algorithm for transform a dataset to *kRef* space (in Python).

```
def kRefTransform(dataset, referencePoints):
    newDataset = [] # create an empty dataset list
    for example in dataset:
        newExample = [] # create an empty example
        for ref in referencePoints:
            newExample.append(dist(ref,example))
        newDataset.append(newExample)

    return newDataset
```

The *kRefTransform*() method returns a new dataset by taking the original dataset, $X$, which contains $m$ examples and $n$ attributes; it then uses the function *kRef*() to map each example into $k$ attributes, where $k$ equals to the number of reference points. The basic algorithm is described in Algorithm 1, which is written in Python programming language. In Algorithm 1, we defined a method, *kRefTransform*(), that takes a list of examples (*dataset*) and a list of reference points *(referencePoints)*, then returns with a list of examples in k-dimensions (*newDataset*). The *dist()* function in our implementation uses Euclidean distance, because we have found it to be a good choice in our experiments. Although the algorithm shows only one nested loop, but the Euclidean distance function actually contains another loop that goes through all the attributes in each *example*. So in the worst case, the algorithm has a run-time of $O(knm)$, where $k$ is the number of reference points, $n$ is the number of attributes and $m$ is the number of examples in the dataset. In most cases, both $k$ and $n$ are fixed constants and only $m$ would grow, so the average run-time is $O(m)$.

### 3.2 Fundamentals

We have identified 4 fundamental factors that can impact the resulting *kRef* space: the position of the reference points, the number of reference points used, the original dimensionality of the dataset and the distance metrics employed. These factors are inter-related, because different dataset dimensionalities and different number of reference points may require a different placement of reference points, which may in turn be affected by the distance metric. But, as a first attempt, we will analyse them independently.

Before we go through the factors, we will first propose a quality assessment measure for the dataset transformation.

#### 3.2.1 Quality measurement

The measurement is intended to evaluate how good a technique is at preserving the proximity of data points in the geometric space. We define this measurement as the correlations between the pairwise distances in the original space and that in the transformed space. For efficiency reasons, if the dataset is large, we will only examine a random subset of the dataset; our empirical results suggest that a sample of 20% seems to give a reasonable estimate of the true value.

For a given set of data points, we need to first generate a matrix that contains $m \times m$ pairs of distances:

$$\begin{bmatrix} dist(x^{(0)}, x^{(0)}) & \cdots & dist(x^{(0)}, x^{(m)}) \\ \vdots & \ddots & \vdots \\ dist(x^{(m)}, x^{(0)}) & \cdots & dist(x^{(m)}, x^{(m)}) \end{bmatrix}$$

The *dist*() function calculates the Euclidean distance between two examples. This matrix is symmetric, i.e. the diagonal line contains zeros and the upper-right half is a mirrored copy of the lower-left half. Therefore, we can flatten the matrix into a vector by appending the rows together, i.e. $D^{sample} = [row(x^{(0)}), row(x^{(1)}), ..., row(x^{(m-1)})]^T$, and the size of $D^{sample}$ is $\mathbb{R}^{m(m-1)/2}$.

We compute the distances vector for both the original and the transformed datasets to get $D^{original}$ and $D^{kRef}$, then $correlation(D^{original}, D^{kRef})$ can be calculated with Pearson Product-moment Correlation Coefficient (Moore, 2006):

$$correlation(X, Y) = \frac{1}{M} \sum_{i=1}^{M} (\frac{X_i - \mu_X}{\sigma_X})(\frac{Y_i - \mu_Y}{\sigma_Y})$$

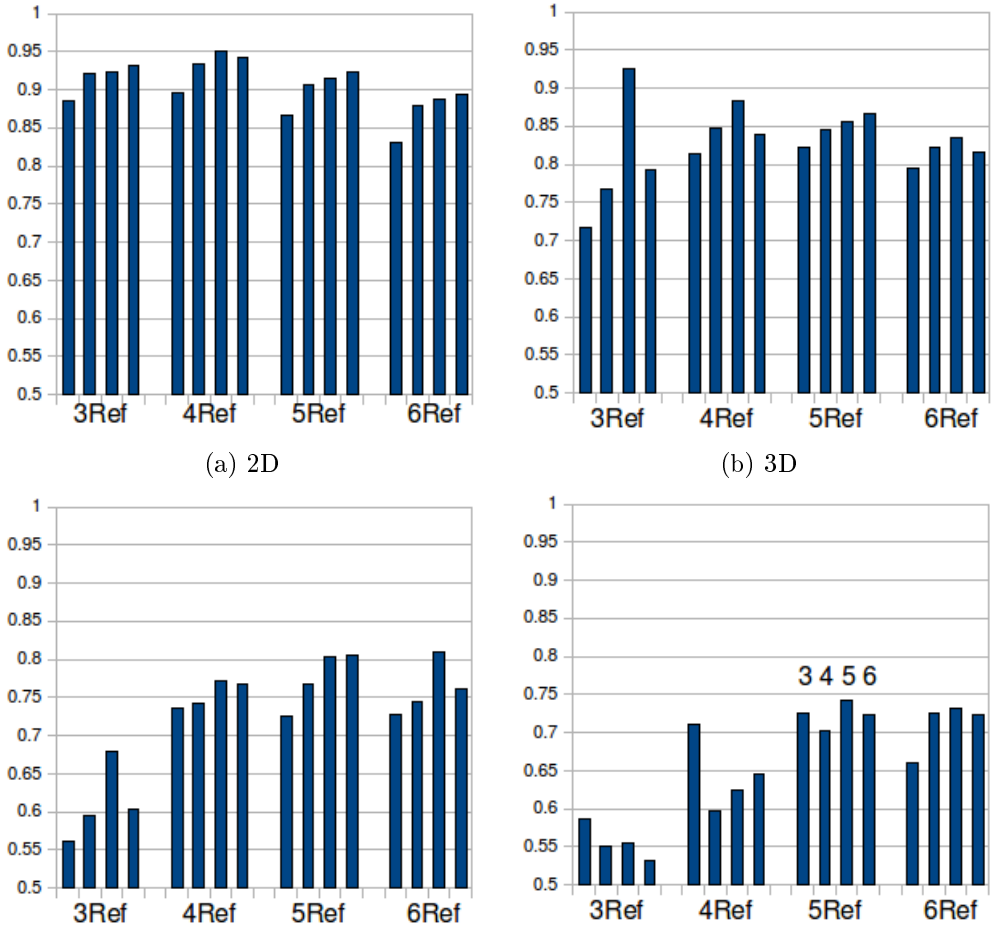
where $\mu_X$, $\mu_Y$, $\sigma_X$and $\sigma_Y$ are the means and the standard deviations for the corresponding vectors, and $M$ is the size of $X$, i.e. $|X|$. The function *correlation*() will produce a real number between -1 and 1 to indicate whether the vectors have a negative correlation (towards -1) or a positive correlation(towards 1), or simply no obvious correlation (around 0). A good quality data transformation should have a strong positive correlation, whereas a poor quality transformation is indicated by severe negative correlation, because it means that the original distances are distorted in the transformed space.

#### 3.2.2 Synthetic test datasets

Test datasets are used to examine the effects of each factor on the quality of the transformed space. A test dataset consists of data points at every possible feature space, e.g. if a dataset is described by 3 attributes and each attribute can take on 5 different values, then all possible spaces in this dataset will be $5^3 = 125$. We have generated 64 test datasets with all combinations from different settings: the original dimensionality $\{2, 3, 4, 5\}$, the number of reference points $\{3, 4, 5, 6\}$ and the possible attribute values $\{3, 4, 5, 6\}$.

#### 3.2.3 Factor 1: Positions of the reference points

It can be shown that the quality of the resulting *kRef* space will be affected by the placements of the reference points. Suppose, in Figure 2; instead of the stated 3 reference points, we choose $y$, $z$ and $a$ to be the reference points, then the new reference points will form a line. When this happens, the line is effectively a mirror-like projection, i.e. the data points on either side of this line will have a symmetrical counterpart that has the same distances to each new reference points. As a result of this mirror-like projection, two




(a) 2D (b) 3D

(c) 4D (d) 5D

Figure 3: Use test datasets to compare different factors
The $y$-axis is the quality measurement described in Section 3.2, so large value means higher quality. These 4 graphs represent 4 different original dimensions (2D to 5D), and in each graph there are 4 different clusters representing different numbers of reference points used (*3Ref* to *6Ref*), then each cluster is further divided into 4 bars, where each bar, from left to right, represents the number of possible attribute values (3 to 6). For visual consistency, the graphs are scaled to sit between 0.5 and 1.

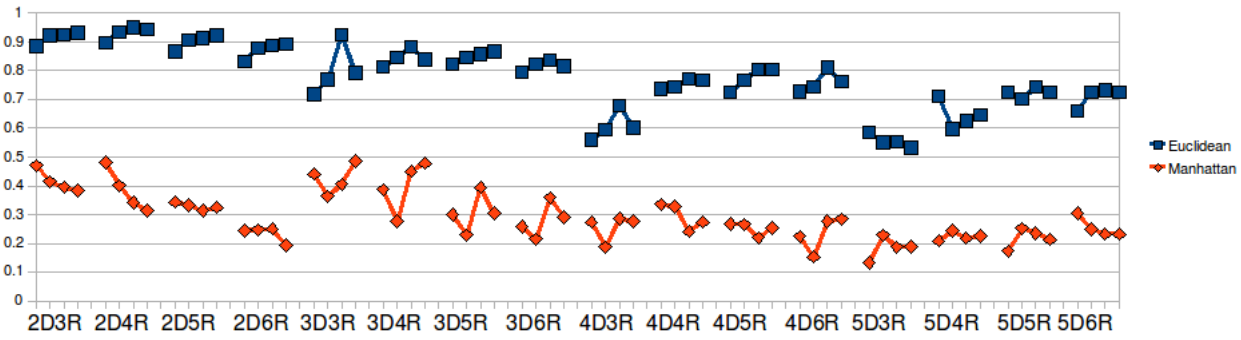



Figure 4: Compare distance metrics: Euclidean versus Manhattan
This graph plots the same set of experiments shown in Figure 3, but instead of bars, we plot the quality measures for each cluster as lines. Each small line segment corresponds to a cluster in the Figure 3, e.g. *5D3R* is the same as the *3Ref* cluster of *5D* graph. The upper lines in this figure are the performances of Euclidean distance, which is the same as the values in Figure 3. The lower line segments are the results when we employ the Manhattan distance instead.

different data points in the original space will overlap in the transformed space.

At the current stage, we have yet to work out a formal derivation of optimal positions for the reference points that would minimise the loss of information after data transformation. Although we do not have a formal proof, we do have 2 general rules that can yield a good performance in practise:

1. Do not place the points too close to each other, because this will result in data points having similar distances to each reference points. An extreme case would be to place all reference points at the same spot, which would be useless. Therefore, one should place the reference points far away from each other, preferably outside the dataset region.

2. The layout of the reference points should aim to produce a unique set of distances for different data points. This can help to reduce the information loss, because, at least, different data points in the original space will not overlap in the $kR$ space.

Based on these two rules, we placed our reference points around the maximum and minimum attribute values, and we also need to induce some randomness to ensure that the reference points do not have the same attribute values. In addition, we followed certain shape layout to make sure that the reference points are far away from each other, e.g. using a triangular layout for 3 reference points. An example set of 3 reference points to transform a 3D dataset would be $\{(0.10, 0.07, 0.98), (0.09, 0.95, 0.05), (0.97, 0.03, 0.04)\}$ (assuming the attributes are real numbers between 0 and 1).

#### 3.2.4 Factor 2: Number of reference points

Figure 2 shows that it requires at least 3 reference points to uniquely identify data points in a 2D space, but will the same observation be made also in a higher dimensional space? Unfortunately, this is another difficult theoretical question that we do not yet have an answer to, but in our experiment it seems to suggest that 3 reference points can still produce a unique set of distances in a higher dimensionality to differentiate different data points. We have found that, within each test dataset, every set of distances is unique. So, 3 reference points can still uniquely identify every possible data points in a 5D space.

Although unique identification was possible, the number of reference points used still seems to have an impact on the quality of the transformed space. As shown in Figure 3, using *3Ref* seems to be more unstable and worse than other choices of reference points. This effect is more so in higher dimensional spaces.

Another observation from Figure 3 is that, as the number of reference points increases, the quality of the dataset does not seem to get better, e.g. *4Ref*, *5Ref* and *6Ref* all seem to have comparable qualities regardless of their dataset dimensionality. We suspect this is due to the layout of the reference points, because although we are introducing more reference points, they are probably not at their optimal positions, so they may have been under-utilised.

#### 3.2.5 Factor 3: Original Dimensions

This factor seems to be the most dominating, as suggested by the results in Figure 3. When the dimensionality increases, the quality of the transformed space generally decreases.

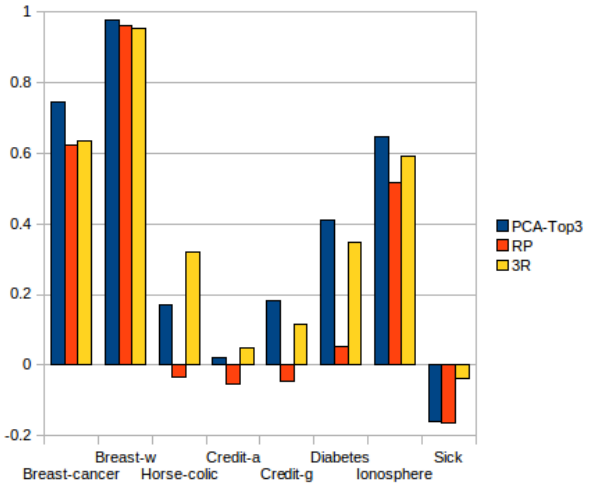



Figure 5: Comparison of transformation quality
The $y$-axis is the quality measurement described in Section 3.2. We compare PCA-Top3 (the first 3 principal components), random projection (RP) and 3 reference points (*3Ref*) in their ability to preserve the proximity of the data points.

#### 3.2.6 Factor 4: Distance metrics

We tested two different distance metrics: Euclidean distance and Manhattan distance. For this experiment, in the quality measurement process, we used the same distance metric as the one used for the dataset transformation, i.e. the Euclidean distance will be evaluated with Euclidean distance in the quality measure, and the same goes for the Manhattan distance.

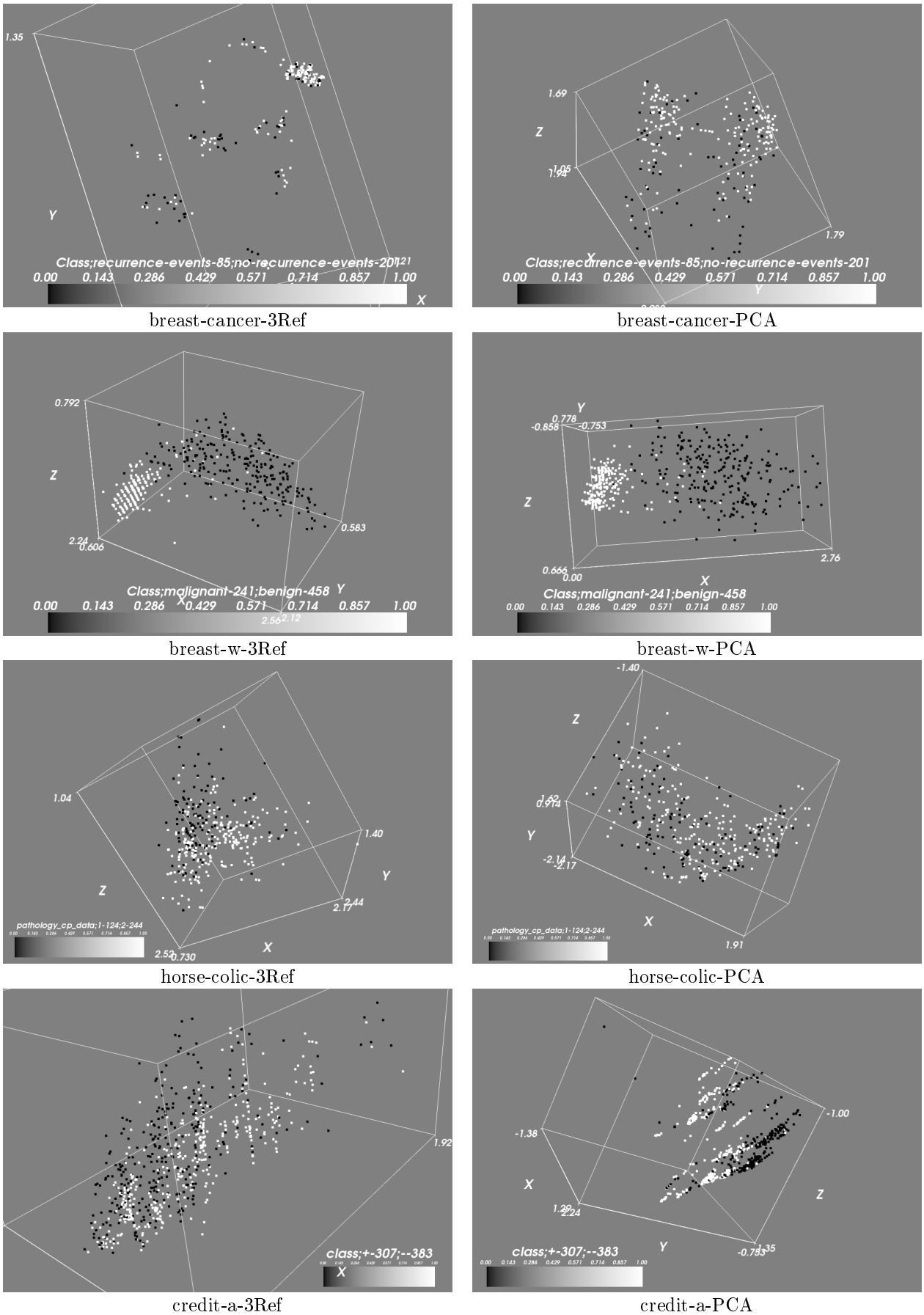
We used the same synthetic test datasets to compare the two distance metrics. The results are presented in Figure 4. We have found that the Euclidean distance is consistently better than the Manhattan distance in all settings.

## 4 Visualisations

For real world dataset visualisations, we used binary class problems from the UCI data repository (Asuncion and Newman, 2007). Some basic information about the datasets is listed in Table 1. In our experiments, we have used three tasks to compare our approach with PCA and RP. The first task compared their ability to preserve the proximity of the data points with the quality measurement. The second task compared the changes in the decision tree's learning performance, before and after the data transformation; the learning performance is measured with the area under ROC curve (Fawcett (2004)). In the third task, we looked at the actual visualisations they produced and see whether insights on the datasets can be gained.

Although 2D graphs look better on paper, but the minimum requirement for *kRef* to work is with 3 reference points, therefore, for all tasks, we will reduce the dimensionality to 3 for visualisation. The visualisations are done in an interactive environment[1], but, unfortunately, this is difficult to present on paper. We will try to minimise the loss of interaction by rotating the 3D models to the most clear angle that we can find. However, the depth information will be lost.

[1] We used mayavi for building the visualisations (http://mayavi.sourceforge.net/).




breast-cancer-3Ref | breast-cancer-PCA

breast-w-3Ref | breast-w-PCA

horse-colic-3Ref | horse-colic-PCA

credit-a-3Ref | credit-a-PCA

Figure 6: Comparing visualisations (Part1)
Comparing visualisations produced by 3 reference points (*3Ref*) and by PCA on 2-class problems from the UCI datasets.

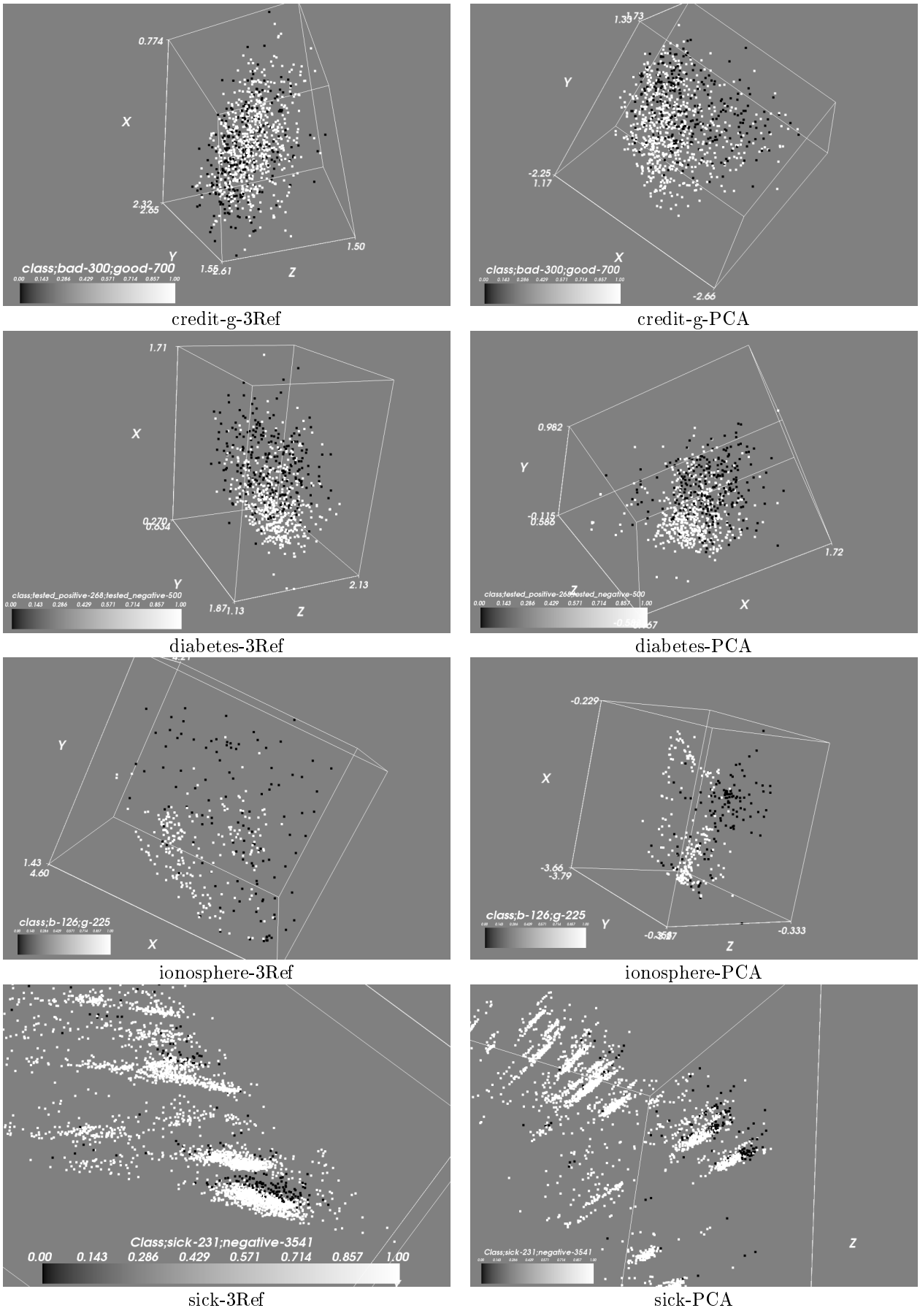



Figure 7: Comparing visualisations (Part 2)
Comparing visualisations produced by 3 reference points (*3Ref*) and by PCA on 2-class problems from the UCI datasets.

| datasets | AUC | Numeric | Nominal | Positive% | Size |
|---|---|---|---|---|---|
| Breast-cancer | 0.58 | 0 | 9 | 29.72 | 286 |
| Breast-w | 0.96 | 9 | 0 | 34.48 | 699 |
| Horse-colic | 0.49 | 7 | 20 | 33.70 | 368 |
| Credit-a | 0.89 | 6 | 9 | 44.49 | 690 |
| Credit-g | 0.64 | 7 | 13 | 30.00 | 1000 |
| Diabetes | 0.75 | 8 | 0 | 34.90 | 768 |
| Ionosphere | 0.89 | 0 | 34 | 35.90 | 351 |
| Sick | 0.95 | 7 | 22 | 6.12 | 3772 |

Table 1: Dataset information
This table shows some basic information about the datasets, including the performance of J48 (WEKA's implementation of C4.5 (Witten and Frank (2005))) in the area under the ROC curve (AUC). *Numeric* and *Nominal* denote the number of numeric attributes and nominal attributes in the dataset. *Positive%* is the percentage of the smaller class. *Size* is the total number of examples in the dataset.

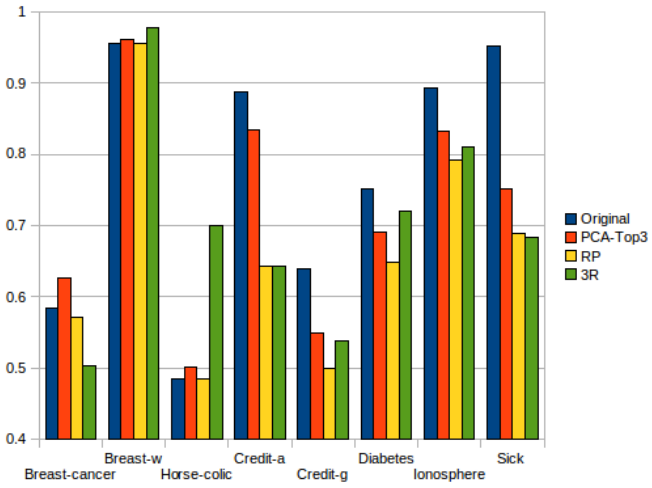



Figure 8: Compare the AUCs of J48
This figure shows how different data transformation impacts the area under the ROC curve. There are 8 bar clusters, representing 8 datasets. In each cluster, there are 4 bars, from left to right, denoting the AUC measures of the original dataset, PCA-Top3, RP and *3Ref*.

### 4.1 Task 1: Comparing quality measurements

The result for the first task is shown in Figure 5. The result shows that PCA is better in most cases except for *horse-colic, credit-a* and *sick*, but *3Ref* is not much behind. On the other hand, RP is the worse of the three in 7 out of 8 cases.

Since the quality measurement is the correlation between the before and after transformation distances, therefore a negative correlation means that the closer distances are getting farther, whereas the farther distances are getting closer. This scenario is considered poor quality, because it distorts the original image of the dataset. Negative numbers occurred 4 times in RP, this indicates that RP is distorting the dataset more than the other two methods. In the sick dataset, all three methods are negative, suggesting this dataset is quite difficult to preserve. The cause of difficulty is the abundance of binary attributes in the sick datasets, because in the distance calculation, binary attributes offers less numerical variations than numeric attributes. Therefore, it is harder to retain an accurate pairwise distances.

### 4.2 Task 2: Comparing learning performances

The results of the second task is presented in Figure 8, which shows that in 5 out of 8 cases, all dimensionality reduction techniques cause the learning performance to go down. If we compare *Original* with the next best AUC value, there are 2 cases, *credit-g* and *sick*, where the drop in AUC is relatively bigger than other cases. The drops in AUC are 0.09 for *credit-g* and 0.20 for *sick*. In both cases, their quality measurements in Figure 5 are also lower than others, except *credit-a*. The biggest AUC drop of 0.20 from sick dataset coincides with the result from task 1: the sick dataset does not project well onto lower dimensions.

Although the performances were dropped, but in most cases the drops were moderate, and because decision tree learner is learning based on the geometric information of the classes, therefore, it suggests that the data transformation technique is able to retain most of the geometric information, i.e. the relative geometric positions.

If we compare *3Ref* with others, *3Ref* does worse than PCA in 5 out of 8 cases, but does better than RP in 5 out of 8 cases with 1 tie.

### 4.3 Task 3: Comparing images

For the third task, we still used the AUC of a decision tree learner as a guide to help compare how much class information is preserved in the visualisations. This task is similar to task 2, but in this task, we try to stress that there are useful information when one has access to the actual visualisation. Task 2 was a quantitative analysis, but a lot of information about the geometrical structure was lost. In this task, without defining a specific aspect of the dataset to quantify, we used the dimensionality reduction techniques as a general purpose dataset visualiser.

In Figure 6 and 7, we present images of the UCI datasets with *3Ref*'s images on the left-hand side and PCA's images on the right-hand side. The colour of the classes is consistent in each pair of images, i.e. in both *breast-caner-3Ref* and *breast-cancer-PCA*, the black data points refer to the same class. In addition to the images, Table 1 provides the performance measurement of the decision tree on the same set of dataset. The observations for each pair of images are as follows:

**Breast-cancer** We believe that *3Ref* has the better image, because it shows different clusters for the black class (recurrence-event class); the white class also seems to form different clusters. The dataset looks more clear cut in the *3Ref* image than in the PCA image. We can also isolate the difficult region, which seems to be the big cluster at the top right corner in *3Ref*'s image. In PCA's

image, the dataset seems to be divided into two symmetrical clusters.

**Breast-w** Both images separate the black and the white classes into one sparse cluster and one dense cluster. The dataset looks reasonably easy to learn and this is confirmed by the good learning performance.

**Horse-colic** Both images seem to be more complex, because the black class and the white class are mixed together. This difficulty is also confirmed by the decision tree's performance. *3Ref*'s image suggests that the difficulty comes from the centre region where classes are more mixed, whereas in the PCA's image, the points are evenly mixed with no obvious clusters.

**Credit-a** The image of PCA seems to be better than *3Ref*, because PCA shows more separation between the two classes and the decision tree is also able to learn quite well from the dataset.

**Credit-g** This is another difficult dataset. Both images are complex and the classes are overlapped. Both images seem to show two clusters, and the source of difficulty lies within the middle region, where the two classes overlap.

**Diabetes** The images are still complex, but both images seem to show that the white class is concentrated more at a specific region. So it looks like the errors are coming from the invading white points into the sparse black cluster.

**Ionosphere** This dataset seems to be sparse in *3Ref*'s image. The white class forms two well-separated clusters, while the black class is sparsely distributed. It is not difficult to learn, because the classes are not mixed. Although it can be hard to see, but the two classes actually lie at a different depth in *3Ref*'s image. PCA seems to be clustering the classes more tightly, and it also shows two white clusters.

**Sick** This imbalanced dataset has a very skewed class distribution (6% rare class), but the learning performance is quite good. The image from *3Ref* shows that the black class (rare class) forms a well-separated cluster, sitting just above a big white cluster. Similar pattern can be seen in PCA's image, but the division between the classes is not as clear as in *3Ref*'s image.

We did not show RP's images because RP's images were not as informative since the data points are tightly packed into lines or planes. For example, when we applied RP on the breast-cancer dataset, as shown in Figure 9, RP produced 4 lines of points, which is comparatively less rich in geometric variation than the other two methods.

## 5 Discussions

In our experiments, we have compared the correlation scores and the change in learning performance of *kRef* with PCA and RP. We have demonstrated that *kRef* transformation is capable of producing comparable visualisations of datasets. We have also shown that data visualisation can be a useful tool for understanding datasets, and it can also be more informative than a single learning performance measurement.

When producing data visualisations, there exists a tradeoff between the accuracy and the computational speed, although *kRef* is slightly less accurate than PCA, *kRef* is much faster. With a little sacrifice in accuracy, *kRef* is a better choice in practise because large datasets are abundant.

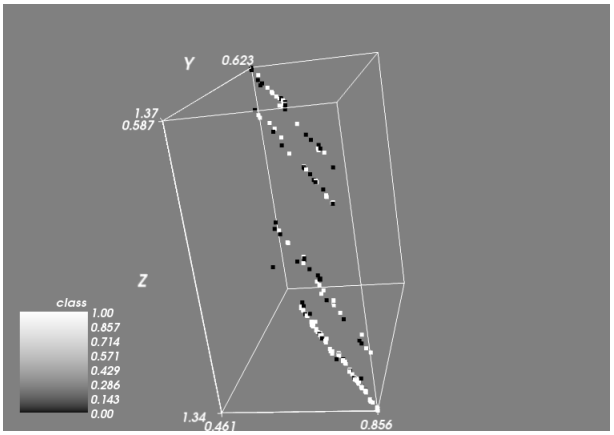
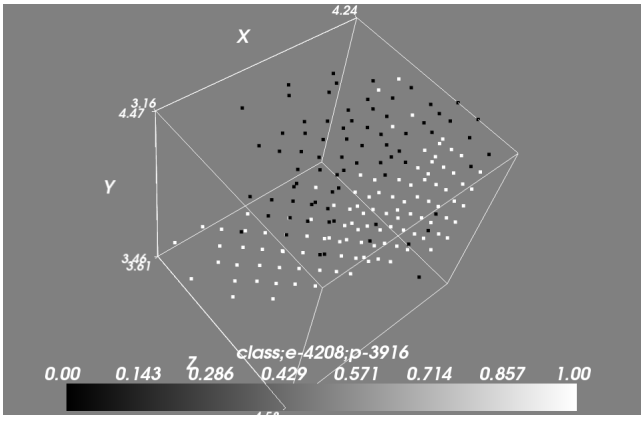
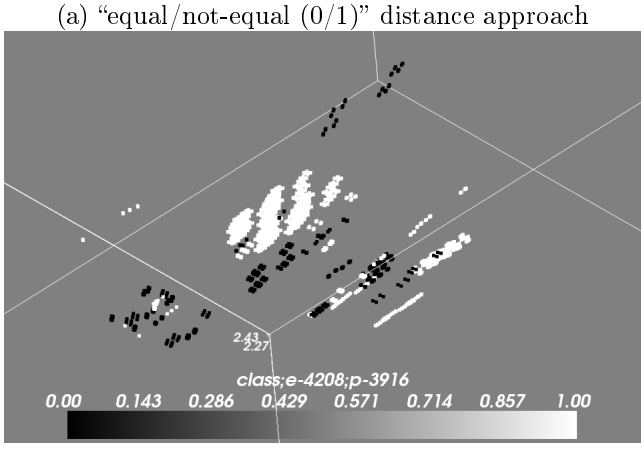



Figure 9: Breast-cancer dataset with RP
Perform random projection on breast-cancer dataset. The data points form 4 lines in the RP space.




(a) "equal/not-equal (0/1)" distance approach




(b) VDM distance approach

Figure 10: Dealing with nominal attributes
These two figures are used to show the difference between the two different approaches of handling nominal attributes. The dataset used is the mushroom dataset from the UCI data repository. Black points represent the *edible* class and white points represent the *poisonous* class. The dataset size is 8124, but in figure (a), the 0/1 approach has placed a lot of data points at the same spot, making it rather difficult to see all 8124 different data points. On the other hand, the VDM approach in figure (b) is able to spread out the data points more so we can see more interesting trends in the datasets.

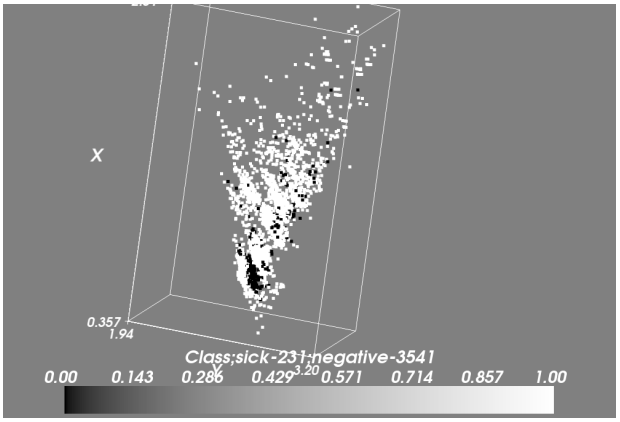



Figure 11: The data points distribution shape The *sick* dataset looking from afar. The dataset shows a cone shape distribution in the *kRef* space.

### 5.1 Implementation issues

The first issue was how to deal with nominal attributes, because they may not be ordinal attributes. In our first attempt, we used the simple equal/not-equal approach: assign 0 distance if the two attributes have the same value, and assign 1 otherwise. Unfortunately, this approach produces very dense clusters, therefore making it hard to visualise because a lot of points are packed at the same point. So, we tried Value Difference Metric (VDM) distance (Wilson and Martinez, 1997), which essentially replaces nominal values with their respective occurrence rates. The VDM distance approach is able to give data points more variations, which allows the data points to spread out, thus leading to better visualisations. The mushroom dataset is used as an example to show the difference between the two approaches. Mushroom dataset has 22 nominal attributes and 8124 examples. It is considered to be one of the easier datasets in UCI, which is also evident from our visualisation. Figure 10(b) shows that the two classes are well separated and both classes have clean clusters that do not overlap. It should be noted that the black and white groups in Figure 10(b) are sitting at a different depth, and only appears to be overlapping because of the 2D perspective.

The second implementation issue was that: if there are numeric attributes that have large values, then the distance measurement will be dominated by these numeric attributes. So, in order to allow an even contribution from all attributes, we need to normalise the numeric attributes.

### 5.2 *kRef* space restriction

There are regions in the transformed *kRef* space that will not be used, for example, under a triangular layout, it would be impossible to have a data point that has 0 distances to all three reference points. So, this suggests that the transformed space will have a hyper-concave shape distribution, where the bottom is sitting at the centre of the reference points. As shown in Figure 11, a zoomed out image of the *sick* dataset, the transformed dataset is distributed in a cone shape.

### 5.3 Comparison with other methods

*kRef* and PCA share similar concepts; they both rely on data point projection, but the difference is that: in *kRef*, the projection is done via different points; whereas in PCA, the projection is done via different eigenvectors (principal components). In the *kRef* method, each data point in the datasets is re-described by all the reference points, but in PCA, the eigenvectors can be used independently. The success of the *kRef* method relies on reference points to generate distance with the highest variance so that the data points will not overlap in a condensed space. Similarly, PCA tries to project onto eigenvectors that would reflect the highest variance.

In terms of computation, *kRef* has a better run-time than PCA. The run-time of the *kRef* method is $O(mn)$, given $m$ number of examples and $n$ number of features. In comparison, PCA is more expensive to compute, with a run-time of $O(min(mn^2, nm^2))$. Thus, given a large enough $m$ and $n$, the run-time of *kRef* method is $O(m^2)$, whereas PCA runs at $O(m^3)$. The run-time of PCA is derived from the run-time of Singular Value Decomposition (SVD), because computing the covariance matrix directly would be too costly if the dataset dimensionality is high, e.g. 10000 features would require a matrix size of 10000x10000. So, the run-time of PCA is dominated by SVD computation, which has a run-time complexity of $O(min(mn^2, nm^2))$.

Another advantage of the *kRef* method over PCA is that the *kRef* method can run in parallel, because most of the computations in *kRef* are not dependent, so the calculations can be distribute across different machines. This parallel distribution is quite attractive because large datasets can be partitioned and stored in separate machines, which is also more memory efficient.

The *kRef* method is similar to random projection as well, because they both transform the dataset in a similar fashion, but our approach is motivated by triangulation, so our random matrix selection is quite different from the ones used in random projection. Also, our method uses distance function in the projection calculation, whereas random projection uses dot products.

We have implemented random projection proposed by Achlioptas (2001) and tested it in our experiments. As shown in Figure 9, random projection generates less geometrical variances. This suggests that while random projection is useful as a dimensionality reduction method and enjoys a theoretical bound, its constraints on the random matrix makes it unable to provide good visualisations in 3D space.

### 5.4 Reflection

In order to address the question of visualisation meaningfulness, as proposed in the introduction, we must first define the *meaning* in the context of the visualisation. For example, in a stock OHLC (open,high,low,close) bar chart, the meaning could be the underlying supply and demand, which is what OHLC bar chart tries to visualise. In the context of this work, we define the meaning as the underlying geometric structure of the dataset. Therefore, for a given dataset, there exists a true "meaning" that different data visualisation methods all try to capture.

Our visualisation is meaningful because it is able to retain most of the geometric structure of the dataset. The accuracy of the *kRef* method is only slightly behind PCA, i.e. in 6 out of 8 cases in Figure 8 the gaps were within 0.1. Moreover, it is possible if a better layout of the reference points can be derived, the accuracy of the *kRef* method could be further improved.

Like any generic dimensionality reduction method, *kRef* will work with any number of dimensions or any dataset size, however, the usefulness of the visualisation depends on the users. Because visualisations are inherently subjective, the same visualisation may be

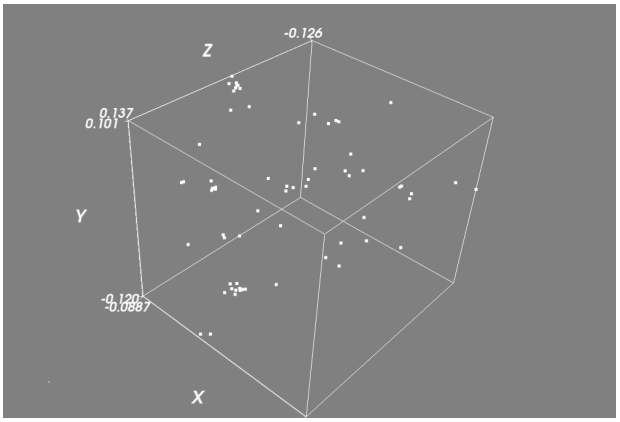



Figure 12: Visualise feature correlations for *audiology* An illustration of the *kRef* method's extension work. In this visualisation, the features are treated as examples and the distance function used here is the correlation function. So, in this space, the clusters imply highly correlated features.

interpreted differently, which could lead to different insights. This creativity is the strength of the data visualisation methods.

### 5.5 Future works

The interactive aspect of the visualisation was not presented in this work because it would require a different evaluation process. However, an interactive environment may have features that give the users the ability to rotate the 3D model, zoom-in from any angle, select a sample of data points and check the data point information by clicking the dots. These interactions allow the user to explore and conceptualise the difficult parts of the dataset, which can potentially lead to a better formulated learning approach for the problem. So, it would be interesting to investigate this active learning approach and see how best to utilise the insights from the visualisations.

It is also possible to apply *kRef* to visualise the feature space instead of the example space. This can be done by taking a transpose of the dataset matrix, so the features become examples and examples become features. The similarity function also needs to be changed from the Euclidean distance function to a correlation function for a more meaningful interpretation of the relationship between features. Using the correlation as the similarity function, we can see correlated features as they form clusters in the *kRef* space. An example of visualising the feature space is presented in Figure 12, which shows the features of the *audiology* dataset from the UCI data repository. In this dataset, there are 69 attributes and the visualisation shows two obvious clusters, which suggests that there are two sets of correlated attributes in this dataset.

Other interesting future works include deeper investigation of theoretical issues about the reference point method, such as the optimal position for the reference points placement; or, applying this method as an efficient way of locating approximate nearest neighbours.

## 6 Conclusions

Visualisation is a useful tool for analysing difficult problems and in this work, we offered a new method to address the problem of visualising high-dimensional datasets. We compared our approach with other generic methods, namely PCA and random projection, and the results suggest that PCA is a computationally expensive approach, while random projection delivers inaccurate visualisations. Our method, on the other hand, is able to produce accurate visualisations in a cost-effective manner. Therefore, we recommend this method as an attractive visualisation tool to assist the data mining process.

## References

Achlioptas, D. (2001), 'Database-friendly Random Projections', *Symposium on Principles of Database Systems* .

Asuncion, A. and Newman, D. (2007), 'UCI machine learning repository', *University of California, Irvine, School of Information* .

Bingham, E. and Mannila, H. (2001), 'Random projection in dimensionality reduction: applications to image and text data', *conference on Knowledge discovery and data mining* .
**URL:** *http://portal.acm.org/citation.cfm?id=502546*

Blum, A. (2006), 'Random projection, margins, kernels, and feature-selection', *Lecture Notes in Computer Science* pp. 52–68.

Borg, I. and Groenen, P. J. (1997), *Modern multidimensional scaling : theory and applications*, Springer series in statistics, Springer.

Fawcett, T. (2004), ROC Graphs: Notes and Practical Considerations for Researchers, Technical report, HP Laboratories.

Fodor, I. (2002), 'A survey of dimension reduction techniques', *Manuscript* pp. 1–18.

Inselberg, A. (1996), Parallel Corordinates: A Guide for the Perplexed, *in* 'Hot Topics Proc. of IEEE Conference on Visualization', pp. 35–38.

Johnson, W. and Lindenstrauss, J. (1984), 'Extensions of Lipschitz mappings into a Hilbert space', *Contemporary mathematics* **26**, 189–206.

Jolliffe, I. T. (2002), *Principal Component Analysis 2nd Edition*, Springer Series in Statistics, Springer-Verlag.

Kohonen, T. (2000), *Self-Organizing Map*, Springer.

Lin, J. and Gunopulos, D. (2003), 'Dimensionality reduction by random projection and latent semantic indexing', *Proceedings of the Text Mining Workshop, at the 3rd* .

Moore, D. (2006), *Basic Practice of Statistics*, WH Freeman Company.

Morris, C. J., Ebert, D. S. and Rheingans, P. L. (2000), Experimental Analysis of the Effectiveness of Features in Chernoff Faces, *in* W. R. Oliver, ed., '28th AIPR Workshop: 3D Visualization for Data Exploration and Decision Making', Vol. 3905, SPIE, pp. 12–17.

Wilson, D. R. and Martinez, T. R. (1997), 'Improved Heterogeneous Distance Functions', *Journal of Artificial Intelligence Research* **6**, 1–34.

Witten, I. H. and Frank, E. (2005), *Data Mining: Practical Machine Learning tools and techniques, 2nd Edition*, Morgan Kaufmann, San Francisco.