Title
On distances between point patterns and their applications

Author
Lu, Weipeng

Publication Date
2013

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

On distances between point patterns and their applications

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Science in Statistics

by

Weipeng Lu

2013

ABSTRACT OF THE THESIS

On distances between point patterns and their applications

by

Weipeng Lu

Master of Science in Statistics

University of California, Los Angeles, 2013

Professor Frederic Paik Schoenberg, Chair

Nowadays distance measures techniques have been quickly developed and widely used to the real application. The main objective of this thesis is to make a brief introduction of different distance measures methods, especially spike-time distance and its application to point pattern prototypes and Multi-Dimensional Scaling (MDS) methods. Meantime, R programming packages for spike-time distance, prototype and Multi-Dimensional Scaling have also been introduced in order to make these methods more practical and convenient to the real world. And, their packages are used to a real dataset as an application.

The thesis of Weipeng Lu is approved.

Yingnian Wu

Hongquan Xu

Frederic Paik Schoenberg, Committee Chair

University of California, Los Angeles

2013

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLGEMENTS

I would first like to express my sincere gratitude to my advisor Professor Frederic Paik Schoenberg, who gives me such a good opportunity to do this thesis topic's research I am much interested in. And, it is his valuable guidance and advice that enable me to complete this thesis during my research process. I would also like to thank my committee members: Yingnian Wu and Hongquan Xu, who have provided me helpful teachings and instructive supports during my graduate study at UCLA. Meantime, I would like to thank Department of Statistics, UCLA for its good educational service and high quality teaching from all the staffs and teachers. At last, I would give my appreciation to my beloved parents for their support in finance, encouragement in spirit and instruction in academics so that I can successfully finish my graduate studies.

CHAPTER 1 Introduction

A point pattern is kind of a group that can describe the spatial locations of all the “activities” or “individuals” observed in a certain area. In this thesis, distance measure methods and their applications to point pattern prototypes and Multi-Dimensional Scaling (MDS) were introduced in order to have a better understanding of collections of point processes. Also, R programming packages for spike-time distance, prototype and Multi-Dimensional Scaling (MDS) has been created to make these developments more widely available to researchers. Furthermore, the packages will be described in details and applied to a real dataset.

1.1 Point process

Point processes are always used to describe data that are localized in space or time. In statistics and probability theory, a point process is a kind of random process for which any one realization consists of a set of isolated points either in time or space. For example, the occurrence of lightning strikes might be considered as a point process in both time and space if each point is recorded according to its location in time and space [1].

A point process is a random collection of points falling in some space. In most applications, each point represents the time and/or location of an event, such as a lightning strike or earthquake [4].

The relationship between time series and point processes is worth noting.

Many datasets that are traditionally viewed as realizations of (marked) point processes could in principle also be regarded as time series, and vice versa. For instance, a sequence of earthquake origin times is typically viewed as a temporal point process; however, one could also store such a sequence as a time series consisting of zeros and ones, with the ones corresponding to earthquake. The main difference is that for a point process, the points can occur at any times in a continuum, whereas the time intervals are discretized in the time series case [4].

1.2 Background for point pattern distance

One of the most influential papers in the area of “point pattern distances” was written by Victor and Purpura [3] to propose a variety of distance measures including spike-time distance. These methods were expanded upon and further developed in the application of earthquake aftershock sequences to include point pattern prototypes [4], [5], [6]. This developing field created a new application based on the pattern distances which is the point pattern prototype analysis. Given a collection of univariate observations, the mean and median can be defined as the minimizers of loss functions: the L_1 and L_2 norms, respectively. Similarly, a loss function may be constructed for collections of point patterns using distances, and the minimizer of this loss function, the point pattern prototype which describes the typical pattern in the collection [7].

Although spike-time distance has been studied in Victor and Purpura [3], Tranbarger [4], Schoenberg and Tranbarger [6], and Tranbarger and Schoenberg [5], efficient methods were never developed for patterns in multidimensional space. This limitation has not only restricted the direct application of spike-time distance but it has also restricted the application of prototype analysis to spatial, spatial-temporal, and marked point processes. At this time, David [7] developed techniques to apply these distance and prototype methods to any finite multidimensional space and also extend the theoretical foundation and applicable methods of prototype analysis [8].

1.3 Thesis structure and outline

This thesis is to outline several types of distances methods for point patterns and their applications to point pattern prototypes method as well as multidimentional scaling method. We also discuss some examples and applications of the proposed distance metrics to cluster analysis and prototype determination.

This thesis is outlined as follows. In chapter 2, we discuss spike-time distance which involves matching individual points of X to corresponding points in Y. And we describe distances that might be especially useful for clustered point processes, which is distanced based on classical function. Applications to summaries of collections of independent realizations of a point process through point pattern prototypes or multidimensional scaling are

given in chapter 3. R programming packages are introduced in details about how to calculate spike-time distance, point pattern prototype and multidimensional scaling and applied to a real dataset in chapter 4. In chapter 5, we make a conclusion on this thesis and discuss some future work.

In this thesis, such a collection of point patterns is referred to a point process dataset.

CHAPTER 2 Distance Metrics for point patterns

Distance is a measure of proximity between any two locations or spatial points. Distance metrics have been used for a wide range of applications in statistics. Nowadays, distance metrics is playing an important and irreplaceable role in the field of computer vision, clustering analysis and multidimensional scaling, etc. We can use distances to compare the distribution of observed data to a particular distribution and to compare two different observed distributions as well. It is obvious to know that how much is the value of distance metrics [4].

Then distance metrics started to be used in the analysis of point processes when Victor and Purpura [3] in the paper “Metric-space analysis of spike trains: theory, algorithms and application” described neuronal spike trains analysis successfully by proposing three distance metrics including the spike-time distance. It also showed how multidimensional scaling can be used to assess the similarity of several distance metrics to Euclidean distances. Even though the aim of Victor’s paper is to improve distance metrics for use in neurology, these distance metrics can also be applied to point process data in the fields of seismology, economic, astronomy, forestry and epidemiology [4].

This chapter aims to solve the problem about how to define as well as to compute the distance between two observed point patterns. This chapter

provides a detailed introduction for several types of distance metrics for point patterns.

2.1 Spike-time distance

In the year of 1997, Victor and Purpura [3] published a paper to introduce the idea for using the spike-time distance to determine how two neuron spike-trains differ. The idea of measuring the difference between two point patterns is to transform one point pattern into the other one. The spike-time distance is a very popular method to be used in computer imaging or related areas. It has been applied to neuronal, earthquake, and wildfire data [13]. And the spike-time distance is a very simply method because this method focuses on transforming the individual points of the two processes instead of their summary histograms [4].

The main idea of how the spike-time distance works is that the addition of points on either pattern, the deletion of points from either patter, or the movement of a pattern to a different location. Each step carries a given cost. It is defined as the minimum sum of costs for transforming one of the two point patterns into the other by deletion, addition and moving operations (one point can be moved at a time). That is to say, transform point pattern X into point pattern Y is the spike-time distance between X and Y [4].

In general, the cost of adding one point is p_a, the cost of deleting one point is p_d and the cost of moving a point is p_m. In order to be a true distance

metric, it must be symmetric thus the constraint $p_a = p_d$ must be imposed by default. However, in some application areas, it might be desirable for p_a and p_d to take on different values. In this thesis, we focus on when the spike-time distance is a symmetric distance metric. Therefore, $p_a = p_d$ is by default [4].

Consider $X = \{x_1, x_2, ..., x_q\}$ and $Y = \{y_1, y_2, ..., y_r\}$ to be two fixed and finite point patterns in a finite-dimensional space. Then spike time distance, $d(X, Y)$, is defined as the lowest cost transformation of X to Y. Points in X may be moved or deleted, and other points may be added until the pattern is the same as Y. The cost of adding a point to X is p_a, and the cost of deleting a point in X is p_d, where p_a and p_d are parameters to be specified. Moving the location of a point in X by a translation vector v incurs a cost $p_m \|v\|$, where p_m is a parameter and $\|v\|$ is a measure of the size of the translation; for instance $\|.\|$ might denote the Euclidean norm or more typically the rectilinear norm (Manhattan distance). Later, we will discuss generalizations of spike-time distance through adjustments in the translational measure, $\|.\|$ [7].

An illustration of the spike-time distance between two sample patterns is shown in Figure 2.1. In this particular transformation from X to Y, three points in X are moved, one point from X is deleted, and one point is added to X [7].

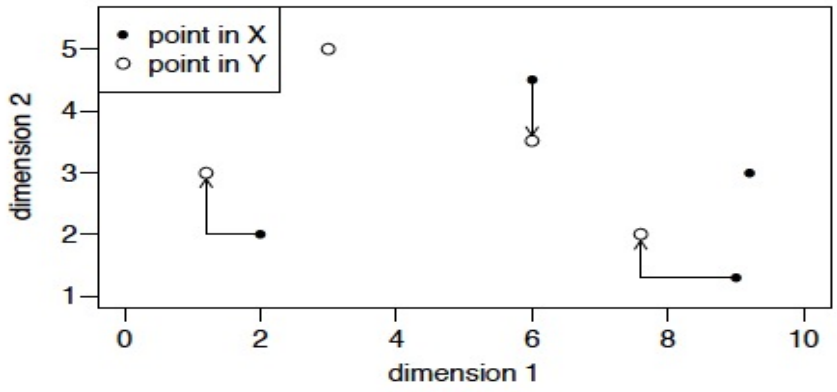

Figure 2.1: A sample transformation of *X* into *Y*, for point processes in a rectangular space. Three points in *X* are moved to points in *Y*, as shown by arrows. The remaining point in *X* that is not moved would be deleted, and the point in *Y* that does not have a point moved to it would be added [7]

Victor and Purpura [3] showed that *d* is a well-defined symmetric distance metric provided that $p_a = p_d$; the case where $p_a \neq p_d$ is not discussed in this thesis.

The computation of spike-time distance involves the notion of a transformation $T_{X,Y}$ from the point pattern *X* into the point pattern *Y*, which may be formalized as follows. Suppose $X = \{x_1, ..., x_m\}$ and $Y = \{y_1, ..., y_n\}$. A transformation $T_{X,Y}$ may be viewed as a mapping from *X* to $Y \cup \phi$, where $T_{X,Y}(x) = y$ means the point *x* is moved to the point *y*; if the point x is deleted under the transformation $T_{X,Y}$, then we write $T_{X,Y}(x) = \phi$. Similarly let $T_{X,Y}^{-1}$ denote the inverse mapping, from *Y* to $X \cup \phi$, so that we may write $T_{X,Y}^{-1}(y) = x$ if the point x is moved to the point y, and $T_{X,Y}^{-1}(y) = \phi$ if the point *y* is added to *X* in the transformation $T_{X,Y}$. Let $T_{X,Y}(X)$ denote the collection of points of *Y* to which points of *v* are moved, rather than added, and let $T_{X,Y}^{-1}(Y)$ denote the

set of points of X are moved, rather than deleted. The definition of spike-time distance as given by Victor and Purpura [3] prescribes that each point of X is moved to a unique point of Y; hence $|T_{X,Y}(X)| = |T_{X,Y}^{-1}(Y)|$, where $|A|$ denotes the cardinality of the set A. For convenience in considering transformations from X to Y where X and Y are fixed, we will abbreviate $T_{X,Y}$ by simply T in what follows [7].

Given two finite point patterns X and Y, the cost associated with a transformation T from X to Y [7] is

$$\cos t(T) = pd(|X| - |T(X)|) + pa(|Y| - |T(X)|) + pm \sum_{x \in T^{-1}(Y)} \|x = T(x)\|$$

The spike-time distance between X and Y is thus

$$d(X,Y) = \inf_T \cos t(T)$$

2.2 Nearest point distance

Another useful distance function that is also defined in terms of such transformations of X into Y is the nearest-point distance, where each point x in X is simply moved to its nearest neighbor in Y. For convenience, call this point y_x. Unlike spike-time distance, nearest-point distances are computed without allowing the addition or deletion of points. The nearest-point distance is defined simply as

$$dn(X,Y) = \sum_{x \in X} \|x - y_x\|$$

where $\|.\|$ may represent Euclidean distance for point patterns in R^k, or some other distance metric for point patterns in a more general metric space. An example of nearest-point distance is shown in Figure 2.2 in the right panel. Some points in Y have several associated points in X while some have no associated points [7].

Note that for two distinct points x, x' in X, we may have $y_x \equiv y_{x'}$, and that in general, $d_n(X,Y) \neq d_n(Y,X)$, so that d_n is not formally a distance metric. However, one may also consider the sum of these two distances as a symmetric nearest-point distance metric nearest-point distance metric:

$$d_N(X,Y) = d_n(X,Y) + d_n(Y,X)$$

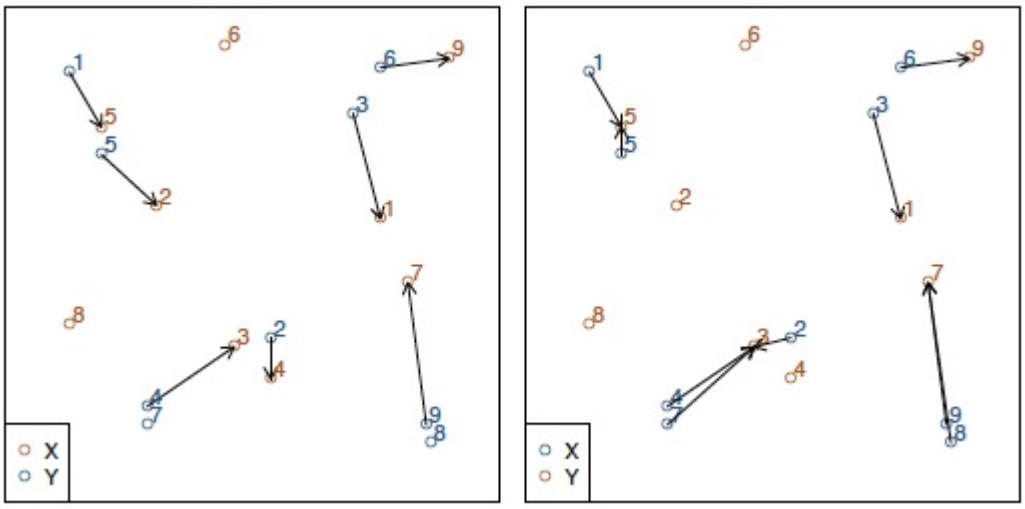

Figure 2.2: Spike-time distance (left) and nearest-point distance for moving Y to X (right)[7]

2.3 Cluster distances

Distance functions appropriate for clustered patterns would also be very useful. This separate family of distances can be characterized by its incorporation of movements of collections of points [7].

2.3.1 Cluster distance

The following metric may be called cluster distance. Let T represent a transformation of X into Y, sequentially moving collections of points in X. Thus, T is itself a sequence of transformations, t_i, where each t_i moves a subset X_i of points by a vector z_i (see Figure 2.3). Then the cost of T may be defined as [7]

$$p_d \left| X_{delete} \right| + p_a \left| Y_{add} \right| + \sum_i p_m \left| X_i \right|^q \left\| z_i \right\|$$

where q is a parameter in $[0,1)$. The cluster distance, $d_c(X,Y)$, is the infimum cost over all such transformations.

It is unclear about how we might compute this cluster distance. The number of possible transformations is much larger than that of spike-time distance, and research into methods to identify optimal transformations is ongoing. However, a visualization of the first steps of such a transformation is shown in Figure 2.3 [7].

2.3.2 Declustered spike-time distance

One may instead define a distance for clustered point processes via first aligning clusters in X with clusters in Y and subsequently applying spike-time distance to

the result. For instance, let $\{R_j\}$ represent a set of disjoint and concave regions of the space that contain all the points of X. One may translate all of the points in a given region by some fixed vector and repeat for each region, assigning a cost of p_c per unit distance to each translated region, and then spike-time distance may be applied after these regions are moved. An illustration is given in Figure 2.3. This declustered spike-time distance is defined as the minimum cost over all such transformations and choices of $\{R_j\}$ [7].

The identification of the optimal choices of $R = \{R_j\}$ is not straightforward and no optimal solution is currently known. Ongoing work focuses on how we might identify these regions through an iterative process. For example, we might initialize such an algorithm by first setting $R_0 = \phi$. Then, if we compute spike-time distance and observe that some clusters are moved along the same direction and at approximately the same distance, we might propose a region R_1 that captures those points and moves them. Thus, we revise R_0 to become $R_1 = \{R_1\}$, where region R_1 is moved some optimal distance and direction based on the spike-time distance movements. Now iterating with R_1 and spike-time distance, we may identify a second cluster that should be moved, and so on [7].

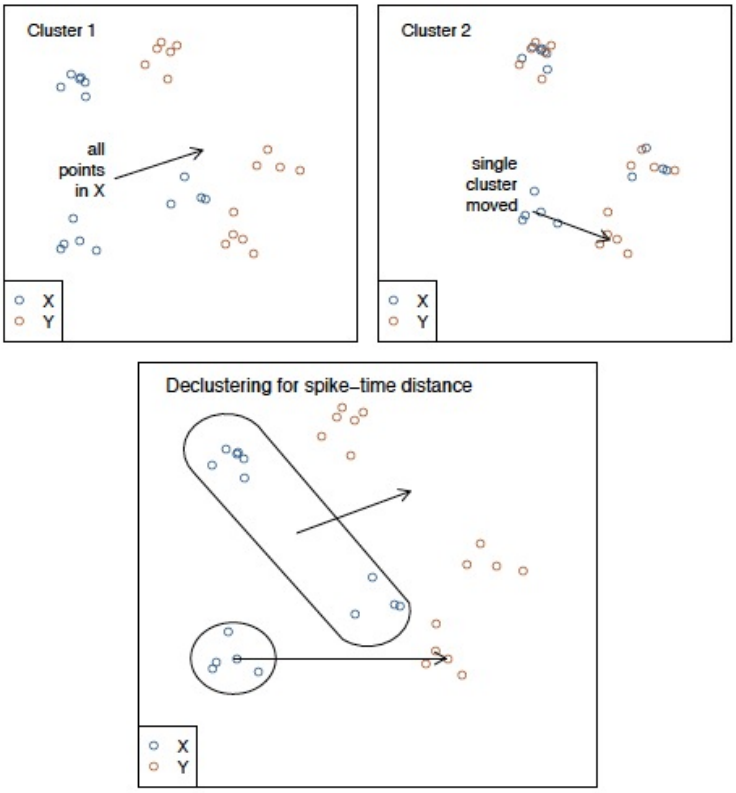

Figure 2.3: The first two steps in the cluster distance as shown in the top panels. The bottom panel represents declustering the point patterns [7]

2.4 Distances based on functional summaries

2.4.1 Model-based distances

Given models for the point processes giving rise to the point patterns X and Y, one may define the distance between X and Y in terms of the differences between characteristics of these models. For instance, if the point processes X and Y are characterized by their conditional intensities $\lambda_X(x)$ and $\lambda_Y(x)$, respectively, then one measure of the difference in these point process models is the integral of the squared difference of the two conditional intensities over the

observation region S:

$$\int_S (\lambda_X(x) - \lambda_Y(x))^2 dx$$

This is illustrated in the right panel of Figure 2.4, where the intensity functions have been estimated based on the data shown at the top of the panel. These methods readily extend to a variety of other summaries, such as the integrated squared difference between the overall mean, or second moment measure, or higher moments or cumulants of the processes. Of course, the conditional intensities may be replaced by their expected values, their overall intensities, or in the spatial point process setting, by the Papangelou intensities [1], [7].

2.4.2 Distances based on classical functional statistical summaries

Differences in point process behavior can also be characterized by comparing classical summary measures of the first or second moments for each of the point patterns. For instance, given two point patterns X and Y, one could imagine looking at an estimate of the intensity of X, e.g. a kernel estimate, and a similar estimate of the intensity of Y. Alternatively, if each of the point patterns is one-dimensional, then one could look at the empirical cumulative distribution function for each point pattern as a statistical summary of the realization. A further alternative would be to take the estimated K-function or its derivative, the estimated reduced 2nd moment measure, for each pattern, and examine the difference.

As an illustration, one could compare the integrated squared difference between estimates of Ripley's *K*-function [9] for two point patterns [10]:

$$\int (K_1(r) - K_2(r))^2 dr$$

This distance will be illustrated in the left panel of Figure 2.4.

First and second order measures represent a valuable description of a spatial point process but do not generally uniquely characterize a point process. Additional important summary descriptions are given by distance methods based on measures of some distances between points. The nearest neighbor distance *D* may be defined as the distance from a point of the pattern to the closest of the other points in the same pattern. The empty space distance *F* is the distance from an arbitrary fixed location to the nearest point of the pattern. For a given point pattern, a set of distances (nearest neighbor or empty space distance) can be summarized by means of their corresponding distribution functions. Let $G(r)$ denote the nearest-neighbor distribution function, and $F(r)$ denote the corresponding first contact distribution function or empty space function. These distribution functions may be estimated from the observed point patterns *X* and *Y* using conventional methods [7].

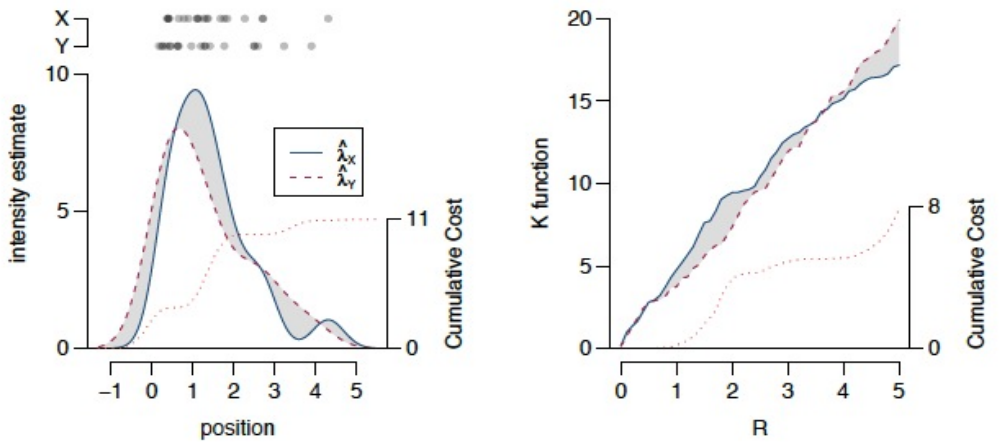

Figure 2.4: Left: A comparison of two intensity models. Right: an example of using Ripley's *K* function.

In practice, each point pattern is typically observed in a bounded region *S*, and without precautions, these boundaries can lead to biased estimates. This problem is known in the context of spatial statistics as edge-effects. Different edge-corrected estimators of the functions *F* and *G* have been proposed. There is a clear analogy with censoring in the context of survival analysis Baddeley and Gill (1997). The different distances observed within a single point pattern are really censored distances. Let d_i denote the observed distance from the i^{th} point of the point pattern to its nearest neighbor within the window *W* [7]. If $c_i < d_i$ the real nearest neighbor could be outside the window and, in this case, the real and unknown nearest neighbor distance d_i fulfills $d_i > c_i$, and the observation is censored. A similar comment applies to empty space distances. In the both cases the censoring distance is the distance from the sampling point to the frame window. Given a statistical summary such as the estimated *F* or

G-function of the point processes, the dissimilarity between point processes *X* and *Y* can be defined as the distance between the corresponding estimated functional summaries. For instance, let F_X and F_Y be the corresponding estimated empty-space distribution functions for *X* and *Y*. The dissimilarity between *X* and *Y* can thus be defined as:

$$D(X,Y) = d(\hat{F}_X, \hat{F}_Y)$$

where *d* stands for a metric between the functions. For instance, one may use the L_2 metric:

$$D_F^2(X,Y) = \int_0^{t_0} (\hat{F}_X(r) - \hat{F}_Y(r))^2 \, dr$$

Or the L_∞ metric:

$$D_F^\infty(X,Y) = \sup_t \left\| \hat{F}_X(r) - \hat{F}_Y(r) \right\|$$

By replacing the empty-space distribution function with the nearest-neighbor distribution function *G* or *K*-function *K*, one can similarly define $D_G^2(X,Y)$, $D_G^\infty(X,Y)$, $D_K^2(X,Y)$ and $D_K^\infty(X,Y)$, respectively. Note that sampling variability in estimates of *K*(*h*) tend to increase with *h* and this can have a great influence on the value of the dissimilarity measure [7].

2.4.3 Distances based on LISA functions

An alternative class of functions one may use to characterize each process *X* and *Y* is based on LISA functions. Given point patterns $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$, one may calculate the set of LISA functions for each pattern

corresponding to any individual or spatial location. Denote both sets by $\{l_i^X(r), i=1,...,n\}$ and $\{l_j^Y(r), j=1,...,m\}$. Then one may derive several possibilities for distances between *X* and *Y*, such as [7]:

(a) Define:

$$A_L^X = 1/(n\times(n-1))\sum_i\sum_j d(l_i^X(r), l_j^X(r))$$
$$= 1/(n\times(n-1))\sum_i\sum_j \int (l_i^X(r) - l_j^X(r))^2 dr$$

as the average of all pairwise distances between LISA functions of points in *X*. Define the same average for points in *Y*, say, A_L^Y. Then, a similarity or dissimilarity measure is given by

$$d_L(X,Y) = A_L^X - A_L^Y$$

(b) Define the average LISA function coming from the set of LISA functions for points in *X* and *Y*. Denote them by $l^X(r)$ and $l^Y(r)$. Note that $l^X(r)$ is a function itself. Then one may define a distance measure via [7]:

$$d_l(X,Y) = d(l^X(r), l^Y(r)) = \int (l^X(r), l^Y(r))^2 dr$$

2.4.4 The proximity function of an individual to a population based on LISA distances

Let $d_X(i,j) = d(l_i^X(r), l_j^X(r)), i, j = 1,...,n$ the set of pairwise distances between LISA functions for points in *X*. We define the geometric variability for pattern *X* as [7]:

$$V_{d_X} = \frac{1}{2n^2}\sum_{i,j} d_X^2(i,j)$$

Then the proximity function for the i^{th} point in X $i = 1,...,n$ is given by:

$$\Phi_{X_d}^2(i) = \frac{1}{n}\sum_{j=1}^{n} d_X^2(i,j) - V_{d_X}$$

Hence the value of the associated proximity-based density function for the i^{th} point in X, i=1,2,…,n is given by

$$f_{X_d}(i) = \exp\{-\frac{1}{2}\Phi_{X_d}^2(i)\}$$

One may proceed similarly with pattern Y defining V_{d_Y}, $\Phi_{Y_d}^2(j)$ and $f_{Y_d}(j)$, for j=1,2,…,n.

Finally, the distance between patterns X and Y may be defined as a measure of similarity/dissimilarity between the densities f_{X_d} and f_{Y_d}, as [7]

$$d_P(X,Y) = d(f_{X_d}, f_{Y_d})$$

CHAPTER 3 Applying distance measures to a Collection of patterns

The application of distance measures is in the construction of a point pattern prototype and multidimensional scaling. This chapter will introduce these two applications.

3.1 Point pattern prototype

Point pattern prototype [7] is a characterization of the prototypical pattern of a collection. It is useful for identifying the typical pattern in a collection. The prototype has generally been used to construct robust representations of a collection of patterns using spike-time distance. It has been widely used in many scientific areas.

We let C represent a collection of finite point patterns as $\{X_1,...,X_t\}$. Then we define the prototype $P_d(C)$ of the collection C based on the distance measure d through a loss function:

$$\sum_{X_i \in C} \alpha_i d(P, X_i)$$

Here α_i represents a weight corresponding to point pattern X_i. The weights α_i are typically chosen to be unity unless certain point patterns are deemed more important than others, or if the point patterns are measured with differential error.

Tranbarger and Schoenberg created a method for estimating the prototype in one dimension. This method was extended to multiple dimensions [7].

3.2 Multidimensional scaling

Multidimensional scaling [7] is often used to explore similarities or dissimilarities in data. It is the most common technique used in perceptual mapping.

Multidimensional scaling is useful for pattern classification based on a distance metric. It may be applied to any of the distance metrics introduced. We let C represent a collection of finite point patterns as $\{X_1,...,X_t\}$. And a distance matrix D is computed where $D_{i,j} = d(X_i, X_j)$ for some distance measure d. Multidimensional scaling uses this distance matrix to estimate relative locations of the pattern. Each pattern is itself represented by a point, and multidimensional scaling embeds these points. The goal of this embedding is to place points representing similar or dissimilar patterns close to or far from each other. In order to achieve this aim, multidimensional scaling selects a location for each pattern such that the resulting Euclidean distances between the patterns, described by $\widetilde{D}$, approximates the patter distances D by minimizing the following loss function [11] :

$$L(D,\widetilde{D}) = \sum_{i \neq j} [D_{ij} - \widetilde{D}_{ij}]^2$$

Multidimensional scaling can be useful in both identifying groups of patterns and in classification. If classification are provided on a training set, then a distance metric may be applied to compute the distance matrix D. Applying multidimensional scaling groups the patterns into a new space, where traditional classification methods may be applied.

CHAPTER 4 Spike-time distance, prototype and Multi-Dimensional Scaling in R

In this chapter, spike-time distance, point pattern prototype and Multi-dimensional scaling will be reached by R programming using “ppMeasures” [12] package and “smacof” [7] package, in which prototype methods are implemented using the “ppPrototype” function in the “ppMeasures” package on CRAN and Multi-dimensional scaling methods are implemented using the “smacofSym” function in the “smacof” package. At the end, these R packages are applied to a real dataset.

4.1 Preparation

We need to do some preparation jobs [14] before we start to calculate these methods through R programming [15].

R can be freely available in its official website. After downloading R program, “ppMeasures”, “smacof” and “spatstat” packages are needed to download for this chapter on CRAN.

In order to install the packages, use the “install.packages” function. And the “library” function will be used to load the packages to your computer.

```
> library(ppMeasures)
> library(smacof)
> library(spatstat)
```

4.2 Application for spike-time distance, prototype and MDS

“ppMeasures” package was created for spike-time distance and prototype estimation algorithms [7]. The three main functions are as follows:

“stDist” function was used to compute spike-time distance and its variants.

“ppColl” function was used to prepare a point pattern collection data set.

“ppPrototype” function was used to estimate a the prototype of a point pattern collection.

Computing spike-time distance requires two patterns and penalty parameters. Within the “ppMeasures” package, patterns are represented by matrices where rows represent points and columns represent dimensions. For example, two 2-dimensional patterns are created below:

```
> x <- matrix(rchisq(20,3),nrow=10)
> y <- matrix(rchisq(30,4),nrow=15)
> library(ppMeasures)
```

Here, *x* contains 10 points with two dimensions and *y* contains 15 points.

Then the stDist function may be applied to compute the distance from *x* to *y*:

```
> d1 <- stDist(x,y,pm=0.5)
> d1
[1] 20.17389
```

By default, the addition and deletion penalties are set to 1; they may be adjusted using the pa and pd arguments.

Also the moving penalty could be specified as a vector. For example,

```
> d2 <- stDist(x,y,pm=c(0.5,0.2))
> d2
[1] 18.34532
```

“print”, “summary” and “plot” can still be applied to this method.

According to R graphical manual, we can have this example:

```
> data(collEx2)
```

Here, collEx2 is a collection of two dimensional patterns.

In order to construct the collection of this dataset, we used “ppc” function.

```
> ppc      <- ppColl(collEx2[,2:3], collEx2[,1])
```

Then we used the “ppPrototype” function to compute the prototype. By default, this method is through “margPT”.

```
> protoMP <− ppPrototype(ppc, 0.1)
> points(protoMP, pch=20, cex=3.5, col='#FF000088')
```

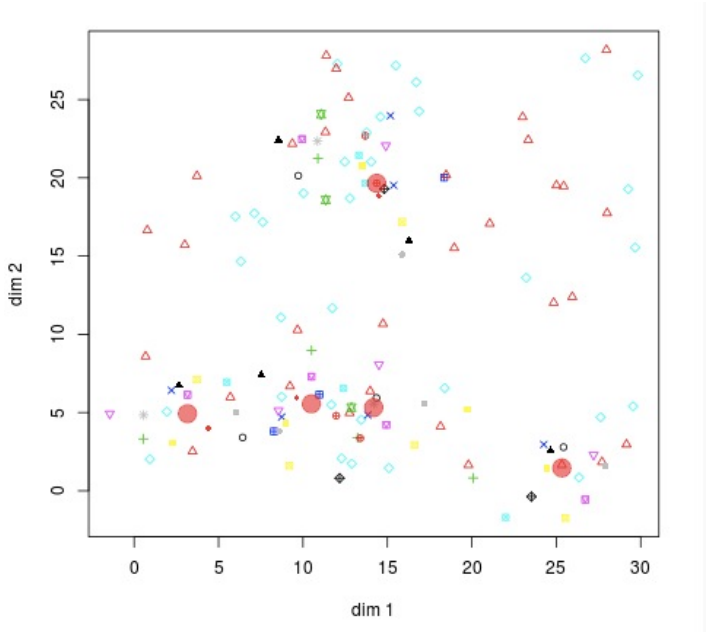

Figure 4.2: output for the dataset

For the “smacofSym” function, a symmetric matrix representing the distance between the objects in the collection is required. The distances are computed by spike-time distance using a loop structure. More information can be get from “Multidimensional scaling using majorization: SMACOF in R” written by de Leeuw, J. and Mair, P [11].

4.3 Real dataset application

The data were collected in Cataby, in the Mediterranean type shrub- and heathland of the south western area of Western Australia, where the locations of 6378 plants from 67 species on a 22 m by 22 m plot have been recorded [16]. This dataset which was used in this application was formed by the 2 most abundant species of seeders and the 10 most dominant (influential) species of resprouters and it was projected to unit square and the variable *t*

indicating the specie, note that t = 2 and 8 correspond to seeders and other ones correspond to resprouters. In this section, we aim to use spike-time distance to approach point pattern prototype and multidimensional scaling methods with real dataset for an application.

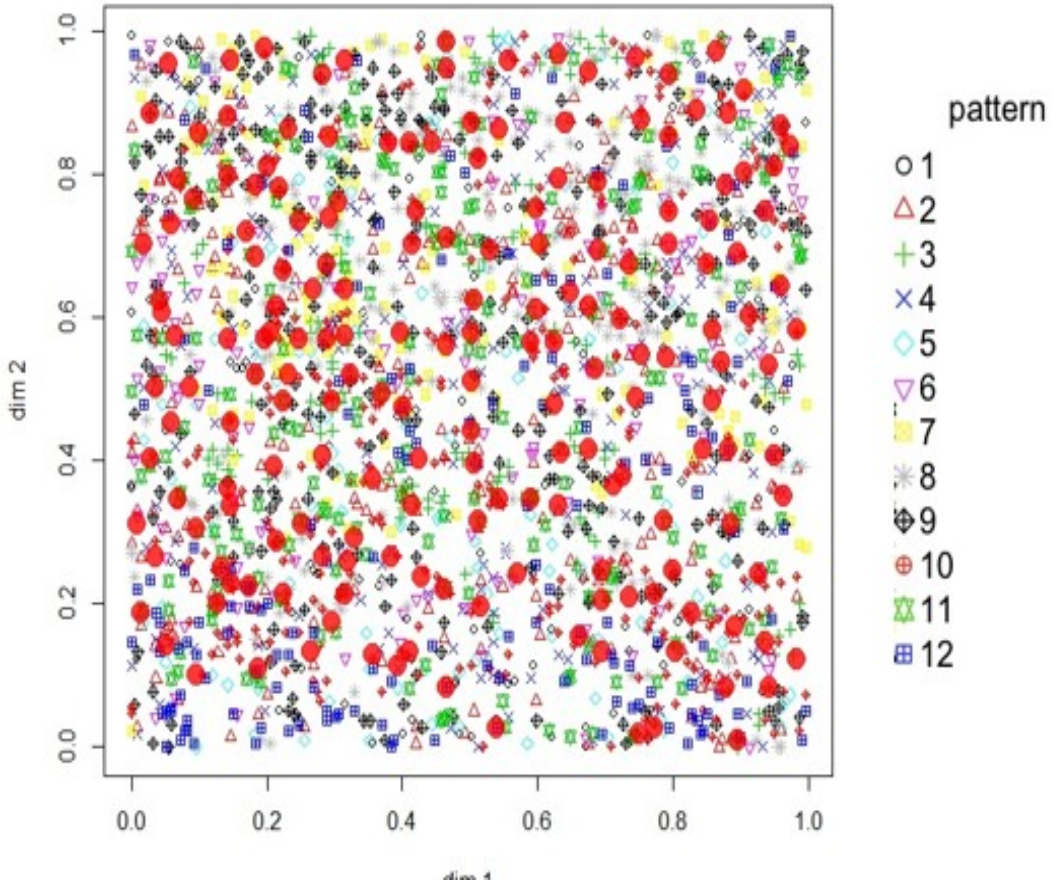

Figure 4.3.1: Prototype for 12 patterns (1)

In Figure 4.3.1 there are 12 species (patterns) drawing with different shape and colors listed on the right, in which the red dot is the prototype.

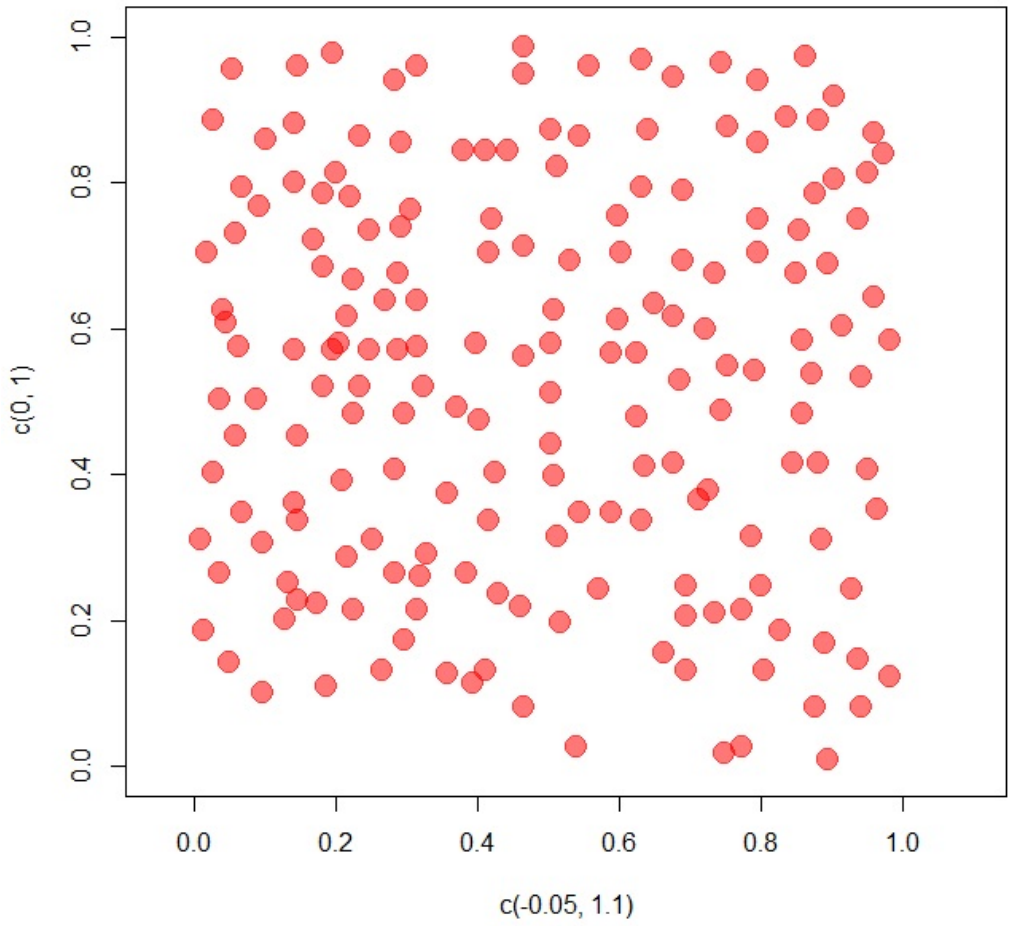

Figure 4.3.2: prototype for 12 patterns (2)

In order to see the prototype clearly, we draw the Figure 4.3.2. This plot is only prototype for 12 patterns. From the plot, it is obvious to know that they are almost very uniform and only a few of them are overlapped which is to say they are interactive.

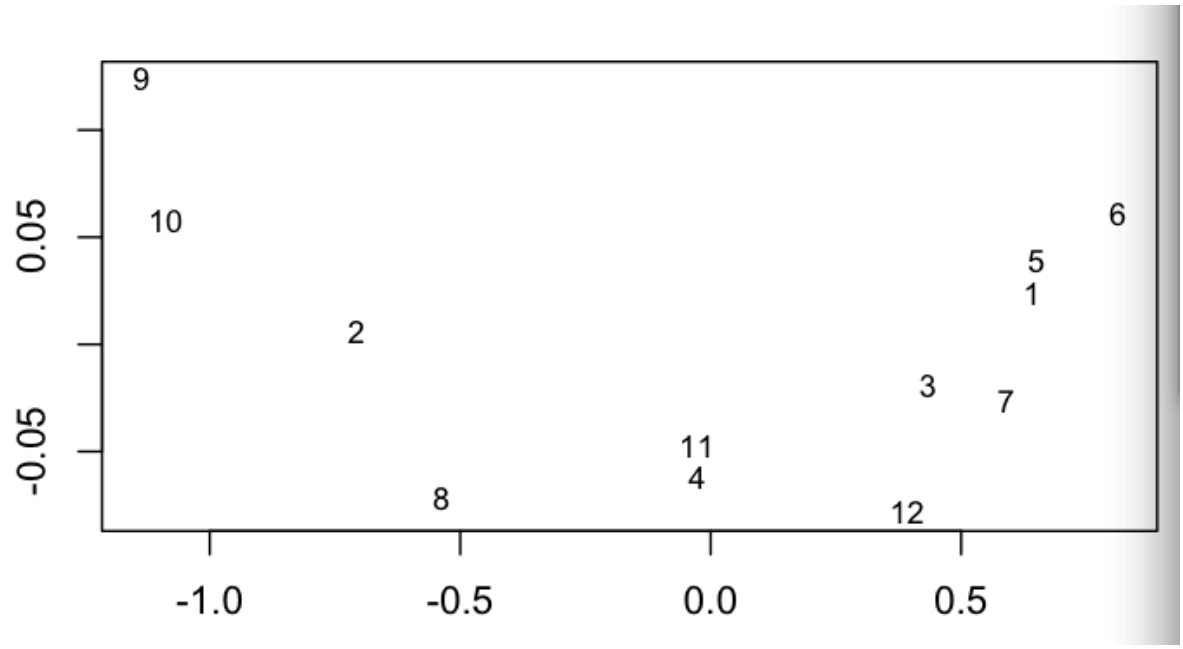

Figure 4.3.3: multidimensional scaling result

We use the “smacofSym” function in “smacof” package to draw the MDS plot for these 12 species (patterns). In this plot, resprouter 4 and resprouters 11 are close to each other as well as resprouters 1 with resprouters 5. There might be some correlations between them. And there could be some clusters among these 12 patterns.

CHAPTER 5 Discussion

5.1 Conclusion

In this thesis, we outlined several types of distance methods. And we introduced two applications based on spike-time distance which are point pattern prototype and multidimensional scaling. In the end, we introduced R package to apply these methods to real world.

5.2 Future work

The development of these methods of distances between point patterns is very important. In the meantime, the development of R software package corresponding to theses methods is also very important. However, the speed of computing is very slow when using the package in R to compute the spike-time distance and point pattern prototype introduced in the thesis. Therefore, the further improvement on the R packages for spike-time distance, prototype and Multi-Dimensional Scaling (MDS) might be done in the near future in order to fulfill the application of the methods well.

References

[1] Daley, D.J., Vere-Jones, D. *An Introduction to the Theory of Point Processes, Volume 1: Elementary Theory and Methods*. New York: Springer-Verlag, 2003

[2] Schoenberg, F. P. Introduction to point processes. *Wiley Encyclopedia of Operations Research and Management Science*. Wiley, 2011: 616-617.

[3] Victor, J., Purpura, K. Metric-space analysis of spike trains: theory, algorithms and application. Journal of Neuroscience Methods, 1997, 8:127-164.

[4] Tranbarger, K.E. Point Process Prototypes, and Other Applications of Point Pattern Distance Metrics. Ph.D. thesis, University of California, Los Angeles, 2005.

[5] Tranbarger, K.E., Schoenberg, F.P. On the computation and application of point process prototypes. Open Applied Informatics Journal, 2010, 4:1-9.

[6] Schoenberg, F. P., Tranbarger, K. E. Description of earthquake aftershock sequences using prototype point processes. Environmetrics, 2008, 19: 271-286.

[7] Diez, D. M. Extensions of distance and prototype methods for point patterns. Ph.D. thesis, University of California, Los Angeles, 2010

[8] Diez, D. M., Schoenberg, F. P., Woody, C. D. *Analysis of neuronal responses to stimuli in cats using point process prototypes*. 2010

[9] Ripley, B. D. Modeling spatial patterns (with discussion). Journal of the Royal Statistical Society. Series B, 1977, 39(2): 172-212.

[10] Besag, J. Contribution to the discussion of Dr Ripley's paper. Journal of the Royal Statistical Society B, 1977, 39(2): 193-195.

[11] de Leeuw, J., Mair, P. Multidimensional scaling using majorization: SMACOF in R. Journal of Statistical Software, 2009, 31(3):1-30.

[12] D. M. Diez, K. E. Tranbarger Freier, F. P. Schoenberg. ppMeasures: Point pattern distances and prototypes. R package version 0.1, 2010B.

[13] K. Nichols, F. P. Schoenberg, J. Keeley, D. M. Diez. The application of prototype point processes for the summary and description of California wildfires, 2010.

[14]R Development Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2010. ISBN 3-900051-07-0.

[15] P. Dalgaard Introductory statistics with R (2nd ed.). New York: Springer. 2008

[16] J.B. Illian, J. Møller, R.P, Waagepetersen. Hierarchical spatial point process analysis for a plant community with high biodiversity. Environm. Ecol. Statist. 2009