Clustering Automatico di Stime di Parametri di un Canale MIMO Usando MATLAB

Pierluigi Cera

10th January 2006

Abstract

This thesis addresses the problem of identifying clusters from an input data set consisting of several MIMO channel parameter estimates, achieved by a channel parameter estimator, such as SAGE. In many previous work visual inspection has been used for the cluster identification. Nevertheless this approach becomes impractical when one has to analyze large amounts of measurement data. Furthermore visual inspection is limited by human senses, since one is hardly able to identify clusters in more than three dimensions.

In this work we introduced a novel scalable framework to automatically identify multi-path clusters, which requires only a very limited user interaction. The proposed framework is novel in five respects: (i) the framework algorithm enables to cluster MIMO channel parameter estimates automatically with a minimum of user input; (ii) as a distance metric we use the multi-path component distance (MCD), which makes the framework able to scale the data, solve the problem of the angular periodicity and hence enable for joint multidimensional clustering; (iii) by including power and the MCD into the clustering algorithm, we make it applicable to cluster identification in propagation research; (iv) the cluster validation package provides a trustworthy estimate of the correct number of clusters; (v) the cluster pruning package does not change the cluster behavior significantly, but improves visibility and future tracking performance. Furthermore, the clustering algorithm introduces a convincing, inherent definition of a cluster itself.

We assess the performance of the framework by clustering both synthetic scenarios and real-world MIMO measurement data from an indoor big hall environment.

Preface

This master thesis is a result of a 7 months (07.03.2005 - 04.10.2005) exchange experience at the Institut für Nachrichtentechnik und Hochfrequenztechnik of the Technische Universität Wien (Vienna, Austria), under the enlightening supervision of Nicolai Czink, Jari Salo and Prof. Ernst Bonek, and in collaboration with Elektrobit Testing Ltd. (Oulu, Finland).
The goal of the project has been to develop an algorithm for the automatic identification of clusters from data sets including several parameter estimates of a MIMO channel, achieved by both synthetic channels and former measurement campaigns.
The final result is a MATLAB framework including: (i) a new clustering algorithm, (ii) a combined cluster validity metric and (iii) an improved cluster shape pruning algorithm.
Part of this work was supported by the European Union founded Network-of-Excellence (NEWCOM).

Questa tesi é il risultato di una esperienza di scambio, durata 7 mesi (dal 07.03.2005 al 04.10.2005), presso l'Institut für Nachrichtentechnik und Hochfrequenztechnik della Technische Universität Wien (Vienna, Austria), coordinato dall'illuminante supervisione di Nicolai Czink, Jari Salo e del Prof. Ernst Bonek, ed in collaborazione con Elektrobit Testing Ltd. (Oulu, Finlandia).
Lo scopo del progetto è stato lo sviluppo di un algoritmo per l'identificazione automatica di cluster a partire da set di dati che comprendono diverse stime di parametri di un canale MIMO, ottenuti sia mediante canali sintetici, sia attraverso campagne di misura precedentemente condotte.
Il risultato finale é un framework, realizzato in MATLAB, formato da: (i) un nuovo algoritmo di clustering, (ii) una metrica di validazione combinata e (iii) un algoritmo perfezionato di cluster shape pruning.
Questo progetto é stato in parte sostenuto dalla European Union founded Network-of-Excellence (NEWCOM).

Acknowledgments

First of all I would like to thank who has followed me during my “adventure” in Wien. Thanks to Niki for his careful supervision, for making me feel to belong to a group of research and for giving me the opportunity to travel a lot. Thanks to Jari for supporting me with clustering and MATLAB and for bearing all of my buggy code. A special thank to Prof. Bonek for his enlightening advices and for taking care of me at the TU-Wien. He is one of the most interesting person I have ever met in my life. I also want to thank all the people I have met at the Institute.

A sincere thanks to Prof. Falciasecca for giving me the opportunity to go to Vienna.

Finally I have to thank all those who have been close to me during my studying. Thanks to my university fellow in Bologna: Lara, Raffaele, Mirko, Mattia, Paola, Giovanna, Pela and all those who shared with me the joys and the pains of university. A special thank to Gianluca for a 24 years long support.
I also have to thank all the Panorama gang. I will never forget of David, Evelina, Renato, Cecilia, Claudio, Maarten, Danilo, Manu, Daniel, Katerina, Gabriel, Masha, Ginevra, Luis, Maria, Matteo (both!), Joana, Kropek, Rado, Stephansplatz, Stefano, Gianluca, Petroula, Muzy and, of course, Gosia, with who i shared this extraordinary experience in Wien.

In the end I want to thank my parents. Thank you for making me realize one of my dreams.

Published material

- N. Czink, P. Cera, J. Salo, E. Bonek, J. Nuutinen and J.-P. Ylitalo, *Improving Clustering Performance by Using the Multi-Path Component Distance*, to appear in IEEE Electronics Letters
- N. Czink and P. Cera, *A Novel Framework for Clustering Parametric MIMO Channel Data Including MPC Powers*, presented in COST 273, Lisbon, Portugal, November 2005
- N. Czink, P. Cera, J. Salo, E. Bonek, J. Nuutinen and J.-P. Ylitalo, *Automatic Clustering of MIMO Channel Parameters using the Multi-Path Component Distance Measure*, presented in WPMC, Aalborg, Denmark, September 2005

Contents

List of Figures

Sommario

Chapter 1

Introduction

1.1 Overview of MIMO systems

Communication systems that are equipped with more than one antenna at both the transmitter and receiver side are called multiple-input multiple-output (MIMO) systems. In contrast to conventional communication systems that use only a single receive and transmit antenna, the so-called single-input single-output (SISO) systems, MIMO systems allow to use the same frequency band at the same time to transmit different data streams by exploiting the spatial structure of the mobile radio channel. Actually, it is possible to construct orthogonal spatial channels that allow to transmit several data streams without any mutual interference between them [1].

The underlying physical phenomenon is the radio wave propagation between the transmit and the receive antenna arrays. When modeling the wave propagation by rays (i.e. in a cluttered indoor environment), there typically exist several distinct paths along which radio waves can propagate from the transmit to the receiver side via different region in space [1]. These paths are usually referred to as multi-path components (MPCs), and each one is characterized by several propagation parameters which are typically the azimuth and elevation of arrival, respectively φ_{AoA} and θ_{AoA}, the azimuth and elevation of departure, respectively φ_{AoD} and θ_{AoD}, and the delay τ. Figure 1.1 illustrates an exemplary propagation scenario (disregarding the elevation), with three scatterers and three paths between the transmit and receive arrays. Depending on the position of the scatterers, different azimuth-of-arrival and azimuth-of-departure occur.

It is the radio propagation channel that determines crucially the characteristics of the entire MIMO system. Therefore, accurate modeling of MIMO

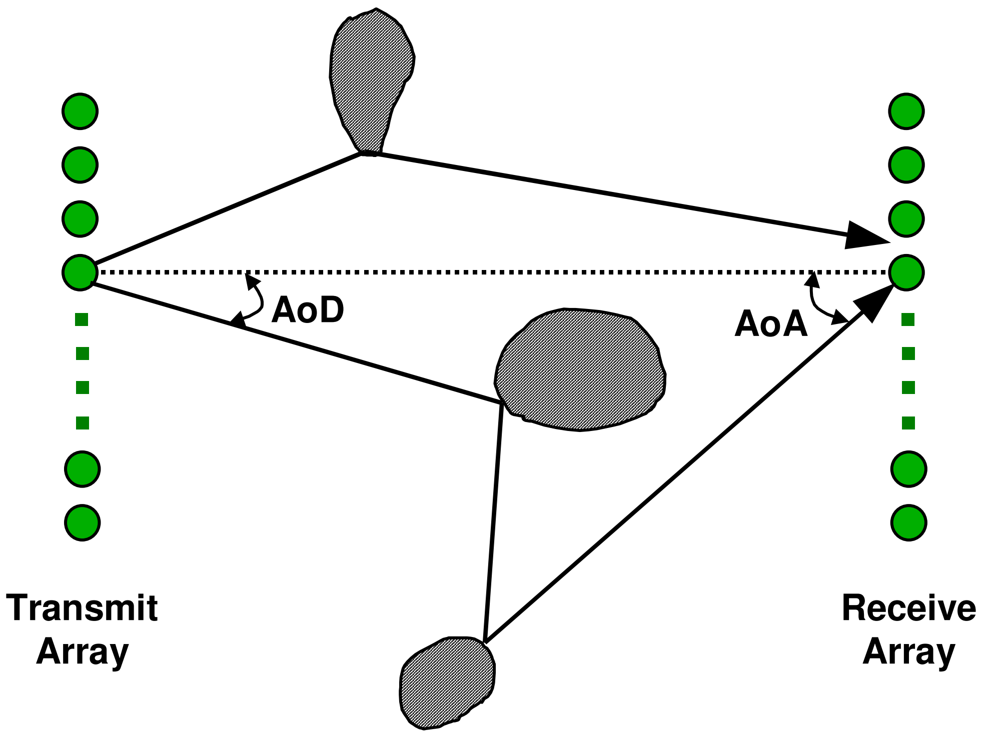


Figure 1.1: Radio wave propagation between transmit and receive antenna arrays.

channels is an important prerequisite for MIMO system design, simulation and deployment [2].

1.2 Cluster based MIMO channel models

Nowadays there are many advanced MIMO channel models based on the concept of multi-path clusters consisting of groups of MPCs showing similar channel parameters such as φ_{AoA}, θ_{AoA}, φ_{AoD}, θ_{AoD}, τ [3, 4]. The major problem of these models is the accurate parametrization of clusters, where the parameters have to be extracted from measurement data (i.e. cluster angular spread, cluster delay spread, cluster lifetime). Hence one has to solve the problem of *cluster identification* out of measurement data.
The starting point is a large number of multi-dimensional parametric channel estimation data, obtained from MIMO measurements. The measurements provide numerous snapshots of the impulse response of the – typically time-varying – radio channel. These measurements are fed to a high-resolution algorithm, e.g. SAGE [5], to estimate the channel parameters for each snapshot individually. Since it has been found in several MIMO studies that these parameters tend to appear in clusters, i.e. in groups of MPCs with similar parameters [6, 7], the problem is to identify these clusters.

In many previous works visual inspection of measurement data was used [6,7] to visually identify clusters. Nevertheless this approach becomes impractical when one has to analyze large amounts of measurement data and, moreover, visual inspection is limited by human senses since one is hardly able to look for clusters in more than three dimensions. Alas, currently there is no fully automatic clustering algorithm available to identify clusters from multi-dimensional parametric MIMO channel estimates.
Recently, a semi-automatic algorithm was introduced in [8], which aims to cluster a non-stationary input set of MPCs, by means of a windowed clustering procedure. This algorithm will be described in Section 3.

With reference to such a clustering algorithm, in this thesis we propose to use a new framework consisting of three algorithms to significantly improve the clustering performance: (i) a new clustering algorithm, (ii) a combined cluster validity metric, which returns an estimation of the correct number of clusters and (iii) an improved cluster shape pruning, which discards the outliers MPCs. We also demonstrate that a previously introduced distance metric, the Multi-path Component Distance (MCD) [9], is particularly suitable for the problem of cluster identification since it leads to a massive performance improvement.

1.3 Organisation

This thesis is organized as follows:

Chapter 2 gives a brief introduction into clustering analysis. It introduces a classification of the principal clustering methods, and it presents the important issue of cluster validity and outlier mining.

Chapter 3 presents the principal practical problem encountered when clustering a set of real measured channel parameter estimates and it introduces the above mentioned semi-automatic clustering algorithm.

Chapter 4 describes the proposed framework, its single components and their interaction.

Chapter 5 assesses the framework performance by means of both simulation results, employing a synthetic scenario, and clustering results of real-world measured data from an indoor big hall environment.

Chapter 6 summarizes the major results of this thesis and draws some conclusions.

1.4 Notation of terms

In the following we formally define the notations which will be used in this thesis.
We consider a set of N estimated MPCs. The generic n-th MPC is represented by its power P_n and by a multidimensional data point vector $\mathbf{x}_n$, collecting the corresponding n-th channel parameters (i.e. $\varphi_{\mathrm{AoA},n}$, $\theta_{\mathrm{AoA},n}$, $\varphi_{\mathrm{AoD},n}$, $\theta_{\mathrm{AoD},n}$, τ_n). The data for all paths are collected in the vector $\mathbf{P} = [P_1 \dots P_N]^T$ and the matrix $\mathbf{X} = [\mathbf{x}_1 \dots \mathbf{x}_N]$. The symbols are summarized in the following table.

N	number of the input MPCs
T	number of time snapshots
L_s	number of data points in the s-th snapshot (observations)
N_k	number of points belonging to the generic k-th cluster
D	number of channel parameters (*attributes*)
K	number of clusters
$\mathbf{X}$	matrix collecting the N data points
$\mathbf{x}_n$	generic D-dimensional data point vector
$\mathbf{C}$	matrix collecting the K cluster representative points (*centroids*)
$\mathbf{c}_k$	generic D-dimensional cluster centroid vector
$\mathbf{P}$	vector collecting the power of the MPCs
P_n	power of the generic k-th MPC
$\mathcal{I}$	vector of the N cluster indices (one index for each data point)
$\mathcal{C}_k$	vector of the MPC indices belonging to k-th cluster

Chapter 2

Introduction to cluster analysis

Imagine we have a set of data objects for analysis where, unlike in classification, the class label of each object is not known. *Clustering* is the process of grouping the data into classes that are relatively homogeneous within themselves and heterogeneous between each other, on the basis of a defined set of variables. These classes are called *clusters*.

Currently clustering is widely used in several fields, such as economy, medicine, statistics and engineering. In order to understand the meaning of clustering, we give the following example [10], which is an actual application of cluster analysis in the field of medicine.

Theoretical syndromes classification Considering patients as a functions of 3 variables, it is possible to plot each patient as a point in a 3-dimensional space. Then, looking for groups of close and similar points (patients), we are able to distinguish between different diseases as shown in Figure 2.1.

The main goal of this thesis is to develop an automatic procedure, based on the cluster analysis, which is able to cluster a given data set, constituted by N MPCs, into a "meaningful" structure. In the sequel we will refer to this problem as the ***MPCs clustering problem***.

In this chapter we present the definition of *cluster analysis* and we introduce its principal components, which are considered and evaluated for the MPCs clustering problem.

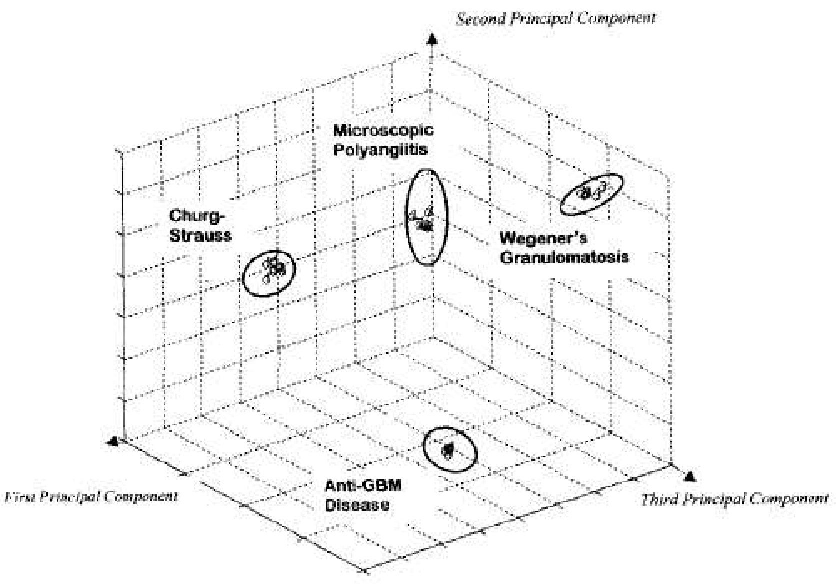


Figure 2.1: 3D clustering of different patient: four different clusters, which represent different disease, are easily distinguishable.

2.1 Problem definition: what is cluster analysis?

Cluster analysis is an important human activity. Early in childhood, one learns how to distinguish between cats and dogs, or between animals and plants, by continuously improving subconscious clustering schemes [11].
Earlier developments of clustering techniques should be credited, primarily, to three areas of research [12]: numerical taxonomy in biology (Sneath and Sokal, 1973), factor analysis in psychology (Holzinger and Harman, 1941) and *unsupervised learning* in *pattern recognition* (Duda and Hart, 1973). In the field of statistics, cluster analysis is an important branch of pattern recognition.
The goal of clustering is to separate a finite data set into a finite and discrete set of "natural groups", which allows us to discover similarities and differences, as well to derive useful inferences about them [13]. Nevertheless, the concept of "natural" is always defined explicitly or implicitly by the adopted clustering procedure [14] and, given a particular data set, different clustering algorithms may lead to different cluster structures. Thus, cluster analysis belongs to the class of *unsupervised learning procedures* [14], since there are no predefined classes and no example that would show what kind of desirable relation should be valid among the data set, which is consequently called *unlabeled.*

In this thesis the following definition of clustering by Han and Kamber is adopted [11].

> *"The process of grouping a set of physical or abstract objects into classes of similar objects is called* ***clustering****. A* ***cluster*** *is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. A cluster of data objects can be treated collectively as one group in many applications".*

A more detailed description of cluster analysis is provided in [11, 12, 14, 15]. More information about pattern recognition and unsupervised learning is provided in [14].

2.2 Clustering of MPCs

Since there is not a rigorous definition of cluster, in order to evaluate the consistence of the clustering results we have to refer to the physical interpretation of the problem, where a cluster corresponds to a real-world scattering object. Therefore, there may be scenarios in which the physical environment does not present any cluster structure. In these cases we expect that the cluster analysis will not produce any well defined cluster structure.
In accordance with the adopted definition of a cluster, we need to find a way to express the similarity (or dissimilarity) between each pair of MPCs of the data set, in order to define a way to say that the MPCs in one cluster are more like one other than the MPCs in other clusters.
The approach we follow is to consider each MPC as a vector in a D-dimensional parameters space, where D is the number of parameters which characterize each MPC. In this way, we are able to calculate the *distance* or the *dissimilarity* between each pair of MPCs.

2.2.1 Distance metrics and dissimilarity functions

A distance metric is the most obvious measure of likeliness between vectors. Many different distance metrics are used in clustering literature. The principal ones are the *Euclidean distance*, the *Manhattan distance* and the more general *Minkowski distance*, which is defined as

$$d(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{d=1}^{D} |\mathbf{x}_{i_d} - \mathbf{x}_{j_d}|^q \right)^{1/q} . \tag{2.1}$$

Setting $q = 1$ or $q = 2$ gives respectively the Manhattan or the Euclidean distance. More information is provided in [11, 14, 15].

A dissimilarity function is a nonnegative number that is close to zero when two vectors are highly similar or "near" each other, and which becomes larger the more they differ [11]. More information is provided in [11, 14, 15].

In the developed framework we employ a specific distance function, the so called *Multi-path Component Distance* (MCD), to quantify the separation between two individual MPCs [9]. This distance will be introduced in Section 4.2, where we will demonstrate it leads to a massive performance improvement when applied to the considered clustering problem.

2.3 Overview of clustering methods

There exists a large number of clustering algorithms in literature. The choice of one clustering algorithm depends both on the type of data available and on the particular purpose and application. In general, major clustering methods can be classified into the following categories [11]:

1. *partitioning methods*,
2. *hierarchical methods*,
3. *density-based methods*,
4. *grid-based methods*,
5. *model-based methods.*

2.3.1 Partitioning methods

Partition-based methods construct the clusters by creating various partitions of the data set. Thus, partitioning assigns to each data point $\mathbf{x}_n$ a cluster index $\mathcal{I}_n$, which assume integer values in the interval $[1,\ K]$, where the number of clusters K must be provided by the user.
The clustering partition must respect the two following rules [11]:

1. each cluster must contain at least one object,
2. each data point must belong to exactly one cluster.

Notice that the second requirement can be relaxed in some *fuzzy* partitioning techniques. More information about partitioning methods is provided in [14, 15].

2.3.2 Hierarchical methods

Hierarchical clustering methods build a cluster hierarchy, i.e. a tree of clusters also known as *dendrogram*, which is a tree diagram often used in the cluster analysis to represent the relation of similarity between the elements of the data set. Hierarchical clustering methods can be further classified as follow [11]:

1. *agglomerative hierarchical clustering*,

2. *divisive hierarchical clustering.*

The former achieve the hierarchical decomposition using a bottom-up process. The latter are based on a top-down approach. An agglomerative clustering starts with one-point (*singletone*) clusters and recursively merges two or more most appropriate clusters. In contrast, a divisive clustering starts with one cluster consisting of all data points which is recursively split into non-overlapping clusters.
The computation to go from one level to another is usually simpler for agglomerative procedures, which are preferable for implementation [14].

2.3.3 Density-based, grid-based and model-based methods

The key idea of density-based methods is to continue growing the given cluster as long as the density in the "neighborhood" exceeds some threshold, which is correlated with the minimum number of data points within a given radius.
Grid-based methods quantize the data points space into a finite number of cells that form a grid structure. All of the clustering operations are performed on the grid structure.
Model-based methods hypothesize a model for each of the clusters and find the best fit of the data set to the given model. These methods may locate clusters by constructing a density function that reflects the spatial distribution of the data points.

2.4 Suitable algorithms for the MPCs clustering problem

In the considered MPCs clustering problem we are not allowed to impose any statistical model to the input data set, because this is not available and, on

the contrary, our purpose is to discover this model due to cluster analysis. Moreover the distribution of the MPCs in the parameter space do not show any kind of regularity and it is typical to have a cluster structure with very big differences in shape and dimension, which reflects the difference of the physical objects in the considered environment.
For these reasons, the most prominent clustering methods for the MPCs clustering problem seem to be the partitioning and the hierarchical ones.
More precisely, the two most suitable algorithms are:

1. the *k-means algorithm* (partitioning),
2. the *hierarchical tree algorithm* (agglomerative hierarchical).

These two algorithms are well known, not too time-consuming and easy to implement [16]. Moreover, these algorithms are used in a previously published *semi-automatic clustering toolbox* [17], which will be described in Section 3.

2.4.1 The k-means algorithm

The classic k-means algorithm partitions the set of input objects into a given number of K clusters so that the resulting *intra-cluster* similarity is high but the *inter-cluster* similarity is low [11]. Cluster similarity is measured with respect to the mean value of the objects in a cluster, which represent the cluster's center of gravity and therefore is called *cluster centroid.*
The algorithm proceeds as follow. First, it randomly selects K objects, which initially represent a cluster centroid each. All the remaining objects are assigned to the cluster to which it is the most similar, based on the distance between the object and the centroid. It then computes the new mean for each clusters. This process iterates until the *criterion function* converges. Typically, the *squared-error criterion* is used, defined as

$$D = \sum_{n=1}^{N} d(\mathbf{x}_n, \mathbf{c}_{\mathcal{I}_n})^2 \ , \tag{2.2}$$

where D is the sum of the squared distance d between each point and its own cluster centroid, where d is an appropriate distance function.
In accordance to its definition as cluster's center of gravity, the generic k-th cluster centroid is defined as

$$\mathbf{c}_k = \sum_{n \in \mathcal{C}_k} \mathbf{x}_n \ , \tag{2.3}$$

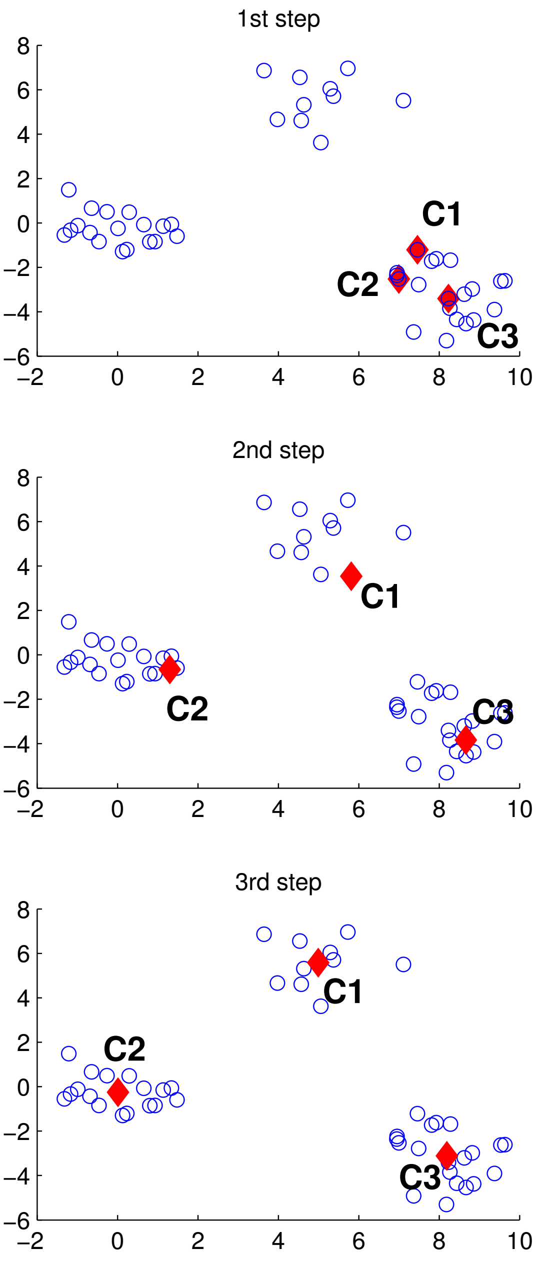


Figure 2.2: Evolution of the centroid positions during one run of the k-means algorithm. For the given data set, the algorithm converges in three steps.

where $\mathcal{C}_k$ is the vector of the MPC indices belonging to the k-th cluster. This criterion tries to make the resulting K clusters as compact and as separate as possible.
Figure 2.2 shows centroid positions evolution during one run of k-means on a numerical 2-dimensional input data set, represented by the blue circles. The cluster centroids are represented by the red diamonds. In the first step three centroids ($C1$, $C2$, $C3$) are randomly chosen from the data points. In the second step we can observe $C1$ and $C2$ move to the top and to the left, respectively, pulled by the two corresponding groups of points. The third centroid $C3$ moves a little downward, because some points belonging to the lower group are still associated to $C1$ and $C2$. The third step is the final one. All the MPCs are closer to their own centroid, than to the other ones. Since there is no change in the centroid positions the algorithm stops.
The k-means algorithm is very simple and can be easily implemented. It can work very well with compact and hyper spherical clusters [18].
The time complexity of is $\mathcal{O}(N \cdot K \cdot i)$, where i is the number of iterations. Since K and i are usually much less than N, k-means can be used to cluster large data sets.
The main drawbacks of k-means are summarized in the following [18].

1. There is no efficient and universal method at this point for identifying the initial partitions and the number of clusters K. The convergence of centroids vary with different initial points. Therefore, a general strategy for the problem is to run the algorithm many times, with random initial centroids.

2. The iteratively optimal procedure of k-means cannot guarantee convergence to a global minimum and sometimes it gets stuck in local minimum. This can be circumvented by running the algorithm with many initial guesses.

3. The k-means algorithm is sensitive to outliers[1] and noise. Even if an object is far away from the cluster centroid, it is still forced into a cluster and, thus, distorts the cluster shape.

More information about k-means is provided in [11, 14, 15].

2.4.2 The hierarchical tree algorithm

The hierarchical tree algorithm belongs to the class of agglomerative hierarchical clustering algorithms. This bottom-up strategy starts by considering

[1]The concept of outliers will be defined in Section 2.6

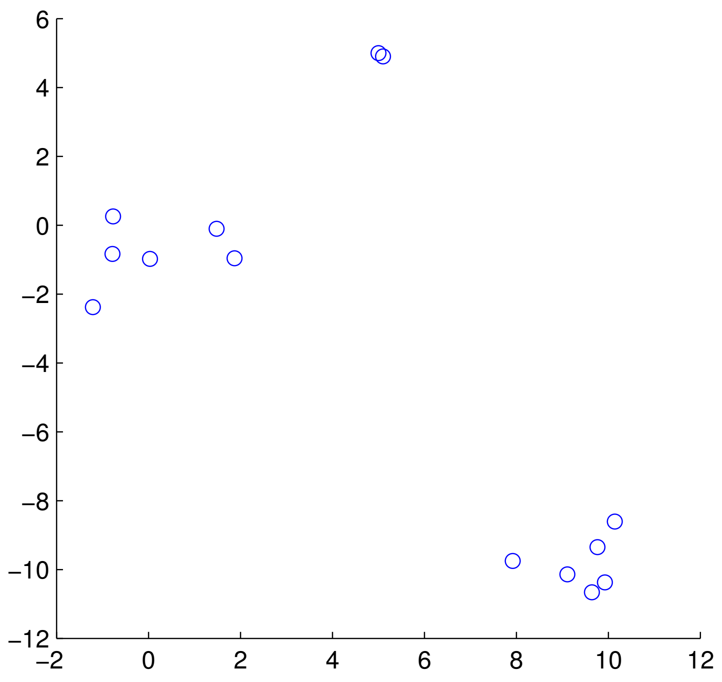


Figure 2.3: Clustered input data set.

each object as a singletone cluster and then, at each step, merges the two closest clusters, in accordance to a specified distance function. Thus, step by step, the clusters are merged into larger and larger clusters, until the algorithm produces a specified number of clusters or until all of the objects are collected in the same clusters.

If we stop the algorithm when all the data points belong to the same cluster we can produce a dendrogram, which represents the merging sequence. For instance, applying this process to the numeric input data of Figure 2.3, we obtain the dendrogram shown in Figure 2.4.

Each node of the tree represents a group of data points, therefore a cluster. The lengths of the branches of the tree represent the distance between the clusters. If we cut the dendrogram at a specific height by an horizontal line, as shown in Figure 2.5, we obtain the cluster structure for a specified number of clusters, which is the number of branches which intersect the line.

In literature, the most common distance functions used to evaluate the distance between cluster are the following [11]:

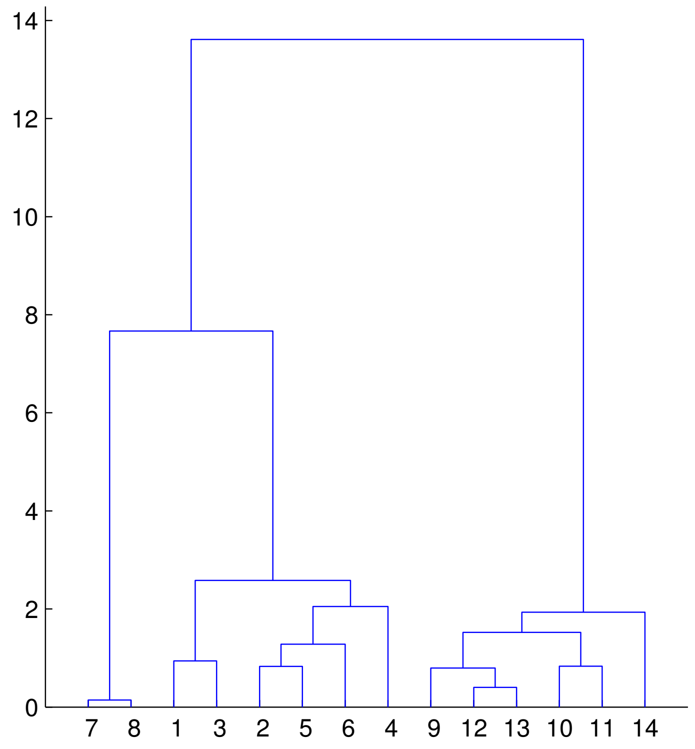


Figure 2.4: Dendrogram: the tree represents the merging sequence.

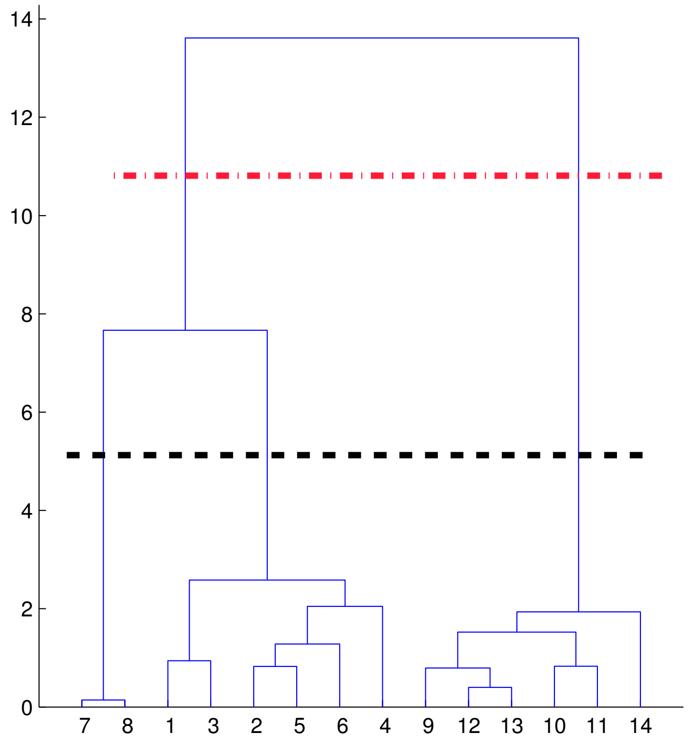


Figure 2.5: By cutting the tree at a specific height we get a different number of clusters (i.e. two for the red line or three for the black one).

$$\begin{aligned}
d_{min}(i,j) &= \min_{\substack{h\in\mathcal{I}_i\\ k\in\mathcal{I}_j}} d(\mathbf{x}_i,\mathbf{x}_j)\ , && (2.4)\\
d_{max}(i,j) &= \max_{\substack{h\in\mathcal{I}_i\\ k\in\mathcal{I}_j}} d(\mathbf{x}_i,\mathbf{x}_j)\ , && (2.5)\\
d_{avg}(i,j) &= \frac{1}{N_i N_j}\sum_{h\in\mathcal{I}_i}\sum_{k\in\mathcal{I}_j} d(\mathbf{x}_i,\mathbf{x}_j)\ , && (2.6)\\
d_{mean}(i,j) &= d(\overline{m}(i),\overline{m}(j))\ , && (2.7)
\end{aligned}$$

which correspond respectively to the *minimum distance*, the *maximum distance*, the *mean distance* and the *average distance*, where

$$\overline{m}(k) = \frac{1}{N_k}\sum_{n\in\mathcal{I}_k}\mathbf{x}_n\ .$$

The principal drawbacks of the classical hierarchical tree algorithm are summarized in the following [18].

1. The algorithm lacks of robustness and is sensitive to noise and outliers. Once an object is assigned to a cluster, it will not be considered again, which means the algorithm is not capable of correcting possible previous misclassification.

2. The computational complexity is at least $\mathcal{O}(N^2)$, limiting its application in large-scale data sets.

More information about hierarchical tree is provided in [11, 15].

2.5 Cluster validity

Clustering is an unsupervised process. As a consequence, in most applications the final partitions of a data set require some sort of evaluation. The procedure of evaluating the results of a clustering algorithm is known under the term *cluster validity* [19].

2.5.1 Different validity approaches

In general terms, there are three main approaches to investigate cluster validity [19]:

1. *external criteria* based approach,

2. *internal criteria* based approach,
3. *relative criteria* based approach.

The first approach implies that we evaluate the results of a clustering algorithm based on a *pre-specified* structure, which is imposed on the data set and reflects our intuition about the clustering structure of the data set.
The goal of the second approach is to evaluate the clustering results of an algorithm using only quantities and features inherited from the data set.
The basis of the validation methods based on external or internal criteria is *statistical testing*. Thus, the major drawback of these techniques is their high computational demands. On the contrary, since an approach based on relative criteria does not involve statistical tests, it performs well for the MPCs clustering problem in which we cannot impose any statistical model to the MPCs input data set.
The principle of this approach is to choose the best clustering scheme of a set of defined schemes according to a pre-specified criterion. More specifically, the problem can be stated as follows [20].

> *"Let $\mathcal{P}_{alg}$ be the set of parameters associated with a specific clustering algorithm (i.e. the number of clusters K). Among the clustering schemes S_i, $i = 1, \ldots, K$, achieved by a specific algorithm, for different values of the parameters in $\mathcal{P}_{alg}$, choose the one that best fits the data set".*

2.5.2 Number of clusters

The choice of the number of clusters K to be identified from the data set is an important subproblem of the MPCs clustering problem and, probably, the most difficult one. A division with too many clusters complicates the results, therefore, makes it hard to interpret and analyze, while a division with too few clusters causes the loss of information and misleads the final decision [18]. Both of these situations have to be avoided.
Since we do not have a-priori knowledge of the optimum number of cluster K^{opt}, we aim to find this number by means of a relative criteria based approach. This procedure is based on a *validity index*. Selecting a suitable performance index $I_{\text{perf},K}$ we proceed with the following steps [20].

1. The clustering algorithm is run for all values of K, between a minimum K_{min} and a maximum K_{max} number of clusters, defined a priori.

2. For each of the values of K, the algorithm is run R times, using different set of values for the other parameters of the algorithm (i.e. different initial conditions), evaluating $I^{(i)}_{\text{perf},K}$, for $i = 1, \ldots, R$.

3. The best values of the index $I^{(\text{opt})}_{\text{perf},K}$ corresponds to the validity index v_K, in correspondence of K clusters, and it is plotted as the function of K, from K_{min} to K_{max}.

Based on this plot we may identify the optimum number of cluster K_{opt}. We have to stress that there are two approaches for defining the best clustering, depending on the behavior of v_k with respect to K.

1. The validity index exhibit a regular profile as K increases (i.e. increasing trend until a certain value of K and decreasing trend after).

2. The validity index does not exhibit a regular profile as K increases.

In the former case we seek the maximum (minimum) of the plot. Differently, in the latter case we search for the values of K at which a significant local change in the value of the index occurs. This change appears as a "knee" in the plot and it is an indication of the number of clusters underlying the data set [20]. Moreover, the absence of a knee may be an indication that the data set contains no clustering structure.
All the indices usually emphasize the *compactness* of intra-cluster MPCs and *isolation* of inter-cluster MPCs and consider the comprehensive effects of several factors.
Milligan and Cooper compared and ranked 30 indices according to their performance over a series of artificial data set. Among these indices, the *Caliñski-Harabasz index* achieve the best performance [18]. This index will be introduced in Section 4.4.1.
Since the validity indices are usually data dependent, the good performance of an index for a certain data set does not guarantee the same behavior with a different data set. Thus, cluster validity results are more reliable if they are achieved by a combination of different validity indices [18].
More information about cluster validity methods is provided in [12,14,18–20].

2.6 Outlier mining

Very often there exist data objects that do not comply with the general behavior or model of the data. Such data objects, which are grossly different from or inconsistent with the remaining data set are called *outliers*.

Outliers can be caused by measurement or execution error. Alternatively, outliers may be the result of inherent data variability.
Many algorithms try to minimize the influence of outliers or eliminate them all together. This, however, could result in the loss of important hidden information, since the outliers themselves may be of particular interest [11].
In literature the outliers detection and analysis is known under the term *outlier mining*, which can be described as follows [11].

> *"Given an input set of N data points and the number O of expected outliers, find the first O data points that are considerably dissimilar, exceptional, or inconsistent with respect to the remaining data".*

As a consequence, the outlier mining can be decomposed in the following two subproblems.

1. Define which data points can be considered as inconsistent in a given data set.

2. Find an efficient method to mine these outliers.

On the base of the same reasonings done for the cluster validity problem, it is evident that we cannot use visual identification methods in outlier mining. More information about outlier mining is provided in [11, 21].

Chapter 3

Automatic clustering of non stationary MPCs

Recently, a heuristic semi-automatic clustering tool box was introduced by Jari Salo et al. [8, 17]. This algorithm aims to cluster a non-stationary input data set, based on a windowed clustering procedure.

The input data matrix $\mathbf{X}$ is obtained by the estimation of several MPCs at different discrete instants of time, which in literature are commonly called *snapshots*. The total number N of MPCs is achieved as

$$N = \sum_{s=1}^{T} L_s \ , \tag{3.1}$$

where the number L_s of estimated MPCs per snapshot is not constant.

With reference to such a clustering algorithm, we developed a new clustering framework to cluster a data window formed by S snapshots, where S is chosen to be stationary within the window. The proposed framework can be seen as an extension of the semi-automatic algorithm, in order to significantly improve its performance. The framework will be introduced in Section 4.

This chapter is organized as follows. In Section 3.1 we describe the principal practical problems we encounter in clustering a data set obtained from real-world measurements. Then, the previously introduced semi-automatic algorithm in concisely described in Section 3.2.

3.1 Practical problems in clustering measurement data

The principal practical problems of an automatic clustering procedure that have to be overcome in order to cope with real measured data are listed in the following [17].

1. *Multi-dimensionality of the data.*
2. *Non-stationarity of the data.*
3. *Outliers throughout the data.*

3.1.1 Multi-dimensionality of the data

The input data set is multi-dimensional and its different dimensions are measured in different measurement units and scales. For instance, delay and angular MPC parameters are measured in nanoseconds and radians, respectively.

Moreover, real-world clusters come in strange shapes, or have no simple shape at all. "The nature appears to have no moral qualms on producing non-spherically or non-ellipsoidally structured groups of multipaths. While such non-regular cluster shapes are relatively easily recognized by human eye, they are difficult to be identified automatically by mathematical algorithms" [17].

Two different approaches are suitable to cope with the data multidimensionality problem.

A first strategy consists in the so called *sequential clustering*. This means one sequentially clusters the MPC coordinates corresponding to the same dimension. For instance, we can starting clustering in delay domain first and subsequently clustering AoAs and AoDs (jointly or separately).

A second approach is the so called *joint clustering*. This means one looks for D-dimensional clusters, without any intermediate step, but for this one has to *scale* "somehow" the different dimensions, e.g. by assigning proper *weights*.

Although it is intuitively clear that joint clustering is more promising as the cluster structure in the data is more visible in high-dimensional spaces [22], scaling is a very critical problem, because it involves making some manipulation of the original data which may lead to artifacts and wrong results. Therefore the assignment of weights have to be based on thorough research [15].

3.1.2 Non-stationarity of the data

Typically the input data set is non-stationary, because of the movement of antennas and/or scatterers. From snapshot to snapshot paths are dropped out and new paths pop up or old paths reappear. Sometimes entire clusters can vanish and then reappear after some time. Therefore, clusters are not static, but move in the D-dimensional parameter space [17].
In order to cope with data non-stationarity we have to detect the clusters movement from one snapshot to the following one and couple the corresponding clusters. In literature, this process is known as *cluster tracking.*

3.1.3 Outliers throughout the data

Data outliers are spread everywhere in the data. Typically they are a result of randomly appearing MPCs with short life-times, artifact estimates due to estimation noise or convergence to local minimum in the nonlinear optimization of the parameter estimator [17]. Therefore, an automatic clustering procedure must provide some tool for outlier mining, in order to *prune* the clusters.

3.2 The semi-automatic clustering algorithm

The semi-automatic algorithm introduced by Jari Salo et al. [8] is based on windowing the data, processing each window with some clustering algorithm, cluster tracking and cluster pruning. In [8] it has been shown that this algorithm has a fully automatic clustering capability with synthetic data, but some manual processing of the clustering results may be required for satisfactory results for real-world data. The improvement of the performance of this algorithm is the motivation of this work.

Figure 3.1 summarizes the structure of the algorithm. A detailed explanation in provided in [17]. The main steps of the algorithm are detailed below:

Step 1 – Windowing data If processing the first window, select a data window with a predefined with. Otherwise slide the data window forward by predefined length.

Step 2 – Clustering data Cluster the MPCs in the data window using on some clustering algorithm.

Step 3 – Cluster tracking Match the cluster indices with the ones from the previous window. If new clusters appear, they will have to be assigned new cluster indices.

Step 4 – Cluster pruning Discard the outliers. Also clusters with only few data points may be discarded optionally.

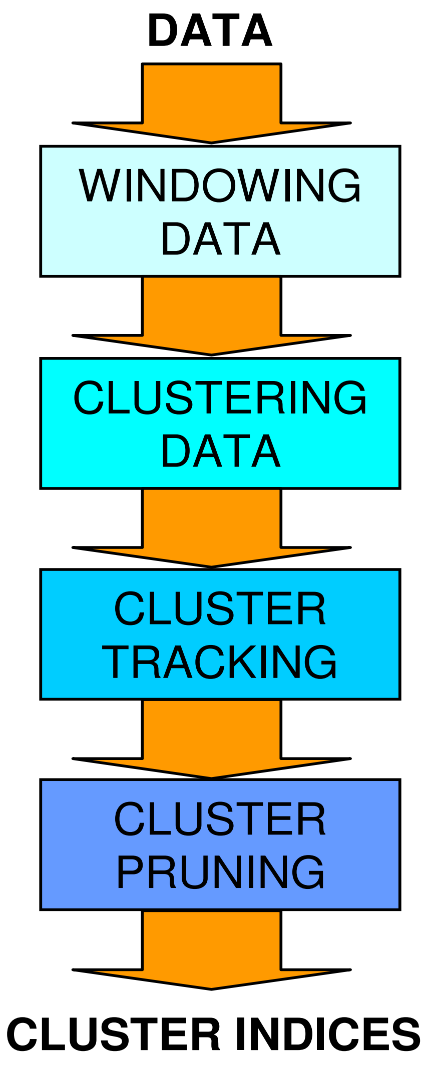


Figure 3.1: Reference semi-automatic clustering algorithm.

Chapter 4

Proposed clustering framework

In this section we introduce the new proposed clustering framework, which consists of the following four components.

1. A new distance metric: the *multi-path component distance* [22].
2. A new clustering algorithm: *KPowerMeans* [23].
3. A combined cluster validity index: *CombinedValidate* [23].
4. An improved pruning algorithm: *ShapePrune* [23].

The framework aims to improve the performance of **Step 2** of the semi-automatic algorithm introduced in Section 3 we chose as reference.
In the following we describe the proposed framework in Section 4.1 and its components in Sections 4.2 - 4.5.

4.1 The proposed clustering framework

In this section we describe the proposed clustering framework [23].

Framework algorithm:

```
1. Do For all number of clusters $K = K_{\min}$ To $K_{\max}$
    (a) Cluster window with $K$ clusters:
        $\mathcal{R}_K = \text{KPowerMeans}(\mathbf{P}, \mathbf{X}, K)$
    (b) Validate $K$ clusters:
        $v_K = \text{CombinedValidate}(\mathcal{R}_K)$
    (c) Next $K$
```

2. Find optimum number of clusters:
   $K_{\text{opt}} = \arg\max_K v_K$, $\mathcal{R}_{\text{opt}} = \mathcal{R}_{K_{\text{opt}}}$
3. Prune optimum cluster set:
   $\mathcal{R}^p = \text{ShapePrune}(\mathcal{R}_{\text{opt}})$

Initially, a range $[K_{\min}, K_{\max}]$ for the possible number of clusters has to be specified. The number of clusters, K, and the data from all MPCs, $\mathbf{P}$ and $\mathbf{X}$, are external parameters for the clustering algorithm. For each possible K, the clustering algorithm KPowerMeans is performed, the results are collected in the output sets $\mathcal{R}_K$. Subsequently, each result is validated by Combined-Validate which provides the validation index v_K.
The optimum number of clusters K_{opt} is finally determined by the largest validation index v_K with corresponding cluster set $\mathcal{R}_{\text{opt}}$. The optimum set is then pruned by ShapePrune for improved visualization and future cluster tracking.

4.2 Multi-path component distance — MCD

The multi-path component distance (MCD) was first introduced in [9] as an intermediate measure on the way to quantify the complete multi-path separation of the radio channel. In the developed framework we use the MCD to quantify the separation between two individual MPCs.
A similar measure which aims to quantify the multi-path separation is also provided in [24].

4.2.1 Theoretical considerations

In the considered clustering problem, the delay and the angular coordinates are typically given in nanoseconds and radians, respectively. Thus, given two generic MPCs we have to cope with the following three subproblems concerning the multi dimensionality of the data (of Section 3.1.1).

1. The different coordinates are expressed in different measurements units.
2. The values of the delay distance are very small compared to the values of the angular distance.
3. The angular coordinates are periodical in $[-\pi,\ \pi)$.

The first point is strictly connected with the possibility to do *joint multidimensional clustering*. In fact, when doing joint clustering one aims to compute the different dimensions all together and therefore one needs all the MPC coordinates to be a-dimensional.
The second subproblem leads to a negligible importance of the delay domain in comparison to the angular one. Thus in the case of *joint multidimensional clustering* the clustering algorithm is not able to identify clusters in the delay domain.
The last point is not critical, but it shall remind that one is not allowed to simply sum angular coordinates, but, in the selected metric, one must take angular periodicity into account. This means one has to map the sum of several angular coordinates to the interval $[-\pi,\ \pi)$.

The multi-path component distance efficiently solves the problem of data multi-dimensionality, both when employing the original delay distance and the modified delay distance. In the first case the metric scales the delay distance and the angular distances to be in the dimensionless interval $[0,\ 1]$. The second case differs by the first for the delay distance scaled to be in the dimensionless interval $[0,\ \zeta]$. Both $[0,\ \zeta]$ and $[0,\ 1]$ are roughly of the same range, which make the scaling to be consistent.
Furthermore the angular distance in (4.6) uses the trigonometric functions to cope with angular periodicity.

As a consequence of the previous considerations, we are able to use the multi-path component distance for *joint multi-dimensional clustering* in the delay-angular domain.

4.2.2 MCD mathematical expression

If we consider two generic MPCs i and j, the MCD distance between them is given as

$$\mathrm{MCD}_{ij} = \|\mathbf{M}_{D,ij}\| = \sqrt{\sum_{d=1}^{D} \mathrm{MCD}^2_{\mathrm{d,ij}}}\ , \tag{4.1}$$

where $\mathrm{MCD}_{\mathrm{d,ij}}$ denotes the MCD distance in a generic dimension d and $\mathbf{M}_{D,ij}$ is the vector which contains all the $\mathrm{MCD}_{\mathrm{d,ij}}$ for $d = 1, \ldots, D$. This distance corresponds to the radius of a hyper-sphere, in the *normalized* D-dimensional MPC parameter space.
In the considered clustering problem typically D equals 3 or 5, including delay

(τ), azimuth-of-arrival (φ_{AoA}), azimuth-of-departure (φ_{AoD}) and, eventually, elevation-of-arrival (θ_{AoA}) and elevation-of-departure (θ_{AoD}).
The generic $\mathrm{MCD}_{\mathrm{d,ij}}$ has a different expression for delay and angular data.

The delay distance is obtained as

$$\mathrm{MCD}_{\tau,ij} = \frac{|\tau_i - \tau_j|}{\Delta\tau_{\max}} \ , \tag{4.2}$$

where τ_i, τ_j denote the delay coordinate of the i-th and j-th MPCs and

$$\Delta\tau_{\max} = \max_{i,j}\{|\tau_i - \tau_j|\} \ .$$

In addition to (4.2), previously introduced in [9], we present a second expression of the delay distance in which we scale the delay distance with the normalized delay spread $\tau_{\mathrm{std}}/\Delta\tau_{\max}$ and we introduce a delay scaling factor (DSF) ζ. The new delay distance[1] is given as

$$\mathrm{MCD}^{(std)}_{\tau,ij} = \zeta \cdot \frac{|\tau_i - \tau_j|}{\Delta\tau_{\max}} \cdot \frac{\tau_{\mathrm{std}}}{\Delta\tau_{\max}} \ , \tag{4.3}$$

where

$$\tau_{\mathrm{std}} = \sqrt{\frac{\sum_{k=1}^{N} P_k \cdot (\tau_k - \overline{\tau})^2}{\sum_{k=1}^{N} P_k}} \ , \tag{4.4}$$

$$\overline{\tau} = \frac{\sum_{k=1}^{N} P_k \cdot \tau_k}{\sum_{k=1}^{N} P_k} \ . \tag{4.5}$$

The delay spread τ_{std} takes into account the power P_k of the MPCs. The DSF ζ gives the delay domain more “importance” when necessary. This can have advantageous effects when automatically clustering real-world data. Without further indication we use the original delay distance.

For angular data the distance is obtained as

$$\mathrm{MCD}_{\mathrm{AoA/AoD,ij}} = \frac{1}{2}\left\| \begin{pmatrix} \sin(\theta_i)\cos(\varphi_i) \\ \sin(\theta_i)\sin(\varphi_i) \\ \cos(\theta_i) \end{pmatrix} - \begin{pmatrix} \sin(\theta_j)\cos(\varphi_j) \\ \sin(\theta_j)\sin(\varphi_j) \\ \cos(\theta_j) \end{pmatrix} \right\| \ , \tag{4.6}$$

[1]In the following we will refer to this distance as to the modified delay distance.

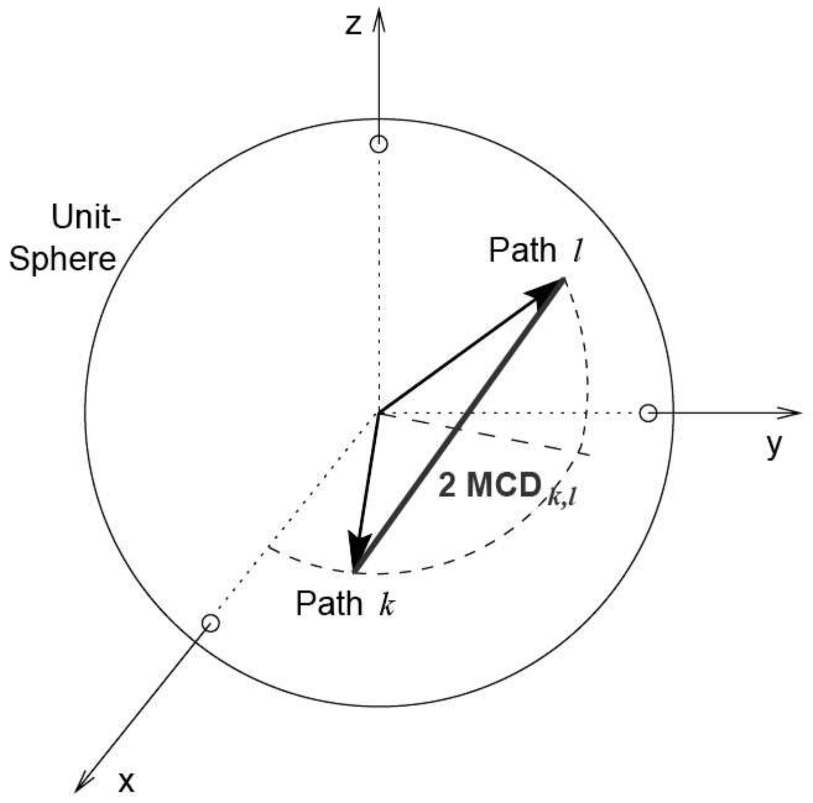


Figure 4.1: Geometrical definition of $\mathrm{MCD}_{\mathrm{AoA/AoD,ij}}$ of the multi-path component distance.

for AoA and AoD likewise, but separately. This corresponds to the normalized Euclidean distance, by means of the factor 2, between two points on a unit-sphere. The two points can represent either the two AoA dimensions (φ_{AoA},θ_{AoA}) or the two AoD dimensions (φ_{AoD},θ_{AoD}), as shown in Figure 4.1 [9].

The resulting distance is given as

$$\mathrm{MCD}_{ij} = \sqrt{\mathrm{MCD}^2_{\mathrm{AoA,ij}} + \mathrm{MCD}^2_{\mathrm{AoD,ij}} + \mathrm{MCD}^2_{\tau,ij}}\ . \tag{4.7}$$

4.3 Clustering algorithm — KPowerMeans

The task of the KPowerMeans (KPM) clustering algorithm is to assign a cluster index to each of the N MPCs. The concept of the k-means algorithm [11] is well suited for this challenge if one uses the appropriate distance function, which is able to scale the data in a suitable way.

4.3.1 Algorithm description

In this section we describe the KPowerMeans algorithm in detail. The algorithm scheme is presented in the following.

KPowerMeans clustering algorithm:

```
1. Randomly choose $K$ initial centroid positions $\mathbf{c}_1^{(0)}, \ldots, \mathbf{c}_K^{(0)}$
2. For $i = 1$ To *MaxIterations*
    a. Assign MPCs to cluster centroids and store indices:
       $\mathcal{I}_n^{(i)} = \arg\min_k \{P_n \cdot \mathrm{MCD}(\mathbf{x}_n, \mathbf{c}_k^{(i-1)})\},$   (4.8)
       $\mathcal{I}^{(i)} = [\mathcal{I}_1^{(i)} \ldots \mathcal{I}_N^{(i)}],\ \mathcal{C}_k^{(i)} = \mathrm{Indices}_n(\mathcal{I}_n^{(i)} \overset{?}{=} k)$
    b. Recalculate cluster centroids $\mathbf{c}_k^{(i)}$ from the allocated MPCs to coincide with the clusters' centers of gravity:
       $\mathbf{c}_k^{(i)} = \frac{\sum_{j \in \mathcal{C}_k^{(i)}} (P_j \cdot \mathbf{x}_j)}{\sum_{j \in \mathcal{C}_k^{(i)}} P_j}$   (4.9)
    c. If $\mathbf{c}_k^{(i)} = \mathbf{c}_k^{(i-1)}$ for all $k = 1 \ldots K$, then break loop
       Else Next $i$
3. Return $\mathcal{I}^{(i)}$, $\mathbf{C}^{(i)}$
```

This algorithm iteratively minimizes the criterion function D (4.10), which corresponds to the total sum of power-weighted distances of each path to its associated cluster centroid. This constitutes the main difference to the classic k-means, in which a non-weighted criterion function is used (typically the sum of squared Euclidean distance).
In the following we describe the single steps of the algorithm in more detail.

Ad 1) Since the algorithm aims to seek the global minimum of the criterion function D using an iterative approach, we need a set of initialization values (the centroid starting positions), from which starting the iterative minimization. The centroid starting positions are chosen randomly from the data $\mathbf{X}$.

Ad 2a) Every MPC is associated with a cluster centroid such that the function of the total sum of differences

$$D = \sum_{n=1}^{N} P_n \cdot \mathrm{MCD}(\mathbf{x}_n, \mathbf{c}_{\mathcal{I}_n^{(i)}}) \tag{4.10}$$

is minimized. We use the MCD as the basic distance function [9,22], described in Section 4.2, but we also include the *power of the paths*. Analytic treatment of this global distance results in (4.8). The index $\mathcal{I}_n^{(i)}$ is the cluster number for the n-th multi-path in the i-th iteration step. Vice-versa, the set $\mathcal{C}_k^{(i)}$ contains the MPC indices belonging to the k-th cluster in the i-th iteration step.

Ad 2b) In the second step of the iteration, the centroids move to the centers of gravity of the groups of MPCs allocated in the previous step. Note that moving centroids can result in a new group of MPCs that will be associated with the centroid in the next iteration step.

Ad 2c) If the centroids do not move any more the algorithm has converged to a minimum of the criterion function D. The upper bound *MaxIterations* to the maximum number of iterations is necessary in practical implementation, to avoid that the number of iterations gets too high.

Ad 3) The output of the algorithm are the index set $\mathcal{I}^{(i)}$ and the associated cluster centroids matrix $\mathbf{C}^{(i)}$, which were obtained by the last iteration.
As described above, we perform KPowerMeans multiple times and we consider the solution determined by the smallest value of the criterion function D.

4.3.2 Initialization

In Section 4.3.1 we have assume a random initialization of the KPowerMeans algorithm. This is a very simple way to initialize the algorithm, but its drawback is that the random choice does not guarantee the algorithm to seek the global minimum. Thus, following this approach, we need to perform KPowerMeans multiple times and to consider the solutions corresponding to the smallest value of the criterion function D.
Typically we have to perform from 20 up to 50 iterations to achieve a stable[2] solution, depending on the structure of the input data $\mathbf{X}$ (the more the

[2]With stable we mean a reproducible solution. If we run KPowerMeans with the same input we expect to get the same partition of the data.

MPCs are spread in the D-dimensional parameter space, the more the data set is challenging, hence difficult to be clustered). It is intuitive that this dramatically affects the required computation time. Therefore, a further improvement of the initialization process is required, in order to speed up the computational time. A new promising method is proposed in [25].

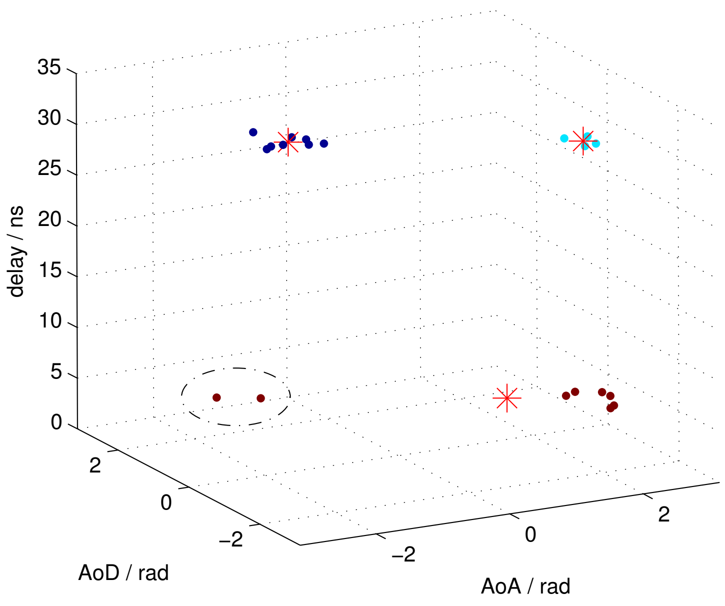
4.3.3 Outliers

By *including power* into the distance function, cluster centroids are pulled to points with strong powers. This is intuitive and yields massive performance improvements, overcoming the k-means sensibility to outliers, which is one of its principal drawbacks, as shown in Section 2.4.1.

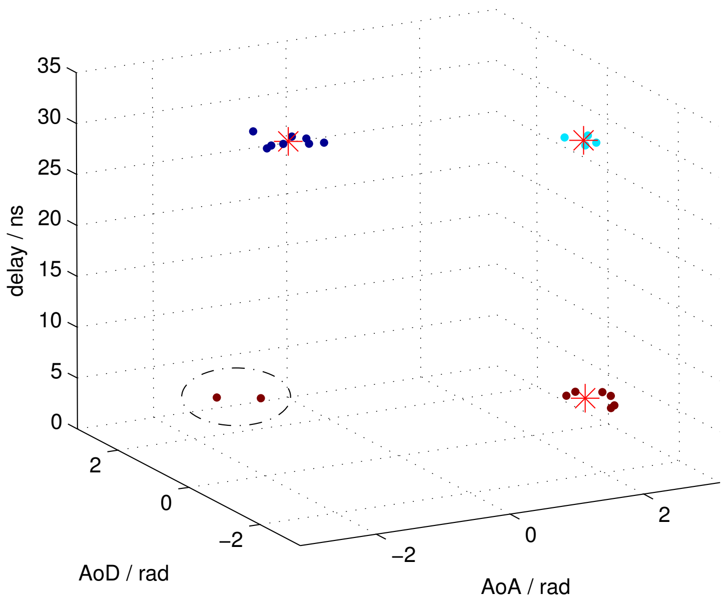
Figure 4.2 is a synthetic example which shows that using a power-weighted distance reduces the clustering sensibility to the presence of weak-powered outliers. We clustered 20 synthetic MPCs represented by colored dots, where a different color denote a different cluster. The MPCs have the same power (say P), except for the two MPCs in the bottom left part of the picture, surrounded by a dash-dotted ellipse, which have reduced power, equal to $P/10$. Figure 4.2(a) shows the clustering result using the classic k-means with the MCD distance[3]. Figure 4.2(b) shows the clustering result using the KPowerMeans. In both cases we specified 3 as the number of cluster and the resulting three cluster centroids are represented by the red stars.
One can see that, even if the clustering result is the same (the corresponding clusters are represented by the same number), in the non-weighted case, the centroid of the "lower" cluster has moved to the "left" part of the picture, due to the presence of the two low-powered MPCs, which act as outliers. It is intuitive that with more challenging data set (i.e. real-world MPCs) this outlier sensibility may lead to erroneous results.
On the contrary, using KPowerMeans, the two outliers do not have any effect on the centroid position, which remains practically the same as if we would cluster the data set without the two outliers. Thus, this simple example prove the robustness of KPowerMeans against the presence of outliers, which is a fundamental propriety we need in clustering real-world measurement data sets, which are typically characterized by the massive presence of outliers.

We want to point out that in the KPowerMeans algorithm, power is used to weight the *point-to-centroid* distance and not the *point-to-point* distance. This aspect is crucial for the final clustering result, since there is no reason

[3]This can be done using the KPowerMeans and assigning unitary power to all the MPCs



(a)



(b)

Figure 4.2: KPowerMeans and classic k-means behavior in presence of outliers. (a) Classic k-means. The presence of the two weak-powered outliers affect the position of centroid 3. (b) KPower-Means. The position of the three centroids is insensible to the presence of the two outliers.

because power could be used to express a measure of dissimilarity between to MPCs. Roughly speaking, if two different MPCs have different power this does not imply they should belong to different clusters.
On the contrary, by power-weighting the *point-to-centroid* distance we pull the centroid positions toward high density power regions of the MPCs space, were it is reasonable to expect a real cluster is located. Therefore the new power-weighted metric is suitable only for *partitioning methods*, based on the concept of centroid, and not for hierarchical methods.

4.3.4 A new cluster definition

The global criterion function (4.10) introduces itself an inherent definition for a cluster.

> *For a given number of clusters, clusters are chosen such that they minimize the total distance from their centroids.*

This implies that clusters *minimize the cluster angular and cluster delay spreads*, which is intuitive and leads to relative compact clusters.

4.4 Cluster validity index — CombinedValidate

For determining the optimum number of clusters, K_{opt}, (cluster validity) we used a combination of two methods [23] well-known in literature [26], the Caliñski-Harabasz index and the Davies-Bouldin criterion. Both indices and their proposed combination are described in the following.

4.4.1 Caliñski-Harabasz index

When clustering N MPCs in K clusters, the Caliñski-Harabasz index (CH) is given as

$$\text{CH}(K) = \frac{\text{tr}(\mathbf{B})/(K-1)}{\text{tr}(\mathbf{W})/(N-K)} ,$$

which corresponds to the ratio between the traces of the *between-cluster scatter matrix* $\mathbf{B}$ and the *within-cluster scatter matrix* $\mathbf{W}$ [26]. Using the

MCD as distance function, tr($\mathbf{B}$) and tr($\mathbf{W}$) are respectively given as

$$\mathrm{tr}(\mathbf{B}) = \sum_{k=1}^{K} N_k \cdot \mathrm{MCD}(\mathbf{c}_k, \overline{\mathbf{c}})^2 \ ,$$

$$\mathrm{tr}(\mathbf{W}) = \sum_{k=1}^{K} \sum_{j \in \mathcal{C}_k} \mathrm{MCD}(\mathbf{x}_j, \mathbf{c}_k)^2 \ ,$$

where N_k denotes the number of MPCs related to the kth cluster and

$$\overline{\mathbf{c}} = \frac{\sum_{l=1}^{N} (P_l \cdot \mathbf{x}_l)}{\sum_{l=1}^{N} P_l}$$

denotes the global centroid of the entire data set.
If we calculate the CH index for different values of K, e.g. in the range $[K_{\mathrm{min}},\ K_{\mathrm{max}}]$, the number of cluster K_{CH} corresponding to the best partition is achieved as

$$K_{\mathrm{CH}} = \arg\max_{K} \{\mathrm{CH}(K)\} \ , \tag{4.11}$$

corresponding to the partition with the most compact and separated clusters.

4.4.2 Davies-Bouldin index

The Davies-Bouldin index (DB) is a function of *intra-cluster compactness* and *inter-cluster separation* [26]. Using the MCD, the *compactness* S_k within the kth cluster is given as

$$S_k = \frac{1}{N_k} \sum_{l \in \mathcal{C}_k} \mathrm{MCD}(\mathbf{x}_l, \mathbf{c}_k),$$

and the *separation*, i.e. the distance, between two clusters i and j, is defined as

$$d_{ij} = \mathrm{MCD}(\mathbf{c}_i, \mathbf{c}_j) \, .$$

Finally the considered DB index is given as

$$\mathrm{DB}(K) = \frac{1}{K} \sum_{k=1}^{K} R_i \ ,$$

where

$$R_i = \max_{\substack{j=1,\dots,K \\ j \neq i}} \left\{ \frac{S_i + S_j}{d_{ij}} \right\} \ .$$

When calculating the DB index for different values of K, the optimum number of cluster K_{DB}, corresponding to the best partition, is achieved as

$$K_{\mathrm{DB}} = \arg\min_{K}\{\mathrm{DB}(K)\} \ .$$

As for the CH index, also the DB index bases on seeking for the partition with the most compact but separated clusters.

4.4.3 CombinedValidate

The basic idea of the CombinedValidate (CV) index [23]is to restrict valid choices of the optimum number of clusters by a threshold set in the DB index. Subsequently the CH index is used to decide on the optimum number out of the restricted set of possibilities.
We consider the set of feasible choices $\mathbf{F} = \{K_1, \ldots, K_C\} \subseteq [K_{\min},\ K_{\max}]$ containing only the values K_i for which the following condition is satisfied,

$$\mathrm{DB}(K_i) \leq t \cdot \min_{K}\{\mathrm{DB}(K)\} \, ,$$

where we chose $t = 2$. The optimum number of clusters K_{opt} is then obtained as

$$K_{\mathrm{opt}} = \arg\max_{K\in\mathbf{F}}\{\mathrm{CH}(K)\} \ .$$

In the unrestricted case $\mathbf{F} \equiv [K_{\min},\ K_{\max}]$ we obtain K_{opt} using (4.11).

4.5 Cluster pruning — ShapePrune

After successfully finding the optimum number of clusters, we use the ShapePrune cluster pruning algorithm for discarding outliers. This is achieved by removing data points that have largest distance from their own cluster centroid. As a constraint, cluster power and cluster spreads must not change significantly. This last condition allows to preserve the clusters' original power and shape, which is fundamental to achieve consistent results. In the following we describe the resulting algorithm.

ShapePrune algorithm:

```
1. Initialize pruned result set with optimum set: $\mathcal{R}^{(p)} = \mathcal{R}_{\mathrm{opt}}$
2. For $k = 1$ To $K_{\mathrm{opt}}$
    a. Save the original power and spread of the $k$th cluster:
       $P_k^{(0)} = \sum_{j\in\mathcal{C}_k} P_j$,
       $\mathbf{S}_k^{(0)} = [\sigma_\tau,\ \sigma_{\varphi_{\mathrm{AoA}}},\ \sigma_{\varphi_{\mathrm{AoD}}},\ \sigma_{\theta_{\mathrm{AoA}}},\ \sigma_{\theta_{\mathrm{AoD}}}]^T$
    b. While $P_k^{(\mathrm{cur})} > p \cdot P_k^{(0)}$ And $\mathbf{S}_k^{(\mathrm{cur})} > s \cdot \mathbf{S}_k^{(0)}$, remove the MPC with
       largest distance
         – Find MPC with largest distance to current centroid $\mathbf{c}_k$
         – Remove MPC from $\mathcal{R}^{(p)}$
         – Recalculate $P_k^{(\mathrm{cur})}$ and $\mathbf{S}_k^{(\mathrm{cur})}$.
    c. Restore the last deleted MPC
    d. Next $k$
3. Return $\mathcal{R}^{(p)}$
```

For each cluster, the algorithm discards the MPCs with the largest distance to the cluster centroid, until one of the constraints is not fulfilled. The single steps of the algorithm are described in the following.

Ad 2a) Before starting to prune the kth cluster, the algorithm stores the original values of its cumulative power to $P_k^{(0)}$, and its (vector-valued) cluster spread to $\mathbf{S}_k^{(0)}$, where σ_τ, $\sigma_{\varphi_{\mathrm{AoA}}}$, $\sigma_{\varphi_{\mathrm{AoD}}}$, $\sigma_{\theta_{\mathrm{AoA}}}$, $\sigma_{\theta_{\mathrm{AoD}}}$ denote the rms cluster spreads of delay, and azimuth and elevation angles of departure and arrival of cluster k, respectively.

Ad 2b) Until the power of the cluster and its cluster spreads are below the two respective specified thresholds, the algorithm removes the MPC with largest distance to the centroid from the cluster, where we use the MCD as distance function. We define the power and spread thresholds as a factor p of the original power and factor s of the original cluster spreads, respectively. Since we have to cope with a vector-valued spread, we define the condition $\mathbf{S}_k^{(\mathrm{cur})} > s \cdot \mathbf{S}_k^{(0)}$ to be satisfied, when it holds true for all dimensions separately.

Ad 2c) Since we want the cluster power and spread to be larger than the specified thresholds, we have to restore the last pruned MPC. This implementation simply allows to speed up computation time.

Ad 3) The output of the algorithm is the pruned set of MPCs $\mathcal{R}^{(p)}$. The last step of the clustering framework consists in pruning the achieved partition, by discarding outliers. The main goal of pruning outliers after clustering (*post-clustering pruning*) is to improve result visualization and future cluster tracking, by discarding all that MPCs which are far from their assigned cluster centroid and whose pruning does not affect the partition structure. We want to point out the importance to find a way to preserve the original partition structure, in order to do not destroy the data structure and achieve inconsistent results.

Chapter 5

Results

In this Section we provide a detailed performance analysis of the singular components of the proposed clustering framework. To access the framework performance we used both synthetic data and real measurement data.

5.1 Simulations

Simulations were performed on synthetic data sets generated using the SCM MIMO channel model specified in [27] and implemented by J. Salo et al. [28]. We randomly generated 6 clusters, where each cluster consisted of 8 MPCs in the delay, azimuth-of-arrival and azimuth-of-departure domain. For simplicity we disregard the two elevation of arrival and departure coordinates.

The synthetic scenarios are easier to be clustered than the real-world ones, because synthetic clusters generally appear as pretty well separated MPC clouds. Therefore we employ simulation results to compare the performance of different distance metrics and validity indices, making a first distinction between "promising" solutions and "not promising" ones. Nevertheless, the ultimate test of the framework shall be performed using real world measurement data as it will be shown in Section 5.2.

5.1.1 Using MCD as a distance function

The MCD distance acts an essential role in the proposed framework, since it efficaciously scales the data, solves the problem of angular periodicity and finally enables for joint clustering. In this Section we present a detailed analysis of the performance improvement of the clustering algorithm, which is achievable due to the introduction of the MCD metric.

To access the performance of the MCD we compare this new metric with the two following distance metrics, previously employed in the clustering business:

1. the *squared Euclidean distance*,
2. the *joint squared Euclidean distance.*

Squared Euclidean distance The squared Euclidean distance (SED) for the delay domain is given as

$$d^2_{\tau,ij} = (\tau_i - \tau_j)^2 \ , \tag{5.1}$$

between any two MPCs i and j.
To cope with angular periodicity we use the metric extension for AoAs and AoDs, provided in [17], according to

$$d^2_{\mathrm{AoA/AoD,ij}} = \mathrm{pv}^2(\varphi_i - \varphi_j, \pi) \ , \tag{5.2}$$

where $\mathrm{pv}(\cdot, \pi)$ maps the firs angular argument to its principal value in the interval $[-\pi, \pi]$. The metric reads similar for the AoDs.
Since SED does not scale the data it cannot be use for joint multidimensional clustering.

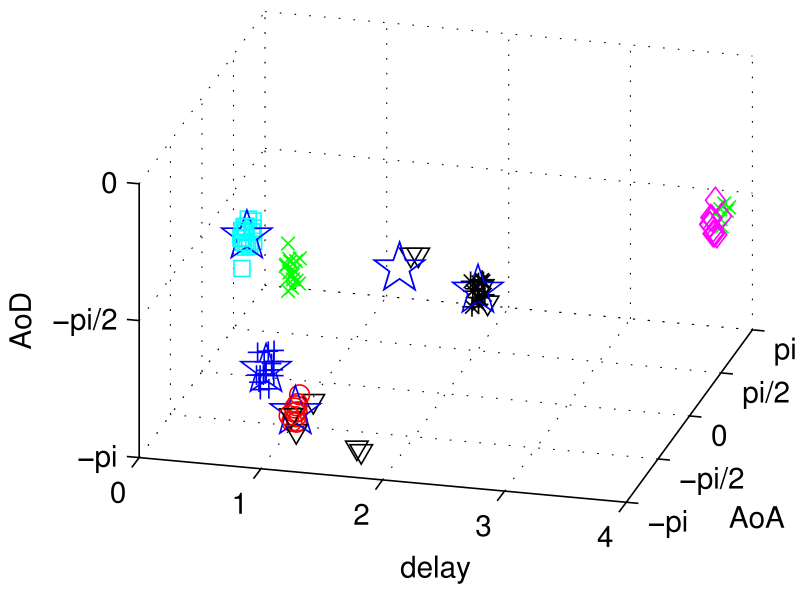
Joint squared Euclidean distance The joint squared Euclidean distance (JSED), briefly mentioned in [17] for the first time, is a straight forward extension of SED with normalized delays. Since we used the $\Delta\tau_{\mathrm{max}}$ normalization, the new delay distance is given by

$$d^2_{\mathrm{JSED}_{\tau,ij}} = \frac{(\tau_i - \tau_j)^2}{\Delta\tau^2_{\mathrm{max}}} \ . \tag{5.3}$$

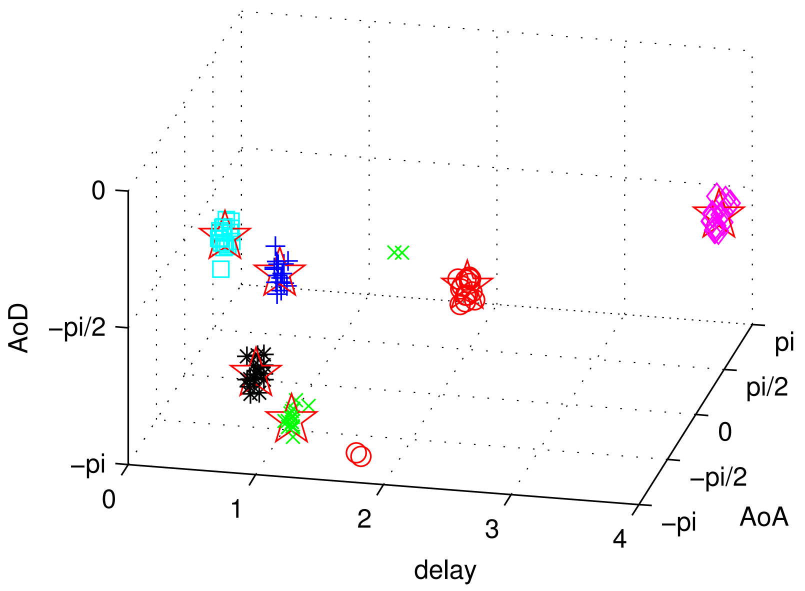
The angular distances are the same defined in (5.2). The introduced scaling makes the delay and angular distance assume comparable values (i.e. same size order), which is essential for enabling joint multidimensional clustering, as explained in Section 4.2.

Performance evaluation First we demonstrate visually that the new MCD metric outperforms the previously used SED. Since the simulation purpose is to compare the MCD metric with the SED metric and not to test the global clustering framework, we assume to know the effective number of clusters, without affecting the comparison result.
We use a synthetic data set, which is chosen such that there occur angular ambiguities as well as closely-spaced clusters. Then we cluster the data set using the hierarchical tree algorithm and we generate partitions of 6 clusters (which is the number of cluster occurring in the data set).
Figure 5.1(a) shows results from *sequential clustering* using the SED. MPCs are shown by colored markers, where MPCs with the same color and style were partitioned to be in the same cluster. The big stars indicate the cluster centroids. MPCs indicated by black triangles were estimated to be in any cluster. It is immediate to see that sequential clustering using SED leads to wrong clustering decisions. Some path are not chosen to be in some cluster at all, other paths are placed into utterly wrong clusters. Even though the SED metric should be able to resolve the angular periodicity, it does not work in practice.
Figure 5.1(b) shows the results from *joint clustering* using the MCD metric. Here the all MPCs are estimated to belong to the correct clusters and the angular periodicity is resolved (clusters indicated by green crosses and red circles). The performance of the applied clustering algorithm are thus significantly improved.
This simulation proves also that joint clustering leads to better partitions if compared with sequential clustering.

Because every distance metric performance is strongly dependent on the cluster angular spreads, we compare the three tested metrics performance using many synthetic scenarios with different cluster angular spreads. This can be done by adding white Gaussian noise, with variances of $\{1^\circ, 2^\circ, \ldots, 10^\circ\}$, to the random MPCs placement process in the AoA and AoD domains. We simulated at least 200 random channels for each cluster angular spread. The results of this evaluation is shown in Figure 5.2.
The sequential clustering using the SED can only be used for very small cluster angular spreads, where it still performs worst of all compared methods, since the performance decreases strongly for larger cluster spreads. Joint clustering using the JSED metric works reasonably better, but the performance deteriorates at larger cluster spreads, due to an unsuitable scaling of the delays.
The MCD shows the best performance for arbitrary cluster sizes. The number of incorrectly clustered MPCs decreases only slightly for larger cluster



(a)



(b)

Figure 5.1: Exemplary synthetic scenario demonstrating clustering results. (a) Sequential clustering using the SED metric. (b) Joint clustering using the MCD metric.

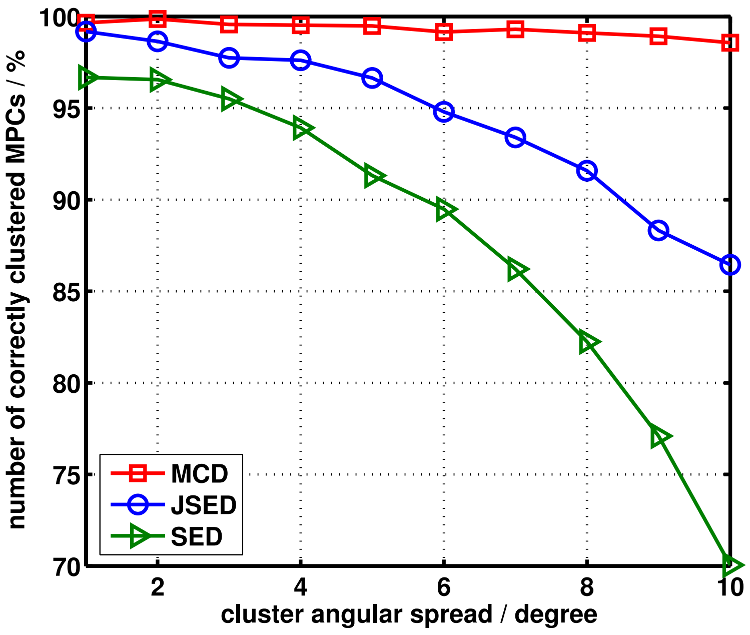


Figure 5.2: Number of correctly clustered MPCs for different cluster angular spreads simulated with the SCM model.

spreads. This can be due to the way the 6 cluster data set is generated. Since the 8 MPCs per cluster are randomly placed in the 3-dimensional space, around their own centroid, increasing the cluster angular spread too much may lead to a wrong placement of the MPCs, which may be closer to a different centroid than to the centroid effectively assigned.

5.1.2 CombinedValidate

In this section we demonstrate the proposed CombinedValidate index outperforms the Caliñski-Harabasz index and the Davies-Bouldin index, by means of their combination. We also demonstrate the CV index outperforms the mean silhouette value, which is employed in the reference semi-automatic clustering algorithm.

Mean silhouette value The silhouette value is a measure of how close a point is to other points in its own cluster compared to points in other clusters [17]. Denoting with $\overline{d}_i(m)$ the average distance of the i-th MPC to the all MPCs in the m-th cluster, the silhouette value for the i-th MPC is defined as [15, 17]

$$s_i = \frac{b_i - a_i}{\max\{a_i, b_i\}} \tag{5.4}$$

where $a_i = \overline{d}_i(p)$ is the average distance of the i-th MPC to all other MPC in its own cluster, say the p-th, and

$$b_i = \min_{\mathcal{K}_q \neq \mathcal{K}_p} \overline{d}_i(q) \ ,$$

is the average distance to a cluster, whose points' average distance to the i-th MPC is smallest over all clusters with $q \neq p$.
The mean silhouette value S is defined as the average of silhouette values over all MPCs

$$S = \frac{1}{N} \sum_{i=1}^{N} s_i. \tag{5.5}$$

The number of cluster that maximizes S is taken as the optimal number of clusters K_{opt}.

Performance evaluation We tested the performance of the cluster validity scheme using the 6 clusters synthetic scenario for different cluster angular spreads. We simulated 200 random channels for each cluster angular spread. Figure 5.3 demonstrates the performance of the different cluster validation indices, i.e. the novel CombinedValidate, the Caliñski-Harabasz, and the Davies-Bouldin index and the mean silhouette value. The Figure shows the fraction of the correctly estimated number of clusters versus the cluster angular spreads. The mean silhouette value decreases with larger cluster spreads and achieves a probability which always remains below the 70%. The DB index decreases with larger cluster spreads too, remaining always below the mean silhouette curve. On the other hand, the CH index has a probability to find the correct number of cluster noticeably higher than the corresponding one of the mean silhouette, but only for cluster spreads approximately higher than 4.5° and it has deep troubles with lower cluster spreads. The CombinedValidate index always outperforms the CH index and outperforms the DB index and the mean silhouette for cluster angular spreads larger than 2.5°.

5.2 Clustering real measured data

Since the goal of the proposed clustering framework is to cope with real-world MPC clusters, the ultimate test we conduct in evaluating its performance

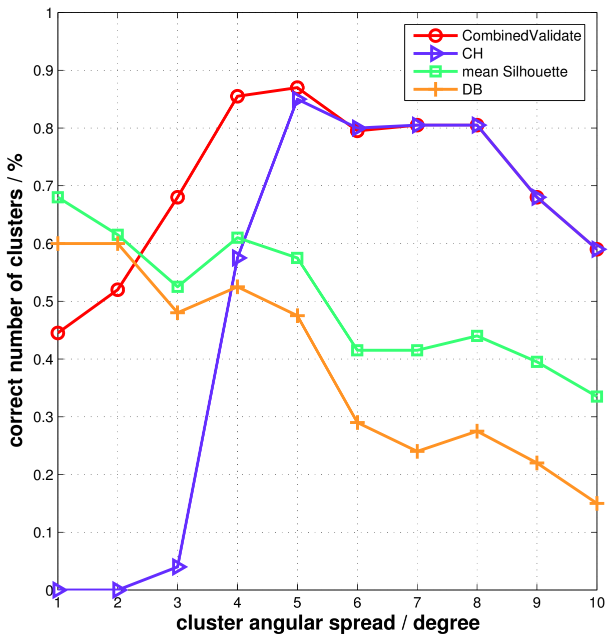


Figure 5.3: Comparing performance of validity indices.

employs real-world measurement data.
In the following, in Section 5.2.1 we describe the measurement equipment[1] and in Section 5.2.2 we present the measurement scenario and the two selected testing data sets. Finally in Section 5.2 we show the framework performance when clustering the selected data sets.
More detailed information about the measurement campaign is provided in [1].

5.2.1 Equipment

In this study the wide-band radio channel sounder *PropSound CS* was applied. This CS utilizes periodic pseudo-random binary signals (PRBS) and consists of a dedicated transmitter and receiver pair with appropriate analysis software. The sounding signal is based on chip sequences of typical spread spectrum signals (M-sequences) with adjustable code lengths. Multi-antenna sounding is based on time-division multiplexed switching of transmit and receive antennas. Thus, sequential radio channel measurement between all possible transmit (Tx) and receive (Rx) antenna pairs is achieved by antenna switching at both the transmitter and the receiver. Antenna switching makes

[1]The measurement equipment was generously provided by Elektrobit Testing Ltd

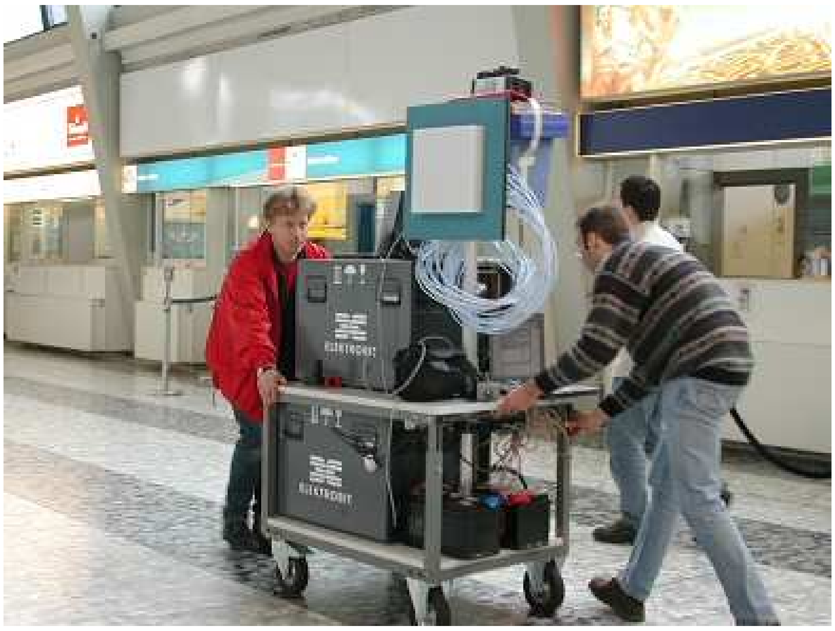


Figure 5.4: Measurement equipment.

the sounder especially suitable for MIMO radio channel studies which incorporate usually a large number of antennas [22]. The measurement equipment is shown in Figure 5.4.
The measurements were conducted at a center frequency of 2.45 GHz with a null-to-null bandwidth of 200 MHz. As Tx array, an omnidirectional uniform circular monopole array with 7 antennas (equidistantly spaced half a wavelength from each other on a circle), plus one additional center antenna was used. For the Rx a 4×4 vertically polarized patch array, where the antennas were spaced by half a wavelength [22], was used. Both the Tx and the Rx are shown respectively in Figure 5.5 and Figure 5.6.
The analysis software includes the SAGE algorithm which was used to estimate the channel parameters, i.e. complex path weights, delays, AoA and AoD [29].

5.2.2 Scenario

The measurements were conducted at the check-in zone of Vienna International Airport during busy hour, i.e. a large-hall indoor scenario. A map of the measurement scenario is shown in Figure 5.7. While conducting the measurements, people were moving, making the radio channel non-stationary.
In a post-processing step parametric channel estimates were gained using the SAGE algorithm [5]. For the following evaluations we make reference to two

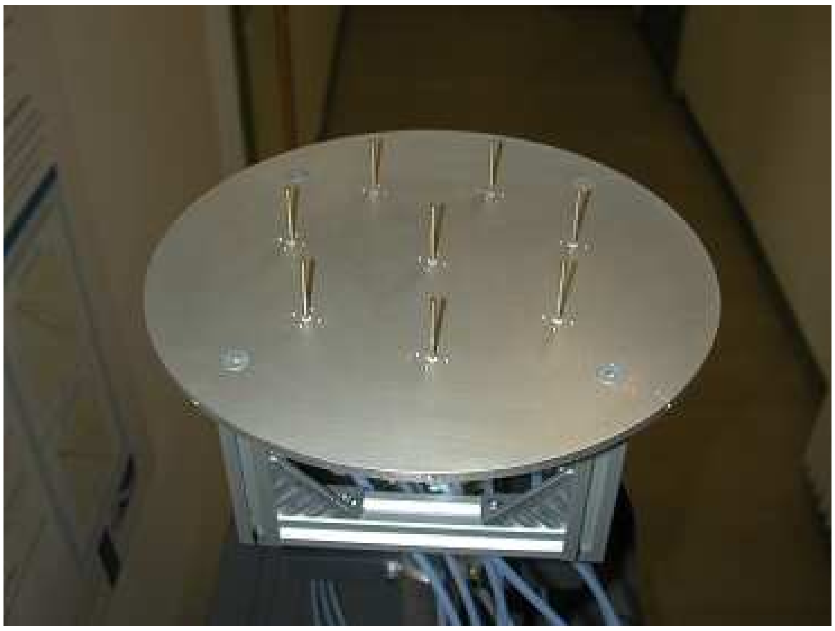


Figure 5.5: The $7 + 1$ omnidirectional uniform circular monopole Tx array.

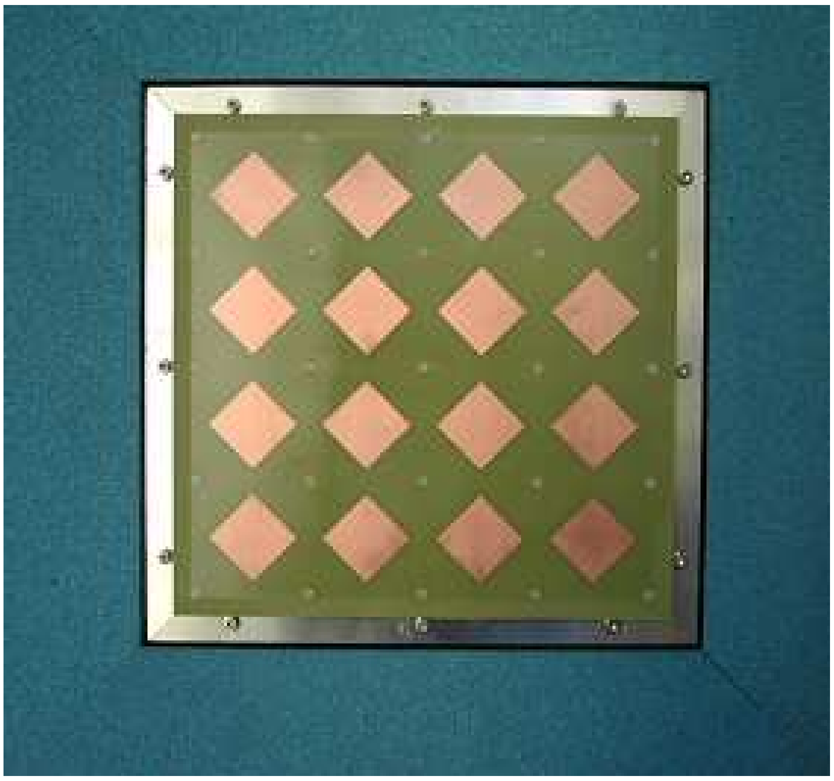


Figure 5.6: The 4×4 vertically polarized patch Rx array.

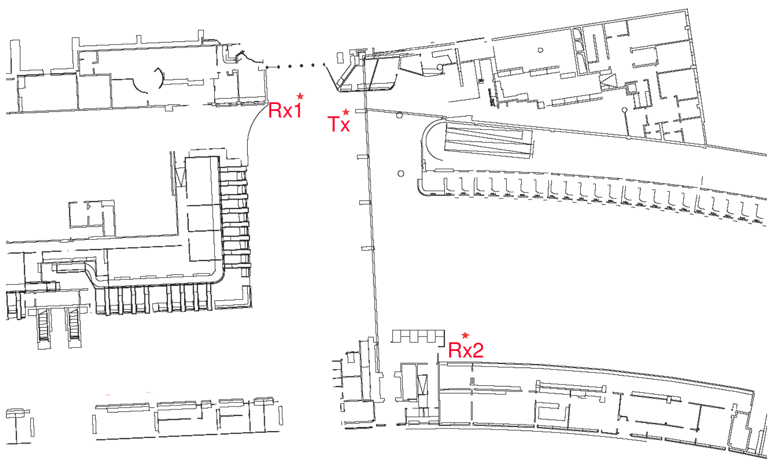


Figure 5.7: Measurement scenario: Vienna International Airport, check-in hall.

exemplary data sets, each corresponding to a different measurement position. MPCs are color-coded with their power in Db.

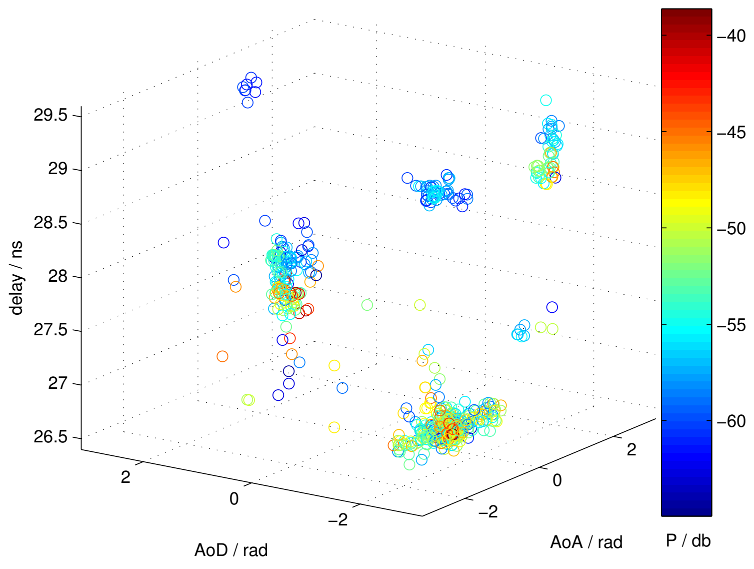
Data set 1 - Rx (Rx1, LOS) with line-of-sight to the Tx. The estimated MPCs show a visible cluster structure, as shown in Figure 5.8(a).

Data set 2 - Rx (Rx2, NLOS) with no-line-of-sight to the Tx. The estimated MPCs do not show a clear visible cluster structure, as shown in Figure 5.8(b).

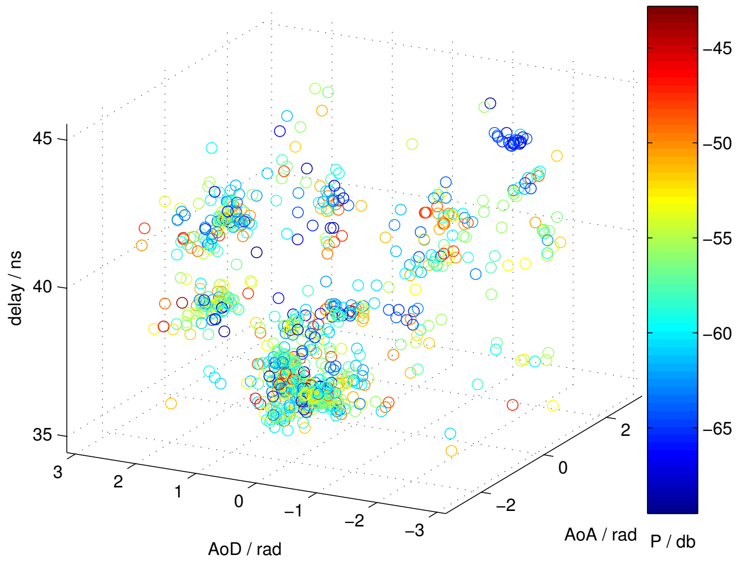
In both Data set 1 and Data set 2 we have considered a number of 100 snapshots of the channel, corresponding to a total channel observation time of 130 ms. All estimated path from these snapshots were considered in the clustering algorithm [22].

5.2.3 Output results when clustering Data set 1

KPowerMeans + CombinedValidate Applying our clustering framework without user interaction (yet without pruning) to this data, we obtain the result depicted in Figure 5.9. The resulting partition into seven clusters realizes the optimum trade-off between cluster compactness and separation. Since the two small groups of MPCs, denoted by purple and light blue color, represent an insignificant contribution to the total power, they are combined with the two larger clusters, represented by the corresponding colors. Surprisingly, the large group of MPCs (around 27 ns), holding most of the total



(a)



(b)

Figure 5.8: The two reference data sets. (a) Data set 1: Rx1 with LOS to the Tx. (b) Data set 2: Rx2 with NLOS to the Tx.

power, is split into four separate clusters. The MPCs powers in this group are strongly varying, cluster centers are attracted by strong powers. Thus, it is most sensible to split up this group into several clusters. Here, the algorithm is able to split up clusters that cannot be seen by visual inspection.

KPowerMeans + CombinedValidate + ShapePrune Using the pruning algorithm, outlier paths are removed. Figure 5.10 shows results of applying the whole framework algorithm including pruning. The clustering algorithm results in well-defined separable clusters. The pruning algorithm improves the visibility without changing cluster parameters and allows for simpler cluster tracking. Clusters can be well identified, they are indicated as MPCs showing the same color.
Obviously, the weak-powered cluster at large delay was pruned. This makes sense as its power did not add much to the channel. Also the large light blue cluster, around 28 ns now looks smaller, but still has similar properties to the original one.

5.2.4 Output results when clustering Data set 2

Data set 2 is more challenging than Data set 1, since it is characterized by the massive presence of outliers. Hence the MPCs are "widely" spread in the delay, azimuth-of-arrival, azimuth-of-departure domain, making the data set do not present any visible cluster structure. By visual inspection it is extremely hard to identify clusters.
The presence of outliers results critical for the clustering framework, which is able to identify only two big distinct clusters, represented in Figure 5.11 by two different colors.
The reason can be found in the employed validity index CombinedValidate. Since this index is based on seeking for the partition which realizes the best trade-off between cluster compactness and cluster separation, the massive presence of outliers makes the task of CombinedValidate extremely hard and leads to a wrong estimation of the number of clusters. Hence further studies are necessary to implement an algorithm, which is able to discard outliers from a given data set before applying the clustering algorithm (*"pre-clustering" pruning*).

In order to prove that is CV which effectively leads to a wrong clustering result, we applied the clustering framework to Data set 2 with a pre-defined number of clusters, which was empirically set to 10. Figure 5.12 and Figure 5.13 show the results of applying KPM (with 10 clusters) and the pruning algorithm.

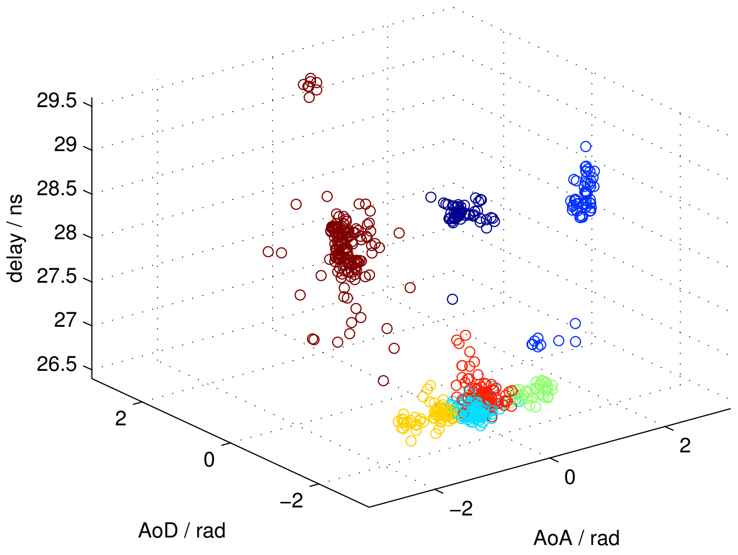


Figure 5.9: Automatically clustered environment without pruning, 7 clusters were identified.

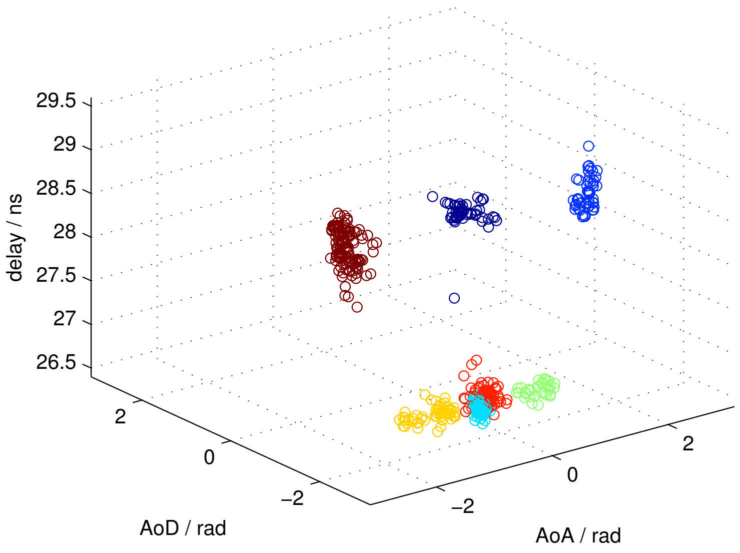


Figure 5.10: Results of clustering Data set 1: weak components were removed.

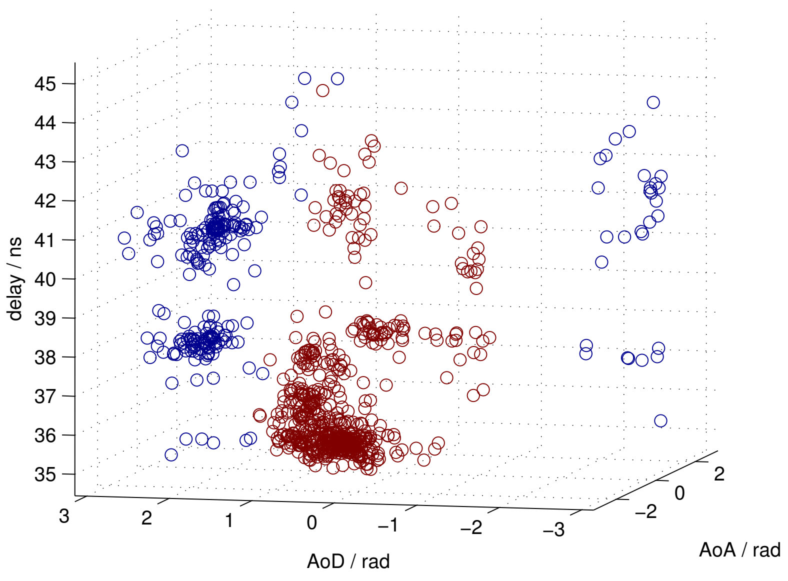


Figure 5.11: Results of clustering Data set2: the framework is able to identify only two clusters.

One can see that, despite of the spread structure of Data set 2, the achieved result is still reasonable. Nevertheless further improvement is expected when clustering a pruned data set.

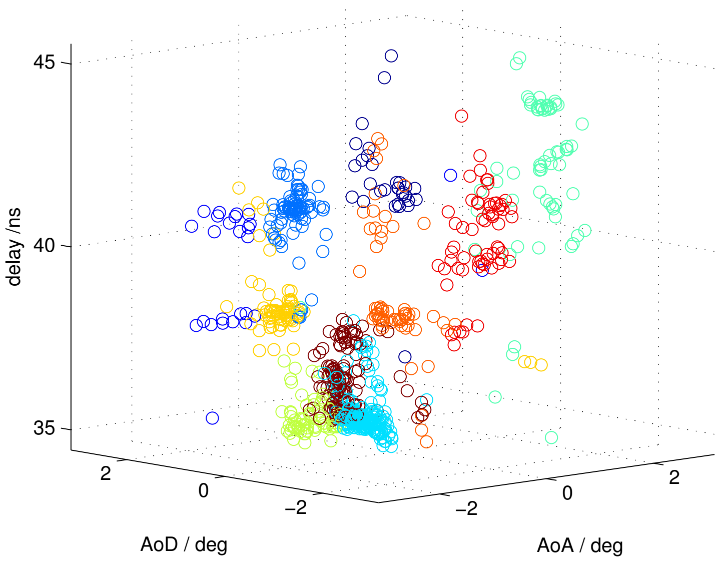


Figure 5.12: Results of clustering Data set 2 when specifying 10 as the number of clusters. MATLAB view corresponding to Az: -44, El: 17.

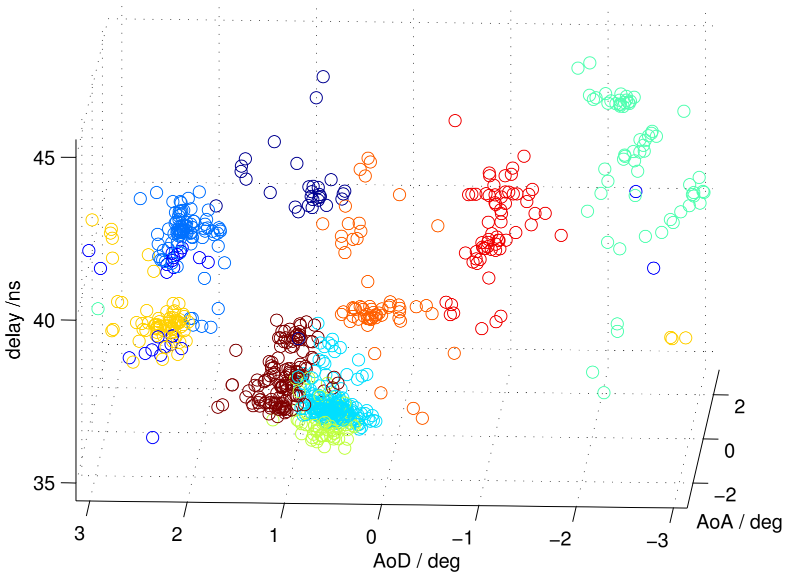


Figure 5.13: Results of clustering Data set 2 when specifying 10 as the number of clusters. MATLAB view corresponding to Az: -87, El: 21.

Chapter 6

Conclusions

Since many advanced MIMO channel models base on the concept of multi-path clusters, the problem of cluster identification is actually one interesting issue in evaluating channel measurements. Alas, currently there is no fully automatic clustering algorithm available to identify clusters from multi-dimensional parametric MIMO channel estimates.
The implementation of an automatic clustering procedure is not a trivial issue, since one has to cope with several practical problems, which are connected to the physical nature of the measured data. The principal problems are the multi-dimensionality and the non-stationarity of the input data and the presence of outliers throughout the data.

The proposed clustering framework In this thesis we presented a scalable framework to automatically identify multi-path clusters in MIMO channel measurement data, which requires only a very limited user interaction. Manual interaction is in fact limited to the definition of the minimum and maximum number of clusters that the framework is able to identify and to the percentage of the original cluster power and cluster spread that the user wants to keep after the pruning process.

The proposed framework is novel in five respects: (i) the framework algorithm enables to cluster MIMO channel parameter estimates automatically with a minimum of user input; (ii) by employing the MCD distance we make it able to scale the data and to solve the problem of the angular periodicity and hence we enable it for joint multidimensional clustering; (iii) by including power and the MCD into the K-means concept, we make it applicable to clustering in propagation research; (iv) the cluster validation provides a trustworthy estimate of the correct number of clusters; (v) the implemented cluster pruning algorithm does not change the cluster behavior significantly, but improves visibility and future cluster tracking performance.

A new cluster definition One important issue still open in literature is to achieve an agreement on a general definition of a cluster. In some previous works clusters have been defined as groups of MPCs with "similar" propagation parameters. This definition is simple, intuitive and it accords to the definition adopted in pattern recognition. Nevertheless, the concept of "similar" is too elusive and imprecise to be adopted as a general definition. In this thesis we showed that the proposed clustering framework introduces a convincing, inherent definition of a cluster itself.

> *For a given number of clusters, clusters are chosen such that they minimize the total distance from their centroids.*

This implies that clusters *minimize the cluster angular and cluster delay spreads*, which is intuitive and leads to relative compact clusters.

Performance evaluation We evaluated the performance of the proposed framework and its singular components with both, synthetic and real-world MIMO channel data. We could demonstrate that the employed MCD metric and the novel CombinedValidate validity index outperform the previous solutions, formerly employed in clustering MPCs. We could also demonstrate that the global framework is even able to outperform visual inspection. Nevertheless we showed that the framework is not able to achieve a correct estimation of the number of cluster in presence of too many outliers and this leads to a wrong final output.

Possible further direction of research Further improvements of the framework may include the development of an algorithm which would be able to prune the outliers, before applying the clustering algorithm, and the study of a validity index which would be more robust in the presence of outliers.

In order to cope with non-stationary MIMO channel estimates, future research should also include the development of a cluster tracking algorithm. Since the ultimate test to prove the existence of the cluster identified by the framework is to map them to real-world physical scatterers, the development of a tracking algorithm would be extremely useful to test the consistence of the clustering algorithm, by comparing the movement of each cluster to the movement of the associated scatterer.

Bibliography

[1] M. Herdin, “Non-stationary indoor MIMO radio channels,” Ph.D. dissertation, Institut für Nachrichtentechnik und Hochfrequenztechnik, Technische Universität Wien, August 2004.

[2] H. Özcelik, “Indoor MIMO channel models,” Ph.D. dissertation, Institut für Nachrichtentechnik und Hochfrequenztechnik, Technische Universität Wien, Dicember 2004.

[3] A. Molisch, “Modeling the MIMO propagation channel,” *Belgian Journal of Electronics and Communications*, no. 4, pp. 5–14, 2003.

[4] L. Correia, *Towards Mobile Broadband Multimedia Networks, COST 273 Final Report*, to be published by Elsevier in 2006.

[5] B. H. Fleury, M. Tschuddin, R. Heddergott, D. Dahlhaus, and K. I. Pedersen, “Channel parameter estimation in mobile radio environments using the SAGE algorithm,” no. 3, pp. 434–450, 18 1999.

[6] K. Yu, Q. Li, D. Cheung, and C. Prettie, “On the tap and cluster angular spreads of indoor WLAN channels,” in *Proceedings of IEEE Vehicular Technology Conference Spring 2004*, Milano, Italy, May 17–19, 2004.

[7] C.-C. Chong, C.-M. Tan, D. Laurenson, S. McLaughlin, M. Beach, and A. Nix, “A new statistical wideband spatio-temporal channel model for 5-GHz band WLAN systems,” *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 2, pp. 139 – 150, Feb. 2003.

[8] J. Salo, J. Salmi, N. Czink, and P. Vainikainen, “Automatic clustering of nonstationary MIMO channel parameter estimates,” in *ICT’05*, Cape Town, South Africa, May 2005.

[9] M. Steinbauer, H. Özcelik, H. Hofstetter, C. Mecklenbräuker, and E. Bonek, “How to quantify multipath separation,” *IEICE Transaction on Electronics*, vol. E85, no. 3, pp. 552–557, March 2002.

[10] M. Kurella, L. Hsiao, T. Yoshida, J. D. Randall, G. Chow, S. S. Sarang, R. V. Jensen, and S. R. Gullans, "DNA microarray analysis of complex biologic processes," *Journal of the American Society of Nephrology*, no. 12, 2001.

[11] J. Han and M. Kamber, *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers, 2001.

[12] B. Mirkin, *Mathematical Classification and Clustering.* Kluwer Academic Publishers, 1996.

[13] S. Guha, R. Rastogi, and K. Shim, "ROCK: a robust clustering algorithm for categorical attributes," *Proceedings of the IEEE Conference on Data Engineering*, 1999.

[14] R. O. Duda, P. Hart, and D. G. Stork, *Pattern Classification.* John Wiley & Sons, Inc., 2001.

[15] L. Kaufman and P. J. Rosseeuw, *Finding Groups in data: an introduction to cluster analysis.* John Wiley & Sons, 1990.

[16] *Statistics Toolbox User's Guide, Version 7.* The Mathworks, Inc. 2005.

[17] J. Salo, J. Salmi, and P. Vainikainen, "Practical experiences for semi-automatic clustering of nonstationary multidimensional radio channel data," in *COST 273 TD(05)069*, Bologna, Italy, Jannuary 2005.

[18] R. Xu and D. Wunsch II, "Survey of clustering algorithms," *IEE Transanction on Neural Networks*, vol. 16, no. 3, May 2005.

[19] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: Part I," *ACM SIGMOD Record*, vol. 31, no. 2, June 2002.

[20] ——, "Cluster validity methods: Part II," *ACM SIGMOD Record*, vol. 31, no. 3, September 2002.

[21] S. Cherednichenko, "Outlier detection in clustering," Master's thesis, University of Joensuu, Gennary 2005.

[22] N. Czink, P. Cera, J. Salo, E. Bonek, J. Nuutinen, and J.-P. Ylitalo, "Automatic clustering of MIMO channel parameters using the Multi-Path component distance measure," in *Proceedings of WPMC*, Aalborg, Denmark, September 2005.

[23] N. Czink and P. Cera, "A novel framework for clustering parametric mimo channel data including mpc powers," in *COST 273 TD(05)104*, Lisbon, Portugal, November 2005.

[24] H. M. El-Sallabi, L. Vuokko, and P. Vainikainen, "Characterization of dissimilarity between multipath components at mobile station in microcellular environment," in *Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE*, vol. 2, March 2004, pp. 1133–1137.

[25] Y.-F. Zhang, J.-L. Mao, and Z.-Y. Xiong, "An efficient clustering algorithm," in *Proceedings of the Second Conference on Machine Learning and Cybernetics*, Xi'an, China, November 2003.

[26] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices," *IEEE Transaction on Pattern Analysis and Machine Learning*, vol. 24, no. 12, pp. 1650–1654, December 2002.

[27] *Spatial channel model for Multiple Input Multiple Output (MIMO) simulations (3GPP TR 25.996), v6.1.0*, September 2003, available: www.3gpp.org.

[28] *MATLAB implementation of the 3GPP spatial channel model (3GPP TR 25.996)*, Jannuary 2005, available: www.tkk.fi/Units/Radio/scm/.

[29] B. H. Fleury, M. Tschudin, R. Heddergott, D. Dahlhaus, and K. L. Pedersen, "Channel parameter estimation in mobile radio environments using SAGE algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 3, pp. 430–450, March 1999.