Automatic Clustering of Nonstationary MIMO Channel Parameter Estimates

Jari Salo[†‡], Jussi Salmi[†], Nicolai Czink[‡], and Pertti Vainikainen[†]
[†]Radio Laboratory/SMARAD
Helsinki University of Technology, PO Box 3000, FI-02015-TKK, Finland
Email: {jari.salo, jussi.salmi, pertti.vainikainen}@tkk.fi
[‡]Institute of Communications and Radio Frequency Engineering
Vienna University of Technology, Gusshausstrasse 25/389 A-1040, Vienna, Austria
Email: nicolai.czink@nt.tuwien.ac.at

***Abstract*— Many geometrical channel models with stochastically placed clusters of scatterers have been proposed in literature. A major practical problem related to the parametrization of such models is the identification of scattering clusters from channel measurement data, which is typically multidimensional and nonstationary. Conventionally, visual inspection has been used for the cluster identification. Such an approach may be suitable for short data records, but becomes impractical when a large amount of measurement data has to be analyzed. In this paper, we propose an automatic procedure for finding clusters from an output of a channel parameter estimator, such as SAGE. The algorithm is based on sequential clustering of windowed multipath estimates, and tracking of cluster centroids in consecutive data windows. Visual inspection of the automatically identified multipath clusters is usually still required when processing measurement data. The practical benefit of the present method is that it significantly speeds up the process of cluster extraction with large data records.**

I. Introduction

Several geometric channel models in literature and standards [1]–[4] are based on the concept of stochastically varying clusters of scatterers. These so-called geometry-based stochastic channel models (SGCM) have intuitive physical interpretation and allow flexible embedding of arbitrary antenna patterns into the channel. A major practical problem in identifying the parameters of geometrical channel models is that the statistical properties of the clusters are difficult to extract from measurement data. Only recently results of measurement analysis of multidimensional multipath clusters have started to appear in literature, see e.g. [4]–[9]. In all these papers the identification of clusters has been done manually by visual inspection of measurement data. Obviously such an approach, while applicable to individual case studies, is impractical for analysis of large data records. For instance, the identification of parameters of dynamic[1] channel models based on the SGCM structure requires analysis of large amounts of measurement data to find the salient features of the underlying nonstationarities (assuming of course that such features exist, or can be identified, in the first place).

[1]By 'dynamic' we mean that parameters of the clusters are time-varying within a single simulation run. This is in contrast to quasi-static (block fading) channel models, such as [1], in which the cluster parameters are constant over a single fading block.

To our knowledge, the only measurement-based study on automated clustering with application to radio channel measurement analysis is [10], where a delay domain (1D) clustering is performed using a parametric model for the observed data. Our approach differs from [10] in that we focus on clustering of multidimensional, nonstationary channel parameter estimates that are the output of a channel parameter estimator based on a deterministic signal model.

The main contribution of this paper is a heuristic algorithm for finding clusters in nonstationary, multidimensional channel parameter data. The algorithm is based on windowing the data, processing each window with some clustering algorithm, cluster tracking, and cluster pruning. Our experiments with real measurement data indicate that clustering techniques can significantly speed up the processing of radio channel measurement data. This is demonstrated with a practical example with measurement data.

The paper is organized as follows. In Section II we state the problem. In Section III we describe the proposed clustering method. Section IV present a numerical example with measured data. Section V concludes the paper.

II. The problem set-up

The starting point is that we have a large number of channel parameter estimates obtained from a real-time measurement of a MIMO radio channel with a moving receiver and/or transmitter. The estimates may have been obtained by any channel parameter estimator, such as SAGE [11], multi-dimensional ESPRIT [12], or RIMAX [13] [14]. It has been noted in several studies that the parameter estimates tend to appear in clusters, i.e., in groups that have similar delays and angular parameters. The problem is to find an automatic procedure to recover and track these clusters from the estimator output data. If the data record is short and/or neither the receiver, the transmitter, nor the environment are moving, the data can be considered stationary, and multipath clustering can be carried out by visual inspection. When this is not the case, finding the clusters may require a frustrating amount of manual work; this observation is the key motivator for this study.

The input data to the clustering algorithm is an $L \times P \times N$ array, where L is the number of estimated multipaths (model

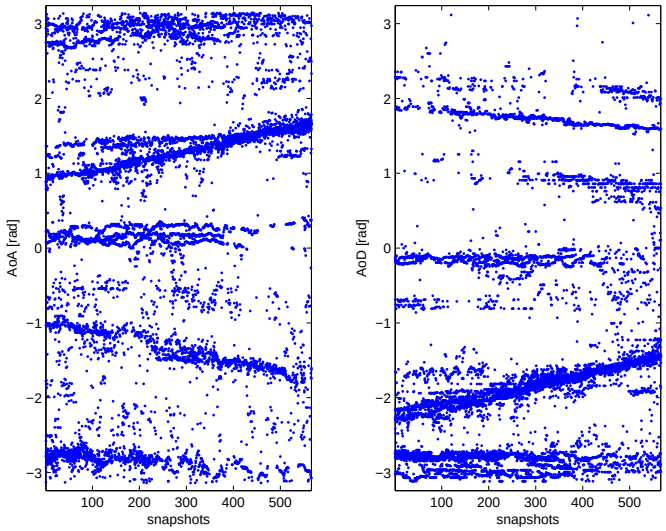

Fig. 1. An example of AoA and AoD estimates over a measurement route. The number of parameters estimated (model order) per channel snapshot is 20.

order), P is the number of estimated channel parameters, and N is the number of observations, or channel snapshots. Typically the dimensions of P are delay, azimuth angle of arrival (AoA), elevation angle of arrival, azimuth angle of departure (AoD), and elevation angle of departure.

Practical problems in clustering of channel parameter estimate data are manifold. We list the most prominent ones.

- The data is nonstationary. 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. Since the measurement equipment is moving, even the relatively stable clusters are not static, but move in the P-dimensional parameter space.
- The data is multi-dimensional and its different dimensions are measured in different measurement units and/or different scales. For example, delays and AoAs are measured in seconds and radians, respectively. This complicates cluster analysis.
- Data outliers are everywhere in the data. For example, plotting AoA and AoD estimates (Fig. 1) from a measurement route illustrates this "data scintillation" phenomenon. An outlier path estimate may be, for example, a result of randomly appearing multipaths with short life-times, phantom estimates due to estimation noise or convergence to local minima in the nonlinear optimization of the parameter estimator.
- 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 identify automatically by mathematical algorithms.

In the next section we propose a procedure that tries to cope with these problems.

III. The Proposed Clustering Algorithm

The overall clustering algorithm consists of several blocks (Fig. 2). In the following subsections we describe these blocks. We start with an overall description.

A. An overview of the algorithm

The basic idea of the algorithm is to cluster the non-stationary data in a sliding window whose width is chosen such that the data within each window can be considered stationary. From the clustering algorithm's point of view it is not necessary to require strict stationarity in the statistical sense as long as the inherent cluster-like structure of the multipath data is preserved over the data window.

A commonly occurring problem in cluster analysis of multidimensional data is the scaling of data when different coordinates are measured in different units. With channel multipath data the dimensions are typically delay (seconds), angles of arrival and angles of departure (radians). Suitable scaling of these dimensions is required for undistorted distance computation. Another twist in the problem is that one cannot simply use one of the usual distance metrics for measuring distances in these dimensions, since angles are periodic variables whereas delays are not. To overcome these inconveniences we propose using sequential processing of the parameter domains.

Sequential processing of parameter domains: Instead of processing all data dimensions jointly, it is also possible to cluster them sequentially, i.e., one subset of dimensions at a time. In practice this means that first one finds clusters in one or more dimensions, and then processes the discovered clusters one-by-one in the remaining dimensions, either jointly or sequentially. This approach has several benefits. First, it avoids the problem of multidimensional data scaling, since one-dimensional data need not be scaled, or because one can select suitable subsets of the data dimensions where scaling is not required, e.g. receiver and transmitter elevation angle. Second, it is more flexible than processing all dimensions jointly, since one can easily change and/or reconfigure the clustering algorithm between data layers in order to optimize the performance of the overall procedure. In Section IV we apply sequential processing by first 1D clustering the delay domain, then 1D clustering the AoA domain, and finally 1D clustering the AoD domain. In our case, we found the sequential approach convenient since the distance metric can be changed between delay and angular dimensions. Furthermore, sequential processing provides an opportunity for visualizing the data between clustering of different data dimensions for improved control over the overall clustering procedure. A disadvantage of sequential processing is that it is sensitive to errors in the initial processing step.

B. Read next data window (S1)

If processing the first data window, select a data points window with predefined width ($\leq N$). Otherwise, slide the

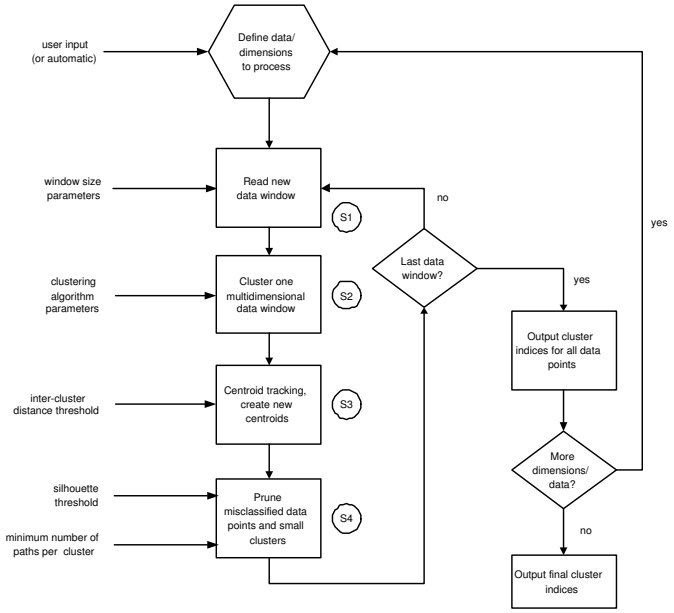

Fig. 2. Block diagram of the clustering algorithm for multidimensional nonstationary data. The circled numbers correspond to steps explained in Sections III-B–III-E.

data window forward by predefined length, i.e., read new data points and drop out oldest ones on first-in-first-out basis. Optionally, data scaling can also be done at this point.

C. Clustering algorithms for stationary data (S2)

1) K-means clustering: K-means algorithm iteratively groups the data points so that the sum of distances from points to their own cluster centroid is minimized over all clusters. Mathematically, the algorithm finds K centroids $\mathbf{c}_k(\mathbf{x}_n)$, $k = 1, \ldots, K$, such that

$$J = \sum_{n=1}^{N} \|\mathbf{x}_n - \mathbf{c}_k(\mathbf{x}_n)\|_2^2 \,,$$

is minimized. Here $\mathbf{c}_k(\mathbf{x}_n)$ denotes the centroid, which is closest to the point $\mathbf{x}_n$. The algorithm always converges, but not necessarily to a global optimum. Usually the algorithm needs to be run repeatedly for different initial centroids to improve the probability of finding a global optimum.

K-means requires the number of clusters, K, as an input parameter. To automatically determine the optimum number of clusters with the K-means algorithm we run the algorithm for an increasing number of clusters and choose K so that the mean silhouette value is maximized, as explained in Section III-C.3.

The complexity of the K-means algorithm increases linearly with the number of data points, which makes it well-suited for processing large data mountains.

2) Hierarchical tree clustering: Hierarchical tree clustering, or agglomerative clustering, is an algorithm that constructs a binary cluster tree from the input data. The lengths of the branches of the tree encode distances between the nodes. Each node in the tree represents a group of data points. The leaf nodes are just the data points themselves, and the root node contains all data points. The actual clusters can be visually assessed by plotting the tree in a *dendrogram*, or by seeking structural divisions in data by comparing average distances between links below a node to the distance above it. It is also possible to use the silhouette analysis described in Section III-C.3 by cutting the cluster tree at several different heights (each resulting in different number of clusters), and computing the mean silhouette value for each cut. However, this method is somewhat artificial as tree clustering is a more natural technique for small data samples, where visual inspection of the cluster tree can reveal the inherent structure of data.

The tree clustering differs from the K-means algorithm in that distances between clusters, instead of points, need to be computed. Several *linkage* methods have been proposed for finding the inter-cluster distances. For an introduction to these methods, see [15]. In this study we use the group average linkage, where the inter-cluster distance is obtained as the average of all pair-wise distances between the points in the two clusters. The pair-wise distances, in turn, can be computed with any standard distance metric suitable to the problem at hand. We use the squared Euclidian distance.

The complexity of hierarchical clustering increases quadratically in the number of data points. In our application we need to window the entire data set into smaller subsets, hence this is not a big problem.

We noticed that the tree clustering allows for a convenient means for implementing the centroid tracking procedure required for combining clusters between consecutive data windows, see Section III-D.

3) Choosing the optimum number of clusters: After computing K clusters it is possible to examine how good the clustering result is by computing and plotting the so-called *silhouette values* for all data points. 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. Denoting by $\bar{d}_n(k)$ the average distance of the nth data point to all data points in the kth cluster, the silhouette value for the nth point is defined as [15, p. 85]

$$s_n = \frac{b_n - a_n}{\max(a_n, b_n)} \,,$$

where $a_n = \bar{d}_n(l)$ is the average distance of nth data point to all other data points in its own cluster whose index is denoted by l, and

$$b_n = \min_{k \neq l} \bar{d}_n(k)$$

is the average distance to a cluster, whose points' average distance to point n is smallest over all other clusters with $k \neq l$. The silhouette value is in the interval $[-1, +1]$, where values close to -1 indicate that the point is very likely in the wrong cluster. Points whose silhouette value is close to $+1$ are likely to have been correctly clustered.

With a small data sample one can plot the silhouette values for all clusters to visually inspect how well separated the

clusters are. This can be repeated for varying K to deduce the optimum number of clusters in the silhouette sense. Since we are interested in processing large amounts of data we automate this search by computing the average of silhouette values over all data points for an increasing sequence of the number of clusters, K. We then select a K that maximizes the mean silhouette. The heuristic reasoning is that the optimum number of clusters is chosen such that the clusters are maximally separated [15, p. 96].

D. Centroid tracking (S3)

A problem arising from the sliding window processing of measurement data is that the cluster indices have to be paired between two consecutive windows, since the cluster index numbers assigned by the clustering algorithm are arbitrary. This can be also considered a problem of tracking the drifting of the cluster centroids. Our approach is to treat it as a clustering subproblem:

1) Take all the cluster centroids computed for windows t ('old') and $t+1$ ('new'). The number of the centroids from the two windows is, in general, different. The cluster indices of the data points in the new window are initialized to zero.
2) Use hierarchical tree clustering to cluster the centroids using an inter-cluster distance criterion. The distance criterion determines how close to each other the cluster centroids need to to be in order to be combined into a single cluster. Clusters that satisfy the criterion are combined into one in order to track the movements of the cluster. After this step each cluster consists of data points from the 'old' and/or 'new' window.
3) Assign indices to data points from the 'new' window $(t+1)$ according to the following rule: (i) if all data points in the cluster have a zero cluster index, assign all points an unused index hence creating a new cluster; (ii) If one or more data points in the cluster have a non-zero cluster index, assign all data points the minimum non-zero cluster index present in the cluster.

Provided that the sliding window does not engulf too many new data samples at each step, the cluster centroids drift slowly, and cluster movements can be tracked. A new cluster is created, when a centroid isolated from the old ones is found, and old clusters that drift too close to each other will be combined. An inter-cluster distance threshold parameter determines how close to each other the clusters must be in order to be combined. Selecting a too small parameter value will cause more clusters to be created. Selecting a too large threshold will cause distinct clusters to be combined into one. Hence, the cluster distance threshold can also be considered a kind of resolution adjustment parameter.

E. Cluster pruning (S4)

Parameter estimates may contain stray paths that are a result of sporadically appearing short life-time reflections, or bad multipath estimates due to e.g. measurement noise or convergence to local minima. The clustering algorithm may classify such outlier data points in their own cluster, or combine them to another cluster. We combat this phenomenon by pruning the clusters by

- computing silhouette values for all points and removing points whose value is less than a certain threshold in the range $[-1, 1]$, called silhouette threshold in the sequel. This procedure ideally removes data points that have been misplaced in wrong clusters.
- discarding clusters that have less points (path estimates) than a value determined by a threshold parameter. This step removes entire clusters that can be considered to be a result of a short-term reflection.

The silhouette thresholding is carried out after each processed data window, whereas small clusters can also be discarded after processing all windows, or after processing all parameter domains.

In practice the cluster indices for the discarded points are set to a negative value and they are excluded from any further processing.

IV. Numerical example

In this section we give an example of clustering multipath parameters estimated from real measurement data shown in Fig. 1. Real-world radio channels are far more complex than present radio channel models are able to reproduce. The measurement was conducted in an urban micro cell environment in downtown Helsinki, Finland. The clustering is done for channel parameter estimates, which are computed from raw measurement data by using the parametric channel parameter estimator tool developed by Elektrobit. For each channel snapshot, parameters of 20 multipaths are estimated. There are a total of 566 channel snapshots in the measurement route. From the perspective of the clustering example other details of the measurement are irrelevant. The data is clustered with the sequential 1D+1D+1D clustering scheme in delay, RX-azimuth, and TX-azimuth domains. In the 1D+1D+1D scheme all data are first 1D clustered in delay domain. Then, the data points in each delay cluster are 1D clustered in AoA domain. Finally, each delay/AoA cluster is 1D clustered in the AoD domain. We also tried a 1D+2D scheme, where (1D) delay clustering was followed by a (2D) AoA/AoD clustering of each delay cluster. The differences between the 1D+1D+1D and 1D+2D schemes with the measurement data were insignificant, however. There are several practical problems in applying a full 3D clustering of delay/AoA/AoD dimensions, most notably reliable automatic scaling of the delay and angular (circular) dimensions; suitable scaling of data dimensions is very important for clustering and centroid tracking performance. Further research is required here.

In Fig. 3 we plot six clusters obtained by running the 1D+1D+1D scheme on the same data as in Fig. 1. All data has been plotted in yellow color. No manual processing or cluster editing has been done. The processing parameters in this example are as follows: data window width is 20 channel snapshots, at each step the window is moved 5 snapshots forward, the inter-cluster distance threshold is set to 4 samples

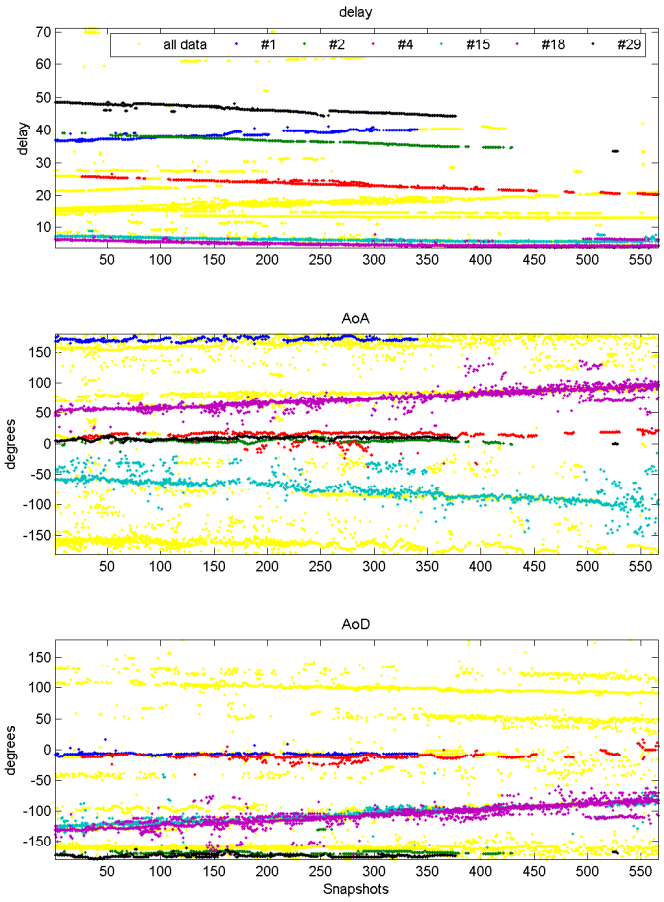

Fig. 3. Six clusters without any manual processing.

and 15 degrees in delay and AoD/AoA domains, respectively. The minimum number of paths per cluster is set to 30 and the silhouette threshold is 0.2. We note that the clusters can be tracked reasonably well. For example, the first arriving delay path has been split to two paths in the AoA domain (Clusters #15 and #18). The algorithm can be seen to recover even when a cluster disappears for a few dozen snapshots. On the other hand, sometimes the centroid tracking fails and loses a cluster. Tuning of window sizes and parameter thresholds may improve the performance for a given data set. Not all data from Fig. 3 have been separated as well (not shown). This is not uncommon with the present algorithm. Typically some manual combining or splitting of clusters is required to achieve acceptable results. This is not surprising considering the ill-behaving nature of real-world measurement data. However, when processing data from large measurement campaigns, an automatic clustering procedure can speed up the cluster extraction process significantly.

V. Conclusion

We have proposed a heuristic algorithm for finding clusters from nonstationary, multidimensional radio channel parameter estimates. The approach is based on sequential clustering of the desired parameter dimensions, and windowing the data. The overall algorithm also includes automatic selection of the number of clusters, centroid tracking and cluster pruning functionalities. Real channel measurement data is very difficult to cluster reliably and with the current algorithm some manual processing of the clustering results may be required for satisfactory results. However, when processing data from large measurement campaigns our automatic clustering tool can save a considerable amount of manual effort and time.

Acknowledgment

The authors benefited from illuminating discussions with Hüseyin Özcelik. Elektrobit is gratefully acknowledged for letting us use their channel parameter estimator tool. The authors would also like to thank Academy of Finland, Foundation of Commercial and Technical Sciences, Nokia foundation, TKK foundation, and Graduate School of Electronics, Telecommunications, and Automation for financial support.

References

[1] "Spatial channel model for Multiple Input Multiple Output (MIMO) simulations (3GPP TR 25.996), v6.1.0," Sep. 2003. [Online]. Available: www.3gpp.org

[2] A. F. Molisch, "A generic model for MIMO wireless propagation channels in macro- and microcells," *IEEE Trans. Signal Processing*, vol. 52, no. 1, pp. 61–71, Jan. 2004.

[3] L. M. Correia, Ed., *Wireless Flexible Personalised Communications*. John Wiley, 2001.

[4] C.-C. Chong, C.-M. Tan, D. I. Laurenson, S. McLaughlin, M. A. Beach, and A. R. Nix, "A new statistical wideband spatio-temporal channel model for 5-GHz band WLAN systems," *IEEE J. Select. Areas Commun.*, vol. 21, no. 2, pp. 139–150, Feb. 2003.

[5] Q. H. Spencer, B. D. Jeffs, M. A. Jensen, and A. L. Swindlehurst, "Measurement and modeling of temporal and spatial indoor multipath characteristics," *IEEE J. Select. Areas Commun.*, vol. 18, no. 3, pp. 347–360, Mar. 2000.

[6] A. S. Y. Poon and M. Ho, "Indoor multiple-antenna channel characterization from 2 to 8 GHz," in *Proc. ICC 03*, vol. 5, May 2003, pp. 3519–3523.

[7] L. Vuokko, P. Vainikainen, and J. Takada, "Clusterization of measured direction-of-arrival data in an urban macrocellular environment," in *Proc. PIMRC 2003*, vol. 2, Sept. 2003, pp. 1222–1226.

[8] F. Mikas, L. Vuokko, and P. Vainikainen, "Large scale behaviour of multipath fading channels in urban macrocellular environments," in *Proc. COST273 10th Management Committee Meeting*, June 9-10, 2004, Gothenburg, Sweden, TD(04)101.

[9] N. Czink, M. Herdin, H. Özcelik, and E. Bonek, "Number of multipath clusters in indoor MIMO propagation environments," *El. Lett.*, vol. 40, no. 23, pp. 1498–1499, Nov. 2004.

[10] D. Shutin and G. Kubin, "Cluster analysis of wireless channel impulse responses with hidden Markov models," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Proc.*, vol. 4, May 2004, pp. 17–21.

[11] B. H. Fleury, M. Tschudin, R. Heddergott, D. Dahlhaus, and K. I. Pedersen, "Channel parameter estimation in mobile radio environments using the SAGE algorithm," *IEEE J. Select. Areas Commun.*, vol. 17, no. 3, pp. 434–450, Mar. 1999.

[12] A. Richter, D. Hampdicke, G. Sommerkorn, and R. Thomä, "MIMO measurement and joint M-D parameter estimation of mobile radio channels," in *Proc. of IEEE 53rd Vehicular Technology Conference*, vol. 1, 2001, pp. 214 –218.

[13] A. Richter, M. Landmann, and R. Thomä, "RIMAX - a flexible algorithm for channel parameter estimation from channel sounding measurements," in *Proc. COST273 9th Management Committee Meeting*, January 26-28, 2004, Athens, Greece, TD(04)045.

[14] R. S. Thomä, M. Landmann, G. Sommerkorn, and A. Richter, "Multidimensional high-resolution channel sounding in mobile radio," in *Proc. IMTC 04*, vol. 1, May 2004, pp. 257–262.

[15] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons, Inc., 1990.