Deterministic walks as an algorithm of pattern recognition

Mônica G. Campiteli,* Pablo D. Batista,† Osame Kinouchi,‡ and Alexandre S. Martinez§
Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto, Universidade de São Paulo, Avenida Bandeirantes, 3900, 14040-901, Ribeirão Preto, São Paulo, Brazil
(Received 28 November 2005; published 4 August 2006)

New tools for automatically finding data clusters that share statistical properties in a heterogeneous data set are imperative in pattern recognition research. Here we introduce a deterministic procedure as a tool for pattern recognition in a hierarchical way. The algorithm finds attractors of mutually close points based on the neighborhood ranking. A memory parameter μ acts as a hierarchy parameter, in which the clusters are identified. The final result of the method is a general tree that represents the nesting structure of the data in an invariant way by scale transformation.

DOI: 10.1103/PhysRevE.74.026703 PACS number(s): 02.70.−c, 02.50.Sk, 42.30.Sy, 07.05.Kf

I. INTRODUCTION

With recent technological advances, the amount of data produced daily in research centers increases dramatically as a function of time. Techniques for exploring this amount of crude, frequently multidimensional data are essential [1–4]. The aim of these works is to identify correlations and patterns in a data set.

In most cases there is no *a priori* knowledge about the data set, thus being indispensable the use of unsupervised methods of clustering. Such methods partition the data set minimizing dissimilarities within and maximizing dissimilarities between the clusters, driven solely by the data. The practical procedure to access this relational array of elements is to set a convenient distance measure that converts the relationships into numerical values.

Thus, grouping data frequently involves some implicit assumptions about the data set structure to better fit the data in the appropriate space. A common sense in data mining research is that there is no optimal approach. Each problem demands its own optimal solution and each technique gives different results and carries different limitations. Therefore, easy-to-implement, fast, and robust techniques are welcome.

Due to their easy implementation and intuitive visualization, hierarchical methods are among the most popular adopted approaches [5–7]. They organize data in a hierarchical structure according to a proximity matrix. Groups are formed by a process of agglomeration or division driven by the minimal pairwise distances. The result is usually a hierarchical binary tree.

Although not as thoroughly studied as random walks [8,9], the study of deterministic walks [10–12] has atttracted the interest of researchers, and new applications are being considered. Following a previous work on this subject, we are interested in exploring a partially self-avoiding deterministic walk algorithm, known as the tourist walk (TW) [13–16], for pattern recognition purposes.

*Electronic address: monicacampiteli@pg.ffclrp.usp.br
†Electronic address: pablofisica@yahoo.com
‡Electronic address: osame@ffclrp.usp.br
§Electronic address: asmartinez@ffclrp.usp.br

The tourist walk algorithm can be pictorially viewed as a tourist subjected to visit N cities randomly distributed on a map, following the deterministic rule of going to the nearest city, that has not been visited in the past μ time steps. The tourist follows his route, passing by a transient trajectory until he or she gets trapped in a cycle of period p. Although easy to formulate, this algorithm presents complex behavior according to the chosen memory window μ. The deterministic nature just described suggests a property of finding natural clusters in a given data set (here we may understand a city as a point in a multidimensional space) based on local interactions among the elements.

Here we propose the use of the deterministic tourist walk algorithm to identify patterns in data sets in a hierarchical way. In Sec. II we briefly review the hierarchical method most commonly used (the single linkage hierarchical clustering) and present the tourist walk algorithm as a clustering method, together with the procedure developed for visualization—a general tree. In order to evaluate the performance and to illustrate the method better, we have used two artificial data sets and one set of real-world data. The description of these test sets is given in Sec. III. In Sec. IV we present and discuss the results of the applied TW method, outlining some of its features in contrast to the single linkage hierarchical clustering results. In Sec. V we summarize and present the final conclusions.

II. THE ALGORITHMS

Here we briefly review the traditional hierarchical method and introduce one based on the deterministic tourist walk.

A. Hierarchical methods

Traditional hierarchical clustering can be classified into two categories: the agglomerative methods and the divisive methods, corresponding to bottom-up and top-down strategies, respectively [3,4,7]. The agglomerative methods have been used more frequently than the divisive ones and are focused on here.

In agglomerative methods, each element is signed to its own cluster and the distance matrix is computed (the metrics used is up to the convenience of the problem). The essential

feature of this algorithm is that the two closest clusters in this symmetric matrix are merged, resulting in a single point. At each iterative step the distance matrix must be updated and the process continues until there is only one cluster, representing the whole data set.

The result of a hierarchical algorithm is a binary tree representing nested grouping of patterns and similarity levels at which groupings change. A hierarchical tree is a very useful tool for abstractions on the structure of the data set, especially when hierarchical relations really exist in the data. The ultimate clustering result can be obtained by cutting the branches of the resulting tree at different levels of dissimilarity according to the problem.

The definition of "close" for a pair of clusters varies and the application depends on the nature of the data set. One of the most commonly used algorithms is the single linkage. For a review of the main measurements of proximity between two groups of objects, see Refs. [3,4]. For the single linkage method, the distance between two clusters is determined by the two closest objects in different clusters. Nevertheless, all of the algorithms act in the same way, coalescing the pairs of clusters at each level of the hierarchy, imposing a pairlike structure to the data.

B. Tourist walk

Let us now turn our attention to a different approach to the clustering problem.

Consider a set of N elements in a space of d dimensions. For illustrative reason, we call each element a city. Consider a tourist beginning his route in a given city of this set. The tourist moves according to the following rule: go to the nearest city that has not been visited in the last μ time steps (it is worth mentioning that in [13] the memory window is designated by τ, being $\tau=\mu-1$).

For $\mu \geqslant 1$, self-avoidance is limited to this memory window, and trajectories can intersect outside this range. The tourist trajectory consists of a transient part of length t and a final cycle of period p, with $\mu+1 \leqslant p \leqslant N$. For $\mu=0$ (the tourist has no memory), the solution is trivial and every datum point represents a singleton attractor. If $\mu=1$ (remember only the last visit—the actual city), the tourist goes to the nearest city until two reciprocally nearest neighbors are found, entering a two cycle. The full analysis of this situation is given in Ref. [17]. With $\mu=N-1$, the trajectory is totally self-avoiding and the whole set with N points covers an attractor. This particular case is known as the nearest-neighbor construction heuristic. Our interest is devoted to the intermediate cases $(1<\mu<N-1)$, which present complex behavior.

The tourist movements are entirely performed based on the neighborhood table, which neglects the distance among the cities but keeps the neighborhood rank within the memory windows. An example of the tourist walk in a bidimensional random map is shown in Fig. 1.

In the context of clustering, one can consider each attractor at a given value of the parameter μ as a cluster. Increasing μ leads to an increase in the self-avoidance and the clusters tend to coalesce. Notice that clusters have at least $(\mu+1)$ elements contrasting to the pairwise interaction of the traditional hierarchical models.

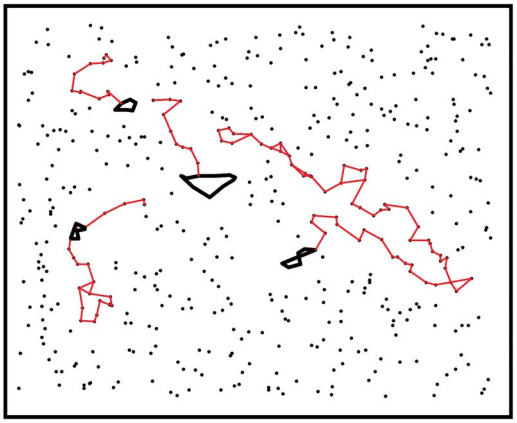

FIG. 1. (Color online) A random map with $N=500$ and $\mu=5$. Beginning at four different points, the tourist performs a transient trajectory (thin line) until falling in an attractor (thick line). The attractors can have different periods.

The visualization of the clustering process is a general tree, similar to the ones used in hierarchical methods, although it is not constrained to a binary or any N-ary structure. The levels in the hierarchy represent the window memory instead of a dissimilarity measure.

The algorithm for building the hierarchical tree works as illustrated in Fig. 2. Figure 2(a) shows a bidimensional map with 20 points randomly (uniformly) distributed. In the construction of the tree [Fig. 2(b)], points that belong to the same attractor are considered as a cluster in the given hierarchy level. For $\mu=0$, each point represents an attractor. Therefore one has N singleton clusters as the leaves of the tree. For $\mu=1$, pairs of mutually nearest neighbors are the new attractors. Notice that this is not equivalent to the single linkage (SL) algorithm since the latter only considers the pair with the smallest distance in each iteration. As the memory values are further increased, the walk is performed and new attractors are found. These attractors are formed independently of the results obtained in the preceding steps. Each new attractor can either contain a part of or a whole of preceding attractors. If overlap occurs, the clusters are merged. This nesting process continues until the whole data set is contained in only one cluster, which usually occurs for memory values much lower than $N-1$ for organized data.

III. DATA SETS

For the numerical experiments, we have created two artificial data sets with the purpose of stressing the main features of the algorithm. The real-world data utilized are the iris data set [18]. The results are confronted with the single linkage hierarchical clustering. The sets are described below.

The construction of the distance matrix (SL) and the neighborhood tables (TW) has been based on the Euclidean distance between the data vectors for the three data sets tested. Ties are resolved arbitrarily,and the one that comes first has priority.

(I) This data set consists of 33 elements in two-dimensional space organized in four well-separated clusters

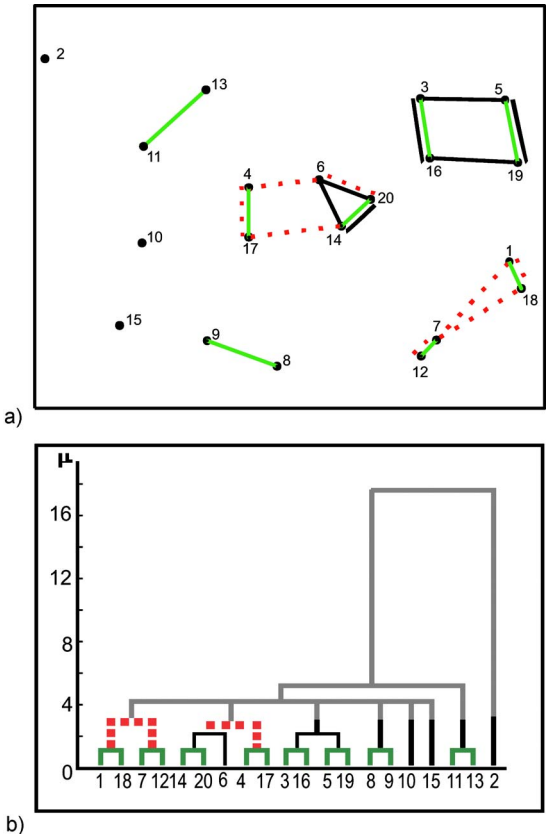

FIG. 2. (Color online) Construction of the hierarchical tree with tourist walk algorithm. (a) Twenty random points in a two-dimensional space. The lines represent the attractors formed in each memory step shown for $\mu=[0,3]$ (solid light line: $\mu=1$; solid dark line: $\mu=2$; dotted line: $\mu=3$). (b) The corresponding tree with the corresponding colors for the drawn attractors.

and one element placed equidistantly from the four other groups. The data are plotted in Fig. 3(a).

(II) This data set consists of 100 elements in two-dimensional space displaced in two clusters. The two clusters are equivalent but plotted in different scales. The larger one is a magnification of ten times the other. The set is shown in Fig. 3(b).

(III) The iris data set [18] is a real data set commonly used as a benchmark for data mining purposes. It consists of 150 samples of iris flowers with the respective measures of length and width for petal and sepal. These flowers are divided into three categories or subspecies (see Fig. 4). A typical multivariate statistical analysis concerns the diagonalization of the measure covariance matrix. The eigenvalues are the principal components and the eigenvectors form the basis for independent measures. This procedure is called the *principal component analysis*. For visualization purposes, the data set is represented only with its two principal components, which are plotted in Fig. 4.

IV. RESULTS

Both SL and TW algorithms have been applied to data sets I to III, resulting in the dendrograms shown in Figs. 5–7, respectively.

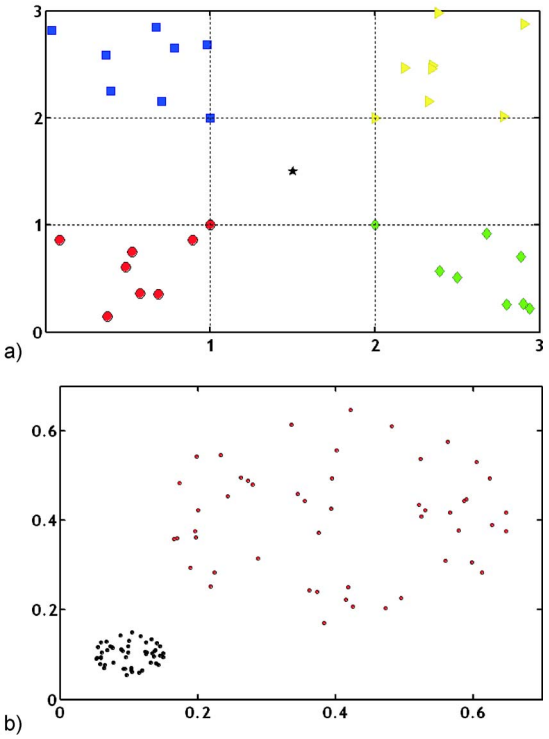

FIG. 3. (Color online) Points distributed in a plane where the distances are in arbitrary units. (a) Data set I, a set of 33 elements distributed in four distinct clusters and one point equidistant of them. (b) Data set II, two equivalent sets of 50 elements each, plotted in different scales.

Figure 5 shows a great similarity in the clustering structure obtained by both methods. However, as discussed above, the linkage method imposes a binary hierarchical structure to the clustering, while the TW is not constrained to any fixed number between the structures, privileging the natural structures of the data. Both trees show the merging of the four distinct groups and the middle point (labeled with the star) at the same level of the hierarchy (at a level of 0.7 units of distance for SL and at memory window 8 for the TW). Notice that in the SL this merging occurs in five different iteration steps, while in the TW it happens at the same time. One

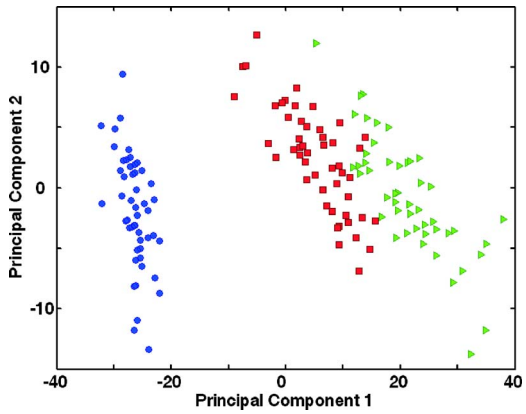

FIG. 4. (Color online) Two principal components of the iris data set. The symbols correspond to the three different categories.

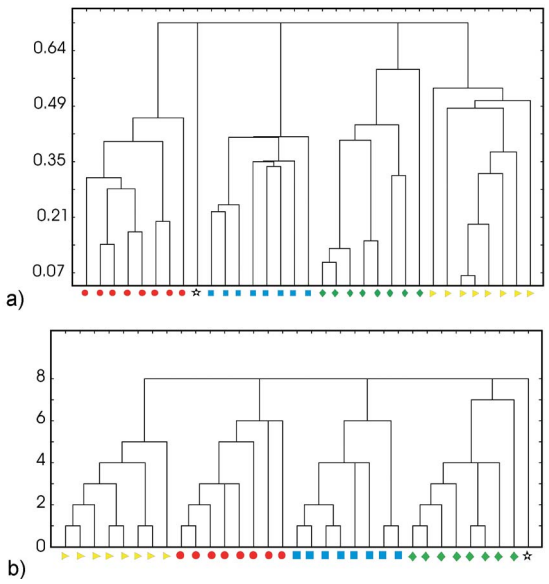

FIG. 5. (Color online) The resulting dendrogram of the SL procedure (a) and the TW procedure (b) over data set I. The abscissas are, respectively, interelements distance, and memory parameter.

interesting feature is that the value $\mu=8$, achieved in the complete merging [Fig. 5(b)], does not occur by chance. Indeed each of the four groups merged at this step has exactly eight elements and as we stressed earlier, the periods of the attractors resulting from the walk are at least $\mu+1$. Thus, starting at a given point of the data set, the self-avoidance imposes the walker to find an attractor of at least nine points. As this procedure is running at the same time for all the N points, the attractors are superimposed, creating a single large group.

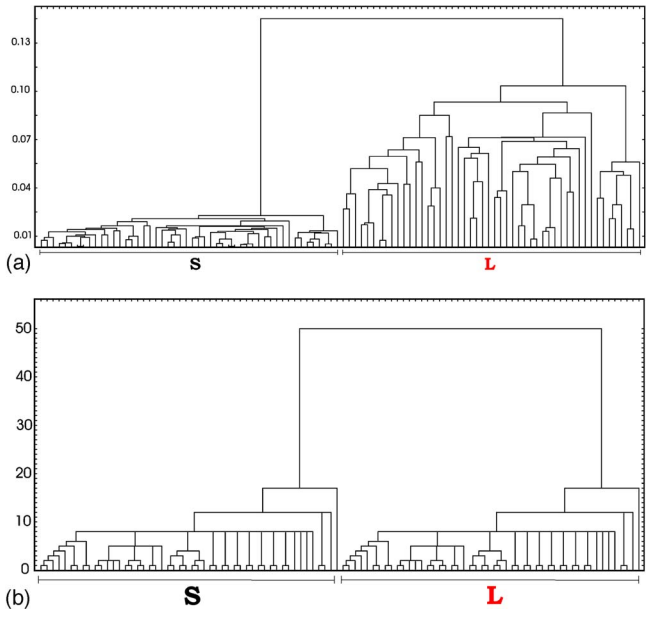

FIG. 6. (Color online) The resulting dendrograms of the (a) SL procedure and (b) TW procedure over data set II. The *L*-labeled cluster corresponds to the larger group in Fig. 3(b) and the *S*-labeled cluster to the smallest one. The abscissas are, respectively, interelements distance and memory parameter.

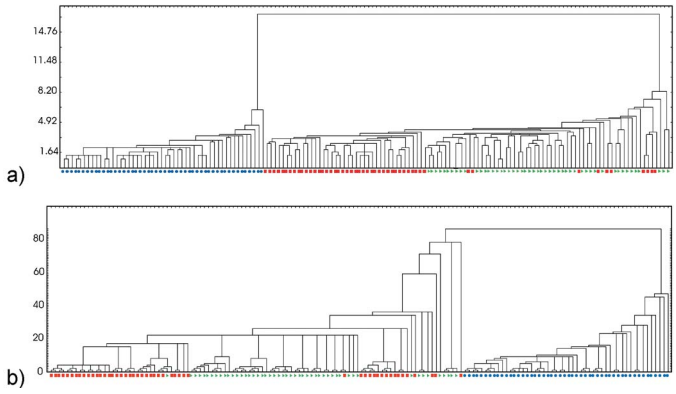

FIG. 7. (Color online) Dendrograms of the iris data set. (a) SL algorithm; (b) TW algorithm. Symbols correspond to the symbols in Fig. 4.

Figure 6 illustrates well the "local action" feature of the TW algorithm. The data set II is constructed by two copies of the same structure in different scales. In the SL dendrogram [Fig. 6(a)], one can see two distinct clusters, yet with different densities. The larger one in Fig. 3(b) gives rise to the branch of the tree with larger internode distances. The smallest one is represented in the denser branch. On the contrary, the TW tree is exactly symmetrical, representing the equivalent sets in an identical manner. This is an important feature of the proposed algorithm. Although having information about the whole system (on the neighborhood rank table), the TW favors local information for merging the clusters. Moreover, the interelement distances are not considered, only ranking relationships. These results suggest a possible advance in the use of automatic pattern recognition in different scales.

The two principal components of the data vectors of the iris flowers data set have been plotted and shown in Fig. 4, where one can clearly distinguish two linearly separated groups, the first labeled with the circles and the other encompassing the two other labels.

The latter labels are not linearly separable as can be confirmed in Figs. 7(a) and 7(b) for the SL result and for the TW result, respectively. One can notice that both methods have been able to distinguish clearly two groups. In turn, the squares plus diamonds group can be divided into subgroups according to the cutoff chosen, and the results are strongly different for each method. The SL method, yet with some misclassifications and many outliers, has been able to distinguish the squares and diamonds into two subgroups. This is not observed in Fig. 7(b) for the TW method, which has been able to separate both labels, although inside a large single cluster. A similar result can be found in Ref. [19] for Kohonen maps. Kohonen maps and the TW method work on a vicinity that are not essentially binary as it is for the SL method. In fact, if one visually tries to recognize clusters in Fig. 4 without the *a priori* knowledge of the labels and with the Euclidean distance among the points as the only differentiable parameter, one would be able to distinguish only between two groups. This result is thus coherent, with the method employed.

V. SUMMARY AND CONCLUDING REMARKS

The rapid development of computer power has allowed access to huge databases but has also brought the desire for elaborated and diversified methods of analysis of such data. In many of the emerging applications, it is clear that no single approach for classification is optimal and often multiple methods have to be combined for the desired outcome [20].

Here we present an algorithm for pattern recognition based on a deterministic walk. The algorithm allows an automatic hierarchical representation of the implicit structure on a given data set in an efficient manner.

Some advantages can be stressed on the use of the proposed algorithm.

(1) One need not work with the distance matrix, but only with a neighborhood rank table for each individual. It could be of great use in problems where the relationships among the objects are qualitative and the introduction of a distance matrix would be arbitrary. Notice that it is not possible to adapt the SL method building a distance matrix from a neighborhood table, for instance, considering the nearest neighbor having a distance of 1, the second one the distance of 2, and so forth. This cannot be implemented since the neighborhood table is a rank table and therefore not necessarily symmetric. Given two points A and B of the table, if point A is the nth neighbor of B, B is not necessarily the nth neighbor of A. The "distance" based on this rank is not symmetric, breaking up the formal definition of metrics imperative to single linkage algorithm.

(2) The tourist walk favors local information for each data point and makes no restriction for the tree structure. Points are merged if they belong to the same attractor, where an attractor is a group of points mutually next. In turn, two attractors form a node in the tree if they are coalesced in a given hierarchy level.

(3) Disregarding the distances among the elements allows a representation of the data set that is invariant by scale transformation.

At present we are studying if the stochastic version of the tourist walk [21,22] can improve the performance of the method. Applications to real-world data sets will be presented in a future work.

ACKNOWLEDGMENTS

The authors acknowledge support from the Brazilian agencies CAPES, CNPq (Grant No. 305527/2004-5), and FAPESP (Grant No. 2005/02408-0).

[1] A. W. Liew, H. Yan, and M. Yang, Pattern Recogn. **38**, 2055 (2005).
[2] J. Clatworthy, D. Buick, M. Hankins, J. Weinman, and R. Horne, British Journal of Psychology **10**, 329 (2005).
[3] R. Xu, IEEE Trans. Neural Netw. **16**, 645 (2005).
[4] A. K. Jain, M. N. Murty, and P. J. Flynn, ACM Comput. Surv. **31**, 264 (1999).
[5] J. Vesanto and E. Alhoieni, IEEE Trans. Neural Netw. **11**, 586 (2000).
[6] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A. L. Barabási, Science **297**, 1551 (2002).
[7] M. S. Su and Y. C. Liu, Pattern Recogn. **38**, 1887 (2005).
[8] B. Derrida, Physica D **107**, 186 (1997).
[9] J. Scholl and E. Schol-Paschinger, Pattern Recogn. **36**, 1279 (2003).
[10] H. Freund and P. Grassberger, Physica A **190**, 218 (1992).
[11] L. A. Bunimovich, Physica D **187**, 20 (2004).
[12] D. Boyer and H. Larralde, Complexity **10**, 52 (2005).
[13] G. F. Lima, A. S. Martinez, and O. Kinouchi, Phys. Rev. Lett. **87**, 010603 (2001).
[14] H. E. Stanley and S. V. Buldyrev, Nature (London) **413**, 373 (2001).
[15] O. Kinouchi, A. S. Martinez, G. F. Lima, G. M. Loureno, and S. Risau-Gusman, Physica A **315**, 665 (2002).
[16] D. Boyer, O. Miramontes, G. Ramos-Fernandez, J. L. Mateos, and G. Cocho, Physica A **342**, 329 (2004).
[17] C. A. S. Terçariol and A. S. Martinez, Phys. Rev. E **72**, 021103 (2005).
[18] R. A. Fisher, Annals of Eugenics **7**, 178 (1936).
[19] J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas, in Proceedings of the Matlab DSP Conference, 1999 (unpublished), pp. 35–40.
[20] A. K. Jain, R. P. W. Duin, and J. Mao, IEEE Trans. Pattern Anal. Mach. Intell. **22**, 4 (2000).
[21] S. Risau-Gusman, A. S. Martinez, and O. Kinouchi, Phys. Rev. E **68**, 016104 (2003).
[22] A. S. Martinez, O. Kinouchi, and S. Risau-Gusman, Phys. Rev. E **69**, 017101 (2004).