How to partition a low-dimensional data set into disjoint clusters of different geometric structures.

Gilad Lerman

Courant Institute for Mathematical Sciences
New York University
251 Mercer Street, New York, NY 10012

Abstract

We present a fast algorithm for partitioning a low-dimensional data set in $\mathbb{R}^n$ into disjoint clusters of different geometric structures.

The main tool used is an L_2 variant of Jones' β numbers, which measure the scaled deviations of the data set from approximating d-planes at different scales and locations. The Jones function is formed by adding the squares of the L_2 Jones numbers at different scales and the same location.

We suggest a fast algorithm for computing a numerical approximation of the Jones function and use it to cluster the given set according to the computed values of this function. The algorithm has been tested successfully on low-dimensional sets in ambient dimensions of sizes at most 25. Here we apply this procedure to an edge map of an image and a six-dimensional set of currency exchange rates.

1 Introduction

We present a fast multiscale algorithm for geometric clustering of low-dimensional data sets in $\mathbb{R}^n$.

The following example illustrates what we mean by geometric clustering. Assume that the data set is a union of two disjoint subsets. The first subset is sampled independently and uniformly along a curve, while the second one is sampled independently and uniformly along a two-dimensional surface. The curve and the surface are not necessarily smooth. The whole data set can be clustered according to the two underlying subsets. Additional clusters or subclusters may correspond to different smoothness properties of the curve and surface, including nonsmooth structure (e.g. corners, crossings and sharp edges). In particular, some clusters may arise due to possible intersections of the curve and surface.

Note that the above clustering is different than the one obtained by some traditional algorithms. Such algorithms are usually based only on the density of points and thus take into account mainly the zero-dimensional geometry.

Our approach for geometric clustering is based on the theory of quantitative rectifiability. This theory was created by Peter Jones [6] and has been extended by different authors (see e.g. [8] for review and some references). The aim of the theory is to fit (if possible) a special d-dimensional surface with nearly minimal d-volume to certain sets in $\mathbb{R}^n$.

Ronald R. Coifman initiated the development of numerical methods based on the theory of quantitative rectifiability for the geometric analysis of data sets.

In this paper we address Coifman's initiative. We use L_2 variants of Jones' β numbers and square function. Given a finite set in $\mathbb{R}^n$ and a fixed dimension d, $d < n$, the d-dimensional β_2 numbers record the scaled

L_2 error of approximating a set by error-minimizing d-planes at different scales and locations. Scales and locations are given here by restricting the data set into cubes in an extended dyadic grid. Following Bishop and Jones [2], we form a square function J_2 by adding the squares of the β_2 numbers from all scales and the same location. In [9, 7] it is shown how to use these β_2 numbers and Jones' function to fit a special d-dimensional surface with small d-volume to a sizeable portion of the given data set. This theory supports the algorithm in this paper.

We first formulate a fast algorithm for computing a numerical approximation of J_2 for a given set and a fixed dimension d. We then suggest an algorithm which partitions a set according to its Jones function. This algorithm can be repeated in a multistage fashion. We exemplify this procedure by using an edge map of an image and a six-dimensional set of currency exchange rates.

A prominent difficulty in pursuing our goal is known as the curse of dimensionality. That is, the constants which appear in analysis of sets lying in $\mathbb{R}^n$ depend exponentially on n. We restrict ourselves to low-dimensional sets in $\mathbb{R}^n$ so that some of the constants depend strongly only on the intrinsic dimension. We also apply some random sampling to speed up the algorithm, where there is a strong dependence on the ambient dimension. The algorithm has been tested successfully on sets in ambient dimensions of sizes at most 25.

I would like to thank Professors Ronald R. Coifman and Peter W. Jones for suggesting to me this area of research and for helpful discussions and insights. My thanks to the reviewers for their suggestions and for pointing out the reference [3].

2 Dyadic grids with respect to a set

In this section we define some dyadic grids with respect to a given cube. We also assign a certain cube to any finite set. The dyadic grids of a given finite set are the ones of its corresponding cube.

Assume that Q is a cube in $\mathbb{R}^n$ with sidelength $l(Q)$. Denote by $\mathcal{D}_j(Q)$ the *dyadic grid at the scale* 2^{-j} *with respect to* Q. That is, the collection of 2^j disjoint cubes with sidelength $2^{-j} \cdot l(Q)$ whose union is Q (fix an arbitrary but consistent partition of the boundaries of these cubes). For example, $\mathcal{D}_1([0,1]^2) = \{[0,\frac{1}{2})^2,\ [0,\frac{1}{2}) \times [\frac{1}{2},1],\ [\frac{1}{2},1] \times [0,\frac{1}{2}),\ [\frac{1}{2},1]^2\}$. Define the *dyadic grid with respect to* Q by $\mathcal{D}(Q) = \cup_{j \geq 0} \mathcal{D}_j(Q)$.

The *extended dyadic grid* $\widetilde{\mathcal{D}}(Q)$ *with respect to the given cube* Q is defined by the formula:

$$\widetilde{\mathcal{D}}(Q) = \bigcup_{e \in \{0,1\}^n} \mathcal{D}\left(Q + \frac{l(Q)}{3} \cdot e\right).$$

For example, $\widetilde{\mathcal{D}}([0,1]) = \mathcal{D}([0,1]) \cup \mathcal{D}([\frac{1}{3},\frac{4}{3}])$.

The use of $\widetilde{\mathcal{D}}(Q)$ instead of $\mathcal{D}(Q)$ avoids some edge effects of dyadic cubes (see e.g. [8] for a quantitative discussion; the example at the end of Section 3 illustrates this idea as well).

We assign to a given finite set E a cube $Q_0^* = Q_0^*(E)$. It is defined by the following procedure. Find a cube $Q_0 = Q_0(E)$ of minimal sidelength containing E:

$$Q_0 = [m_1, m_1 + L] \times \cdots \times [m_i, m_i + L] \times \cdots \times [m_n, m_n + L]. \tag{1}$$

Define

$$Q_0^* = [m_1 - \frac{L}{2}, m_1 + L] \times \cdots \times [m_i - \frac{L}{2}, m_i + L] \times \cdots \times [m_n - \frac{L}{2}, m_n + L]. \tag{2}$$

Note that if e is any vector in $\{0,1\}^n$, then $E \subseteq Q_0^* + \frac{l(Q_0^*)}{3} \cdot e$.

3 L_2 Jones' quantities

In this section we present the two main tools of the L_2 theory of quantitative rectifiability: Jones' numbers and function. We define them for a fixed dimension d, $1 \leq d < n$, and a finite set E in $\mathbb{R}^n$.

Definition 3.1. Jones' β_2 numbers
If $E = \{x_i\}_{i=1}^N$ is a finite set in $\mathbb{R}^n$, $d < n$, Q is a cube in $\mathbb{R}^n$ and N_Q denotes the number of points in $E \cap Q$, then the squares of the d-dimensional β_2 numbers are obtained by the formula:

$$\beta_2^2(Q) = \min_{d-planes\ P} \left(\frac{1}{l(Q)^2 \cdot (n-d) \cdot N_Q} \cdot \sum_{x_i \in Q \cap E} (\mathrm{dist}\,(x_i, P))^2 \right). \tag{3}$$

In other words, the number $\beta_2(Q) \cdot l(Q) \cdot \sqrt{n-d}$ is the averaged L_2 distance of $E \cap Q$ from a best L_2 approximating d-plane for $E \cap Q$. The division by $l(Q)^2$ makes the above quantities dimensionless or equivalently scale-invariant. The further division by $(n-d)$ assures that the β_2 numbers are not larger than one. Note that if $N_Q \leq d+1$, then $\beta_2(Q) = 0$.

The following proposition provides explicit formulae for the Jones β_2 numbers (see [8] for a proof and for a similar formula for a finite set with weights).

Proposition 3.1. *If E is a finite set in $\mathbb{R}^n$, Q a cube in $\mathbb{R}^n$ containing N_Q points of E and A_Q is the $N_Q \times n$ matrix whose rows are the vectors in $E \cap Q$ shifted so that their average is the zero vector. Then the squares of the d-dimensional β_2 numbers are given by the following two equivalent formulas:*

$$1. \quad \beta_2^2(Q) = \frac{\sum_{k=d+1}^{n} \lambda_k}{(n-d) \cdot l(Q)^2}, \tag{4}$$

where λ_i, $i = 1, \ldots, n$, are the eigenvalues of the sample covariance matrix for $E \cap Q$ (the matrix $\frac{1}{N_Q} \cdot A_Q^{\mathrm{T}} \cdot A_Q$).

$$2. \quad \beta_2^2(Q) = \frac{\|A_Q\|_F^2 - \sum_{k=1}^{d} \sigma_k^2}{(n-d) \cdot N_Q \cdot l(Q)^2}, \tag{5}$$

here $\{\sigma_i\}_{i=1}^d$ are the top d singular values of A_Q and $\|A_Q\|_F$ denotes the Frobenius norm of A_Q ($\|A_Q\|_F^2 = \sum_{j=1}^{N_Q} \sum_{k=1}^{n} (A_Q)_{j,k}^2$).

Following [10] (and in view of equation (4)), we interpret the number $\beta_2^2(Q) \cdot l(Q)^2$ as the averaged variance 'lost' in projecting the set $E \cap Q$ on a d-dimensional subspace which maximizes the variance. The average is taken over all 'lost' dimensions (this is another justification for dividing by the factor $(n-d)$ in equation (3)). The number $\beta_2(Q)^2 \cdot l(Q)^2$ also provides the maximum-likelihood estimator for the variance of the isotropic Gaussian noise in the PPCA model for $E \cap Q$ (see [10] for the PPCA model).

Definition 3.2. L_2 Jones' function with respect to a shift e.
If E is a finite set in $\mathbb{R}^n$, Q_0^* is defined with respect to E (see equation (2)) and $e \in \{0,1\}^n$, then for any $x \in E$

$$J_2^e(x) = \sum_{j \geq 0} \beta_2^2(Q_j(x)), \tag{6}$$

where $Q_j(x)$ is the unique cube in $\mathcal{D}(Q_0^* + \frac{l(Q_0^*)}{3} \cdot e)$ with sidelength $2^{-j} \cdot l(Q_0^*)$ containing x.

In practice, we use only $M_{\max}+1$ levels to estimate the function J_2^e, so that $0 \leq j \leq M_{\max}$ in equation (6).

Definition 3.3. L_2 Jones' function.
If E is a finite set in $\mathbb{R}^n$, then the L_2 Jones' function J is the average of all functions J^e, where $e \in \{0,1\}^n$.

In [9, 7] it is shown how to use the L_2 Jones numbers and function in order to fit a special d-dimensional surface with small d-volume to a sizeable portion of the data set.

In order to facilitate the understanding of the above definitions we end this section with the following artificial example.

Example: Sample independently and uniformly 2000 points along the line segment $y = x$, $0 \leq x \leq 1$, and 1000 points along the line segment $y = -x + \frac{13}{8}$, $\frac{5}{8} \leq x \leq 1$. The numbers 1000 and 2000 are arbitrary. Let E be the set containing these 3000 points. The significant geometric "cluster" of this set lies around the crossing point of the two line segments (this is the point $(\frac{13}{16}, \frac{13}{16})$).

Note that $Q_0(E) = [0,1]^2$ and $Q_0^*(E) = [-\frac{1}{2}, 1]^2$. Observe that the only cubes in $\mathcal{D}(Q_0^*)$ with non-zero β_2 numbers are Q_0^* and $Q_{1,4}^* := [\frac{5}{8}, 1]^2$. Indeed, any other cube in $\mathcal{D}(Q_0^*)$ either contains no points or contains points sampled along one line. Therefore

$$J_2^{(0,0)} = \beta_2^2(Q_0^*) \cdot \chi_{Q_0^*} + \beta_2^2(Q_{1,4}^*) \cdot \chi_{Q_{1,4}^*},$$

where χ denotes the indicator function.

The above equation illustrates the disadvantage of using only the grid $\mathcal{D}(Q_0^*)$ and the corresponding function $J_2^{(0,0)}$. Indeed, the crossing point is not well-detected by J. It is only the points within distance $\frac{3}{16}$ to the crossing point (more precisely, the points in the cube $[\frac{5}{8}, 1]^2$) which are detected by having the maximum value of $J_2^{(0,0)}$. On the other hand, one can check that the functions $J_2^{(1,1)}$ and J_2 detect the crossing point within the average distance between points in E.

4 The computation of the L_2 Jones function

Let E be a set of N_E points in $\mathbb{R}^n$, where $N_E \gg n$ and let d be a fixed dimension, $1 \leq d < n$. We describe the numerical evaluation of the d-dimensional Jones' function J_2 for the set E as follows.

Computation of $\beta_2(Q)$: Fix a cube Q in $\mathbb{R}^n$ containing N_Q points of E. We compute the d-dimensional Jones number $\beta_2(Q)$ according to equation (5). The main effort is in evaluating the top d singular values of the $N_Q \times n$ matrix A_Q, whose rows are the vectors in $E \cap Q$ shifted by their average. This computation can be done by using a purely iterative algorithm in $O(N_Q \cdot n \cdot d)$ operations (see e.g. [5] and [1]; [10] implies another $O(N_Q \cdot n \cdot d)$ iterative algorithm for computing $\beta_2(Q)$ by using equation (4)). We remark that the above estimate for the cost of the algorithm does not take into account the speed of convergence.

An algorithm for computing J_2^e: Fix a shift vector e in $\{0,1\}^n$. Denote by Q_0^e the cube $Q_0^* + \frac{l(Q_0^*)}{3} \cdot e$, where Q_0^* is the cube defined in equation (2).

The following are the input variables of the algorithm:

1. A is an $N_E \times n$ matrix whose rows are the vectors of E.
2. $M_{\max}$ is an integer setting the maximum number of levels.
3. k_0 is an integer greater than d which is used in the following stopping rule: Stop at a cube Q in $\cup_{l=0}^{M_{\max}} \mathcal{D}_l(Q_0^e)$, if it satisfies the condition:

$$\#(E \cap Q) \leq k_0 \,. \tag{7}$$

4. ϵ_0 is a (small) real number which is used in the following stopping rule: Stop at a cube Q in $\cup_{l=0}^{M_{\max}} \mathcal{D}_l(Q_0^e)$, if it satisfies the condition:

$$\beta_2(Q) < \epsilon_0 \,. \tag{8}$$

A cube Q which satisfies either equation (7) or equation (8) is referred to as a stopping cube. All non-stopping cubes used at level l, $l = 1, \ldots, M_{\max}$, are indexed as follows: Q_l^j, $j = 1, \ldots, m_l$, where m_l is the number of all such cubes. Denote by N_l^j the number of points in Q_l^j and by N_l the total number of points in level l. That is $N_l = \sum_{j=1}^{m_l} N_l^j$.

The output of the algorithm is a vector J of size N_E. If $i = 1, \ldots, N_E$, then $J(i)$ is the numerical approximation for J_2^e of the i-th row vector of A.

The main two variables which are updated at each level l, $l = 1, \ldots, M_{\max}$, are described as follows:

- A_l is an $N_l \times n$ matrix. Each row of A_l is a vector in $E \cap \cup_{j=1}^{m_l} Q_l^j$. The rows are ordered so that all points in a cube Q_l^j have consecutive indices.
- b_l is a vector of size m_l. If $1 \le j \le m_l$, then $b_l(j)$ is the approximation of J_2 at stage l for points in the cube Q_l^j.

At each level l one has to keep a record of m_l, $\{N_l^j\}_{j=1}^{m_l}$, the centers of the cubes $\{Q_l^j\}_{j=1}^{m_l}$ and the permutation of the rows in A_l with respect to A. This information is used only at the next level.

The algorithm starts at level 0. If Q_0^e is a stopping cube then the procedure terminates and J:=0. Otherwise, A_0 and b_0 are initialized as follows: $A_0 = A$ and $b_0 = \beta_2^2(Q_0^e)$.

At each level $l, 0 < l \le M_{\max}$, the algorithm consists of two parts:

1. Coding, updating main variables and stopping according to k_0.
   Denote by A_{l-1}^j, $j = 1, \ldots, m_{l-1}$, the submatrix of A_{l-1} whose rows are the points in $E \cap Q_{l-1}^j$. Let w^j be the center of the cube Q_l^j. Code each row vector u in A_{l-1}^j by a vector in $\{0,1\}^n$ according to $\text{sign}(u_i - w_i^j)$, $i = 1, \ldots, n$. Note that this code determines the belonging of each row vector in A_{l-1}^j to a dyadic subcube of Q_{l-1}^j. Denote all such dyadic subcubes containing points of E by Q_l^j, $j = 1, \ldots, m_l$. This notation will be updated later to exclude stopping cubes.

   Form the matrix A_l from A_{l-1} by reordering the rows in each submatrix A_{l-1}^j, $j = 1, \ldots, m_{l-1}$, so that rows with the same code have consecutive indices (these rows belong to the same dyadic subcube). Reorder the vector b_l respectively.

   Stop at a cube Q_l^j if it contains at most k_0 points (see equation (7)) and exclude this cube from the collection $\{Q_l^j\}_{j=1}^{m_l}$. Set the J value of each point in Q_l^j to be the corresponding value of b_l. Remove the submatrix A_l^j from A_l and the corresponding part from b_l.

   The operation count for this step for a cube Q_{l-1}^j with N_{l-1}^j points is $O(n \cdot N_{l-1}^j)$. The whole computation at this stage is thus $O(n \cdot N_E)$.

2. Computation of Jones' β_2 numbers.
   For each $j = 1, \ldots, m_l$, compute $\beta_2(Q_l^j)$ according to formula (5). Update the vector b_l as follows:

$$b_l(j) = b_l(j) + \beta_2^2(Q_l^j), \quad j = 1, \ldots, m_l.$$

   Stop at a cube Q_l^j if its β_2 number is less than ϵ_0 (see equation (8)) and exclude this cube from the collection $\{Q_l^j\}_{j=1}^{m_l}$. Set the J value of each point in Q_l^j to be the corresponding value of b_l. Remove the submatrix A_l^j from A_l and the corresponding part from b_l.

   The amount of computation performed for a cube at level l with N_l^j points is $O(N_j^l \cdot n \cdot d)$. The cost for the whole step is $O(N_E \cdot n \cdot d)$.

The above two steps are repeated until level $M_{\max}$. The values of the vector J for the points in the remaining non-stopping cubes at level $M_{\max}$ are the same as those of $b_{M_{\max}}$.

The operation count for the whole algorithm is $O(M_{\max} \cdot N_E \cdot n \cdot d)$. The parameter $M_{\max}$ can be assumed to be at most $\log_2 N_E$.

Computation of J: When computing the exact average of J_2, one needs to repeat the above algorithm with 2^n different grids. In order to avoid this unrealistic computation when $n > 3$, we choose randomly some vectors in $\{0,1\}^n$ and evaluate J_2 by averaging J_2^e over all corresponding random grids. This way the cost of computing J_2 is of the same order as the cost of computing J_2^e. Numerical experience supports this random computation, however we have not performed the analysis which justifies it.

5 The partition algorithm

The one-stage algorithm: The main input of this algorithm is a fixed integer d, $d < n$, and a set E in $\mathbb{R}^n$ with N_E points given as an $N_E \times n$ matrix. The parameters $M_{\max}$, k_0 and ϵ_0 (see Section 4) also need to be specified. The output is a collection of disjoint sets (usually two or three) whose union is E. Assume here that $n \leq 25$ and $d \leq 4$.

The following two steps describe the one stage algorithm:

1. Evaluate the d-dimensional Jones function J_2 of the set E according to the algorithm in Section 4 (J_2 is a vector of length N_E).
2. Partition the set E according to different modes in the histogram of J_2. For example, if the histogram is bimodal, then separate the set E into two disjoint sets according to the belonging of the corresponding J_2 values to each one of the modes. If the histogram is unimodal then no partition is done.

Remarks:
1. We usually transform the set E in advance so that its center of mass is zero and its singular vectors are parallel to the axes.
2. Sometimes we separate the values of J_2 into two sets using the median as threshold and then partition E respectively (this partition is more arbitrary).

The reason why the above partition results in clusters with different geometric structures follows from [9, 7]. Indeed, [9, 7] implies that a set with low values of the d-dimensional Jones function is "well-approximated" by a certain d-dimensional surface with small d-volume. On the other hand, high values of the one-dimensional Jones function occurs either due to higher-dimensional geometric structure or due to crossings, corners and edges of the underlying surface. These observations suggest the partition of E into two subsets with low and high values of J_2 according to a certain threshold (e.g. the median of J_2). The choice of threshold is however arbitrary. Our experience shows that the partitioning according to modes in the histogram of J_2 results in less arbitrary threshold. Moreover, the modes in the histogram may indicate the existence of more than two main geometric clusters.

We are not aware of an objective measure for the quality of our geometric clustering. Therefore we sometimes use visualization techniques to confirm the partitioning if possible. If the data set arises from an image (e.g. blocks of pixels), we check the induced partition on the original image.

We remark that the one-stage partition algorithm might put two separated clusters in the same set. Indeed, if the data set is sampled independently and uniformly from two separated parts of a smooth surface, then our algorithm will not differentiate between the two sets. The use of either visualization techniques (see e.g. [10]) or standard clustering methods can separate the substructures.

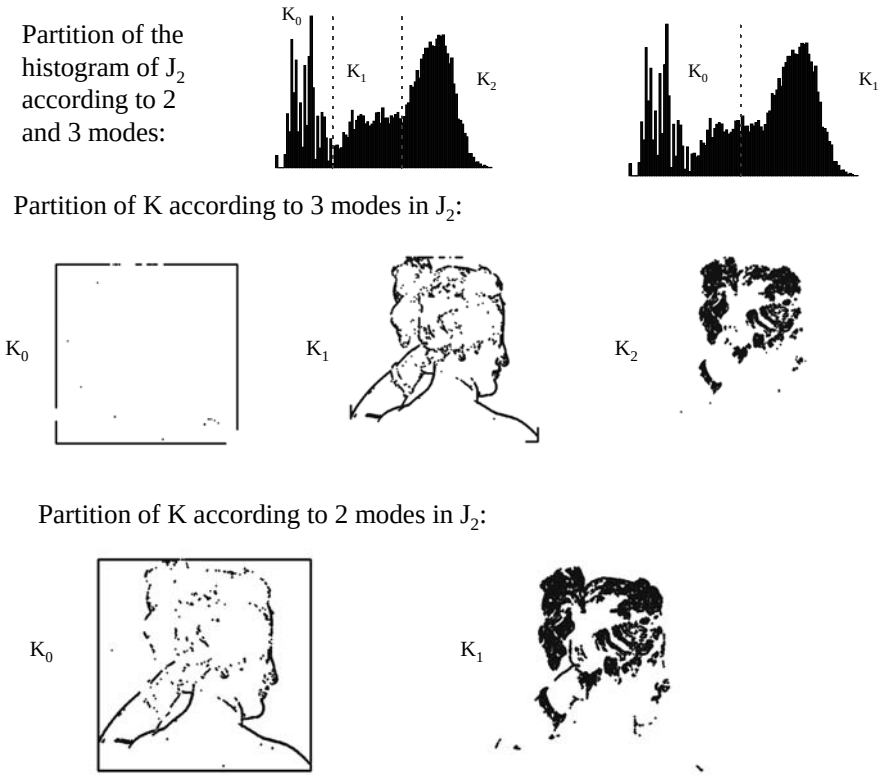

Figure 1: *The histogram of J_2 for "Paolina", its separation to both two and three modes and the induced partitions of "Paolina".*

The multistage algorithm: The input of the multistage algorithm is a set K in $\mathbb{R}^n$ (assume here $n \leq 25$). It repeats recursively the one-stage algorithm in a tree structure. The procedure terminates at a set E in a leaf of the tree if either one of the following two conditions is satisfied:

1. There are no significant modes in the histogram of J_2 (computed for E).
2. $\frac{\#E}{\#K} \leq \epsilon_1$, where ϵ_1 is chosen by the user.

In practice there are only few stages of the multistage algorithm. Therefore the cost of this algorithm is comparable to the cost of computing the Jones function J_2 for K (see Section 4).

The success of the algorithm in dimensions n, $n > 3$ (assume $n \leq 25$) depends on the low-dimensional structure of the data set. The following example of a typical high-dimensional set illustrates the problem: Sample independently and uniformly 2^n points from the n-dimensional unit cube $[0,1]^n$. In this case, practically all cubes of level one are stopping cubes. That is, only the zeroth level is taken into account. On the other hand if the data set K is sampled along a d-dimensional surface with surface area S_d and $l(Q_0(K)) = 1$ (see equation (1)), then the number of effective levels (having mainly non-stopping time cubes) is of order $O(\frac{1}{d} \cdot \log_2 \frac{N}{S_d})$.

The algorithm has been tested successfully on some 25-dimensional data sets obtained from 5×5 blocks of pixels taken from different images. We have not tried the algorithm in very high dimensions. We believe that it has to be modified for high dimensions (and probably also for moderate ones). More precisely, the rigid dyadic grids has to be replaced by "dyadic" regions adapted to the given set.

6 Numerical results

6.1 Example 1: Paolina

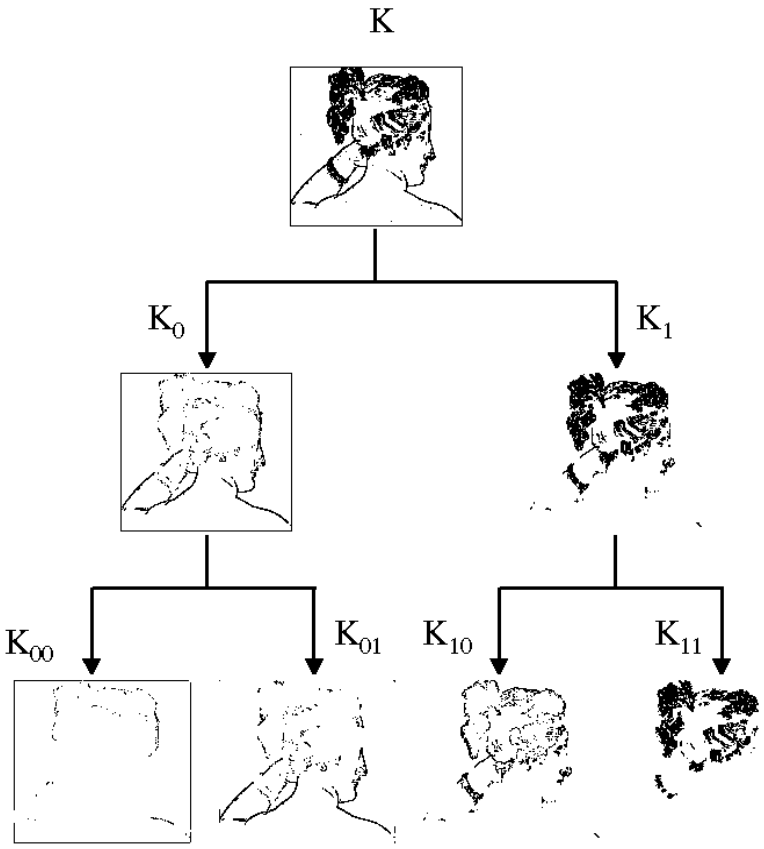

Figure 2: *The two-stage algorithm applied to Paolina's edge map.*

The input data in this example is a set of $80,640$ points in $\mathbb{R}^2$; it is depicted at the top of Figure 2. The set is obtained by an edge map of the image "Paolina" (see e.g. [4] for the original image and for finding geometric structure in the same edge map; unlike [4] we only use the x and y coordinates of the edge map and exclude the curvature information).

We chose $M_{\max} = 9$, $k_0 = 4$, $\epsilon_0 = 10^{-3}$ and $d = 1$. All four dyadic grids in $\mathcal{D}(Q_0)$ are used in computing J_2. We also rotate the set so that its two main principal axes are the coordinate axes. The histogram of J_2 for the set K is shown in Figure 1. There are three apparent modes in this histogram, but it can also be separated according to the two extreme modes. The modes' separation and the induced partitions of K are shown in Figure 1.

In Figure 2, we exemplify the two-stage algorithm. At the first stage we partition K according to the two extreme modes in the histogram. We remark that there were no significant modes in the histogram of the Jones function of K_1; thus the partition of K_1 was done according to the corresponding median.

6.2 Example 2: Six-Dimensional Set of Currency Exchange Rates

In this example we study a six-dimensional set of exchange rates versus the U.S. dollar of the following currencies: Australian dollar, British pound, Deutsch mark, Japanese yen, Swiss frank and the dollar index. Each exchange rate is scaled to have the starting value one. These rates were collected in 2516 days in the periods: 5/1/89 - 12/31/98. Thus the data set consists of 2516 points in $\mathbb{R}^6$.

We run the one-stage algorithm with the following parameters: $M_{\max} = 7, k_0 = 4, \epsilon_0 = 10^{-3}$ and with $d = 1, 2$. All partitions in the multistage algorithm were done according to the medians of the corresponding Jones functions.

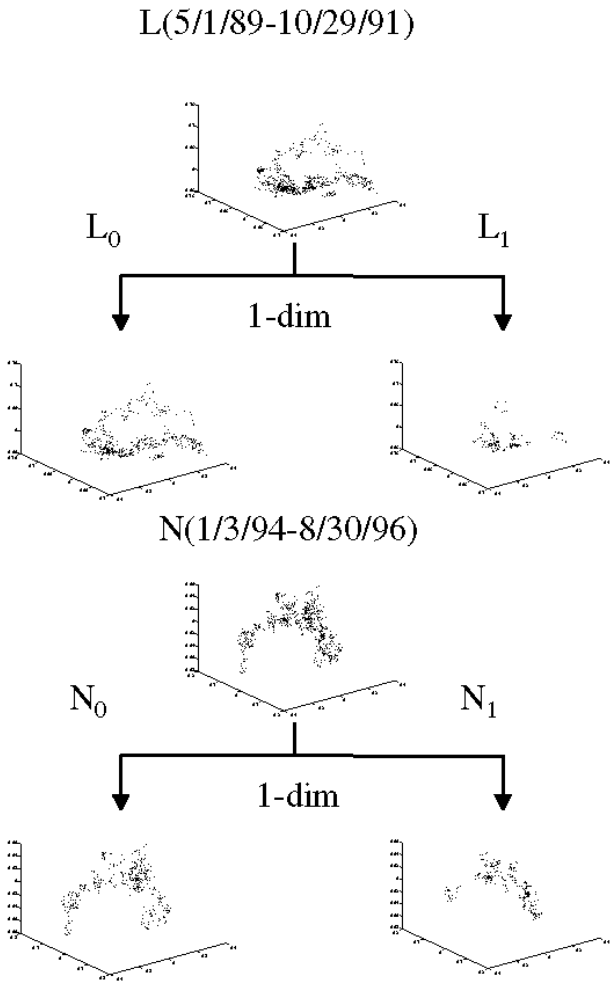

Figure 3: *The one-stage algorithm applied to the sets L and N with* $d = 1$.

When $d = 2$ the sets K_0 and K_1 are more concentrated in certain regions and time intervals than when $d = 1$. We look for correlations between time intervals and time periods of points in the sets K_0 and K_1 when $d = 2$. We find out that the following time intervals: 5/1/89-10/29/91 and 1/3/94-8/30/96 contain less than 10% of the set K_1 when $d = 2$. Consequently we divide K according to the four different time periods with the following notation:

L corresponds to the time interval	5/1/89-10/29/91.
M corresponds to the time interval	10/30/91-12/31/93.
N corresponds to the time interval	1/3/94-8/30/96.
O corresponds to the time interval	9/2/26-12/31/98.

The use of the time information in the above partition assures some coherence of the resulted sets.

We run the one-stage algorithm on each one of the sets L, M, N and O with the same parameters except for d; $d = 1$ for L and N and $d = 2$ for M and O. The resulted partitions are presented in Figures 3 and 4, where the six-dimensional sets are projected onto the top three principal vectors (top singular vectors) of the corresponding parent set. Note that the curve structure is evident in the three-dimensional projections of L and M (we remark that the scaling of the axes is not uniform). A higher dimensional structure and a corner can be seen in the three-dimensional projections of M and O (two substructures are also apparent in both of these sets).

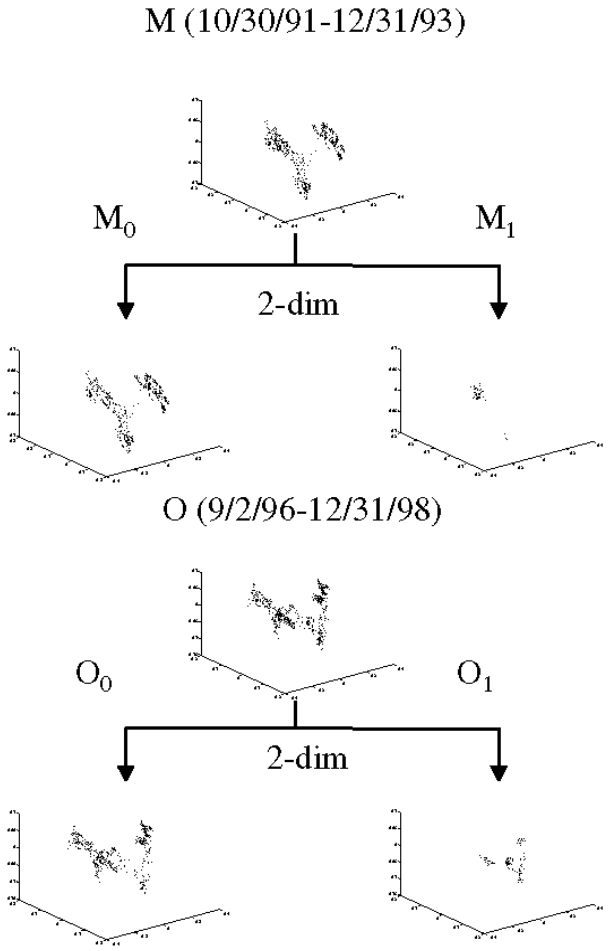

Figure 4: *The one-stage algorithm applied to the sets M and O with $d = 2$.*

The time information can be used as above to obtain more coherent parts of L, M, N and O. The process can then be repeated. Such partitioning can lead to approximating parts of K by local d-planes, $1 \leq d < n$ (the value of d varies at different locations). For each part approximated by a d-plane, it is possible to check the projection of the currencies on the corresponding principal axes and consequently find out the main correlations. For example, assume L is such a part approximated by a line (averaged L_2 error = 0.06); it is

noted that the Japanese yen has a small projection onto the first singular vector of L and a large projection onto the second one. If we remove the Japanese yen from L the averaged L_2 error of approximating the resulted five-dimensional set by a line is reduced by two.

At this point we are not sure about a canonical geometric clustering of K and also about automating the above process. The main problem is with the use of the time information.

References

[1] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe and H. van der Vorst. *Templates for the solution of algebraic eigenvalue problems: A practical guide*. SIAM, 2000.

[2] C. J. Bishop and P. W. Jones. Harmonic measure, L^2 estimates and the Schwarzian derivative. *J. Anal. Math.*, 62:77–113, 1994.

[3] C. M. Bishop and M. E. Tipping. A hierarchical latent variable model for data visualization. *IEEE Trans. on PAMI*, 20(3):281–293, 1998.

[4] B. Dubuc and S. Zucker. Indexing visual representations through the complexity map. In *ICCV95*, pages 142–149, 1995.

[5] G. H. Golub and C. F. Van Loan. *Matrix computations*. The John Hopkins University Press, 1996.

[6] P. W. Jones. Rectifiable sets and the traveling salesman problem. *Invent. Math.*, 102(1):1–15, 1990.

[7] P. W. Jones and G. Lerman. On quantifying d-dimensional rectifiability properties of Borel measures by using L_2 square functions. In preparation.

[8] G. Lerman. Geometric transcriptions of sets and their applications to data analysis. PhD thesis. May 2000.

[9] G. Lerman. Quantifying curve-like structure of Borel measures by using L_2 Jones' quantities. In preparation.

[10] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *J Royal Stat. Soc. B*, 61(3), 1999.