Knowledge discovery by accuracy maximization

Stefano Caciarelli,a,b,c Claudio Luchinat,a,d,1 and Leonardo Tenori
d

*Magnetic Resonance Center (CERM) and Department of Chemistry, University of Florence, 50019 Sesto Fiorentino, Italy; bDepartment of Medical Oncology, Dana–Farber Cancer Institute, Harvard Medical School, Boston, MA 02115; cMetabolomics Platform, Spanish Biomedical Research Center in Diabetes and Associated Metabolic Disorders, Rovira i Virgili University, 43007 Tarragona, Spain; and dFiorGen Foundation, 50019 Sesto Fiorentino, Italy

Edited by Harry B. Gray, California Institute of Technology, Pasadena, CA, and approved February 25, 2014 (received for review December 3, 2012)

Here we describe KODAMA (knowledge discovery by accuracy maximization), an unsupervised and semisupervised learning algorithm that performs feature extraction from noisy and high-dimensional data. Unlike other data mining methods, the peculiarity of KODAMA is that it is driven by an integrated procedure of cross-validation of the results. The discovery of a local manifold's topology is led by a classifier through a Monte Carlo procedure of maximization of cross-validated predictive accuracy. Briefly, our approach differs from previous methods in that it has an integrated procedure of validation of the results. In this way, the method ensures the highest robustness of the obtained solution. This robustness is demonstrated on experimental datasets of gene expression and metabolomics, where KODAMA compares favorably with other existing feature extraction methods. KODAMA is then applied to an astronomical dataset, revealing unexpected features. Interesting and not easily predictable features are also found in the analysis of the State of the Union speeches by American presidents: KODAMA reveals an abrupt linguistic transition sharply separating all post-Reagan from all pre-Reagan speeches. The transition occurs during Reagan’s presidency and not from its beginning.

KODAMA can use several supervised classifiers such as k-nearest neighbors (kNN) (7), support vector machine (SVM) (8), and a combination of principal component analysis (PCA) and canonical analysis (CA) with kNN (PCA-CA-kNN) (9).

A large number of unsupervised feature extraction techniques have been designed, like KODAMA, to preserve the local structure of data. To better assess KODAMA’s performance against this metric, we compared KODAMA with several of these other unsupervised techniques: diffusion maps (DM) (10), isometric feature mapping (ISOMAP) (11), PCA (12), locally linear embedding (LLE) (13), random forest (RF) (14), Sammon’s nonlinear mapping (SAMMON) (15), stochastic proximity embedding (SPE) (16), and t-distributed stochastic neighbor embedding (t-SNE) (17). Despite the strong record of these methods, they are often not very successful when applied to noisy and/or high-dimensional data. Conversely, KODAMA demonstrates a high level of performance as an unsupervised method on datasets with these characteristics, ranging from simulated to a broad spectrum of scientific data. Finally, the greater flexibility of KODAMA is also apparent in a semisupervised context.

Methods

KODAMA consists of five steps, as illustrated in SI Appendix, Fig. S1. For a simple description of the method, we can divide KODAMA into two parts: (i) the maximization of cross-validated accuracy by an iterative process (steps I and II), resulting in the construction of a proximity matrix (step III), and (ii) the definition of a dissimilarity matrix (steps IV and V). The first part entails the core idea of KODAMA, that is, the partitioning of data guided by the maximization of the cross-validated accuracy, as shown in Fig. 1A and in the flowchart in SI Appendix, Fig. S1. At the beginning of this part, a fraction \( \varphi \) of the total samples (\( \varphi = 0.75 \) as default) are randomly selected from the original data. The whole iterative process (steps I–III) is repeated \( M \) times (\( M = 100 \) as default) to average the effects owing to the randomness of the iterative procedure. Each time that this part is repeated, a different fraction of samples is selected. The second part aims at collecting and processing these results by constructing a dissimilarity matrix to provide a holistic view of the data while maintaining their intrinsic structure (steps IV and V, Fig. 1B and C). Although the method itself is a complex multistep procedure, to make things easy for the final user the source code of KODAMA written for R language can be found in Supporting Information.

Significance

We propose an innovative method to extract new knowledge from noisy and high-dimensional data. Our approach differs from previous methods in that it has an integrated procedure of validation of the results through maximization of cross-validated accuracy. In many cases, this method performs better than existing feature extraction methods and offers a general framework for analyzing any kind of complex data in a broad range of sciences. Examples ranging from genomics and metabolomics to astronomy and linguistics show the versatility of the method.

Author contributions: C.L. designed research; S.C. and L.T. performed research; S.C. conceived the algorithm; S.C., C.L., and L.T. analyzed data; and S.C., C.L., and L.T. wrote the paper.

The authors declare no conflict of interest.

This article is a PNAS Direct Submission.

1To whom correspondence should be addressed. E-mail: luchinat@term.unifi.it.

This article contains supporting information online at www.pnas.org/lookup/suppl/doi:10.1073/pnas.1220873111/-/DCSupplemental.

www.pnas.org/cgi/doi/10.1073/pnas.1220873111

PNAS | April 8, 2014 | vol. 111 | no. 14 | 5117–5122
Appendix

The KODAMA accuracy maximization procedure is illustrated for 2D data points, where kNN (with k = 2) classifier was used. Each point is colored according to the cluster it belongs to; the circle represents its distance to the second-nearest neighbor. More details are shown in SI Appendix, Fig. S2. In the fourth step the matrices are averaged. More details are shown in SI Appendix, Fig. S3. In the fifth step the shortest path is calculated.

Maximization of Cross-Validated Accuracy (Steps I and II). Taking a dataset constituted by N = qN samples and f variables, step I consists of the assignment of each sample to a class defined in the class-indicator vector W = (w1, w2, ..., wN), where w1 is the class label of the jth sample. If W is not predefined, each sample is assigned to a different class. Therefore, in step I, N different classes are created. A 10-fold cross-validation procedure is performed on the basis of the classes defined in W. This procedure is performed using a supervised classifier such as kNN (7), SVM (8), or PCA-CA-kNN (9). Details on the use of these classifiers are given in SI Appendix. Because in this first step removal of samples for cross-validation implies removal of their classes, these samples are forced to join other classes. The global accuracy is calculated by summing up the number of correctly classified samples and dividing this number by the total number of samples. The obtained value is stored in the variable Auv. A record of the predicted class labels for each sample is stored in the vector Zuv = (z1, z2, ..., zN), where z is the predicted class label of the jth sample. Obviously, after the first step Auv = 0.

In step II, an iterative MC procedure optimizes the vector W by maximizing Auv. At the end of these iterations, a classification with a high value of accuracy is stored in the vector W. This procedure includes the following points: (i) A new class-indicator vector V = (v1, v2, ..., vN) is created by randomly swapping some class labels of the misclassified samples with the predicted class labels stored in Zuv. (ii) A 10-fold cross-validation procedure is performed on the basis of the classes defined in V. The relative accuracy value is then stored in Auv and the predicted class labels are stored in Zuv = (z1, z2, ..., zN). (iii) If Auv is equal to, or higher than, Auv, the value of Auv is changed to Auv, the vector W is changed to W, and the vector Zuv is changed to Zuv.

This iterative procedure leads to a pruning of the classes because, in some cases, all samples belonging to one class may also be classified as belonging to a different class, and therefore the swapping of vector V will eliminate one or more classes (this is what always happens in step I). The loop is repeated until either Auv becomes equal to 100% or the maximum number of iterations T is reached (the default value is T = 20). Fig. 1A illustrates how the classification gradually emerges during this iterative process. The evolution of the W and Zuv vectors, as well as of Auv and Auv, is exemplified in SI Appendix, Fig. S2 for 2D data points.

Construction of the Proximity Matrix (Step III). In step III, the classification obtained from the iterations described in step II is used to generate a proximity matrix P = (P(ij)) (N × N) from the vector W, where P(ij) = 1 if i and j are assigned to the same class (i.e., w1 = w2), P(ij) = 0 otherwise (i.e., w1 ≠ w2).

Steps I–III are repeated M times, each time by randomly selecting different N = qN subsets from the N samples, to generate M different P matrices. Each P (N × N) matrix is then padded with zeroes to get a sparse (N × N) matrix where the zeroes fill the cells of P corresponding to the unselected samples at the beginning of step I.

Definition of the Dissimilarity Matrix (Steps IV and V). In step IV the M P matrices are then averaged to generate the average proximity matrix Puv = (Puv(ij)) (N × N) (Fig. 1B). Each element of Puv thus ranges from 0 to 1. More information on this process is provided in SI Appendix, Fig. S3.

High proximities are typical of intercluster relationships. Low proximities are expected for intercluster relationships. Very low proximities between samples are ignored by setting puv = 0 for puv < e, where e is a predefined cutoff. This ensures that occasional proximities between two otherwise unrelated samples are not taken as meaningful. We set e = 0.05 as a default value. Then the Euclidean distances d(ij) are calculated in the f-dimensional space between the N samples. By multiplying 1/puv(ij) by d(ij), a weighted dissimilarity matrix Duv = (duv(ij)) (N × N) is obtained. For all Puv elements puv(ij) equaling 1, the corresponding duv(ij) elements equal the Euclidean distance. For 0 < puv(ij) < 1, duv(ij) represents a distance weighted by the probability that i and j belong to the same class. If puv(ij) = 0 then duv(ij) = ∞.

In step V, the final KODAMA dissimilarity matrix Duv is calculated by applying Floyd’s algorithm (18) to find the shortest path distances between all pairs of points (Fig. 1C). For each value of f = 1, ..., N in turn, all entries duv(ij) are defined as duv(ij) = min(duv(ih) + duv(hj)). The final Duv = (duv(ij)) contains the shortest path distances between all pairs of points. This is a way to capture the global topology of the manifold embedded in the data, as reported in a previous study (11).

Initializing and Constraining. The KODAMA procedure can be started by different initializations of the vector W. Without any a priori information the vector W can be initialized, as described above, with each w being different from the others (i.e., each sample categorized in a one-element class). Alternatively, the vector W can be initialized by a clustering procedure, such as k-means, k-medoids, or hierarchical clustering. Finally, supervised constraints can be imposed by linking some samples in such a way that if one of them is changed the linked ones must change in the same way (i.e., they are forced to belong to the same class). This will produce solutions where linked samples are forced to have the lowest values in the KODAMA dissimilarity matrix.

Optimization of the Parameters. As described above, KODAMA contains adjustable parameters (w, T, M, and e). These parameters do not show remarkable effects if changed within reasonable ranges. Their proposed default values are shown in SI Appendix, Table S1. We describe their optimization in default values and SI Appendix, Fig. S4.

The result of KODAMA is also affected by the choice of the classification method to use in the cross-validation procedure. We tested kNN, SVM, and PCA-CA-kNN, the performance of which has been assessed several times in the analysis of multivariate data (9, 19, 20), but any other classifier can be used for the KODAMA analysis. Depending on the structure of the data, one classifier may perform better than others. After extensive tests, we found that the analysis of the distribution of proximity values puv(ij) can be used to select the best classifier (and its relative parameters). To quantify the information contained in Puv and to assess the significance of the KODAMA result on a high-dimensional dataset the Shannon entropy (H) (21) can be used. Details are provided in SI Appendix. The use of H to assess the significance of a KODAMA analysis was tested on three-clusters, Swiss-roll, and a mixture of Gaussian datasets (SI Appendix, Fig. S5). The results are shown in SI Appendix, Table S2.

Visualization of the Data. Visualization is an important aspect in the analysis of high-dimensional data (22). Feature extraction methods such as multidimensional scaling (MDS) can be used to provide a visual representation of the KODAMA dissimilarity matrix Duv by a set of points in a low dimensional space where the distances between the points are approximately equal to the dissimilarities. Alternatively, t-SNE or tree preserving embedding (TPE) (23) can be used to visualize the KODAMA dissimilarity matrix when the intrinsic dimensionality of the data largely exceeds the embedding dimensionality used to visualize the data—the so-called crowding problem (17). Other feature extraction methods applied to KODAMA are described in SI Appendix.

Time Complexity. KODAMA has polynomial complexity, proportional to the product of the number of cross-validations performed by the time complexity of the classifier used. A 10-fold cross-validation performed with kNN classifier has thus a time complexity of O(0.9 × N^2 × f), where N' is the overall number of data points (N' = qN). KODAMA consequently has a time complexity at most of O(0.9 × M × T × N^2 × f), where M is the number of times that the maximization of the cross-validated accuracy is repeated, and T is the maximum number of MC iterations. Some optimizations can be used to improve the efficiency of the algorithm, for instance, using KD-Tree (24) to improve the storage efficiency or approximate nearest neighbor searching (25) to improve the speed at the cost of slightly lower accuracy. Efficient parallel
formulation of the $k$NN search problem based on graphics processing units are proposed (26), observing speed-ups of 50–60 times compared with central processing unit implementation. Some tests of KODAMA performed with $k$NN, SVM, or PCA-CA-$k$NN are provided in SI Appendix both for synthetic (Tables S3 and S4) and experimental (Table S5) datasets. The running times are significantly longer than those of other unsupervised methods, but still acceptable. The performance of KODAMA is relatively insensitive to the number of variables but decreases substantially with increasing number of samples, although not as substantially as for other methods.

**Results**

**Comparative Tests of KODAMA on Synthetic Datasets. Manifolds.** To visually and intuitively demonstrate the features of KODAMA, we tested its performance in representing 2D manifolds embedded in a 3D space. Fig. 2 illustrates three examples. All of them are intrinsically 2D datasets and can thus be projected onto a plane. The first is the well-known Swiss-roll, similar to the one used in ref. 11. The second is a minimal surface example discovered by Baptiste Meusnier in 1776, the so-called helicoid. Its name derives from its similarity to a helix: For every point on the helicoid there is a helix contained in the helicoid, which passes through that point. The third one is a surface (described by Ulisse Dini in 1866) with constant negative curvature that can be created by twisting a pseudosphere. $H$ is calculated to test whether or not KODAMA is able to catch the internal structure of these highly nonlinear datasets. The calculated $H$ values are 11.39, 10.94, and 10.92 for the Swiss-roll, the helicoid, and Dini’s surface, respectively. These values can be compared with those obtained on three sets of 100 random datasets (average $H$ values of 13.80, 11.68, and 11.65, respectively). Statistically significant results ($P < 0.01$) were thus obtained for all three manifolds tested, demonstrating that the KODAMA proximities contain structural information (SI Appendix, Table S2).

Among all tested methods, only KODAMA, ISOMAP, and LLE had the capacity to compute a 2D neighborhood preserving embeddings of the data as shown in Fig. 2A and SI Appendix, Fig. S6. LLE produced an incorrect solution when applied to the Dini’s surface.

**Nonlinear datasets.** In a second series of experiments, KODAMA, LLE, and ISOMAP were applied to spiral datasets with 21 different degrees of noise. For each degree of noise, we created 100 different datasets. To assess the methods’ performance in achieving a low-dimensional representation from a noisy manifold embedded in high-dimensional space, we calculated the coefficient of determination, $r^2$, between the first component of each method and the distribution of each point in the spiral. A high $r^2$ means that the low-dimensional embedding provides an accurate description of the original data. LLE and ISOMAP clearly suffer from problems relative to the “short-circuits” in the neighborhood graph. Short-circuits can lead to low-dimensional embeddings that do not preserve a manifold’s true topology (27). Thus, LLE and ISOMAP performed poorly compared with KODAMA, as shown in Fig. 2B, whereas the other methods (i.e., DM, PCA, RF, SAMMON, SPE, and t-SNE) do not show the capability to compute a monodimensional neighborhood preserving embeddings of the data. Tests on noisy Swiss-roll and helicoid datasets were also performed and are described in SI Appendix, Fig. S7. KODAMA performs better than other methods on the Swiss-roll dataset, and it is comparable to ISOMAP on the helicoid dataset.

**Gaussian datasets.** Most dimensionality reduction methods fail to preserve clusters (28). In multiclass data, ISOMAP and LLE cannot lead to successful embedding owing to unconnected subgraphs. These methods fail if data lie on disconnected manifolds. For further comparison, 100 datasets were generated with three clusters and dimensionalities ranging between 2 and 100. The number of data points for each cluster ranged between 50 and 200. Each cluster was created from a different multivariate normal distribution with a different covariance matrix of variables (29). Each covariance matrix was randomly generated with values that ranged between 0 and 1. The performance of each feature extraction method was analyzed by estimating the relative cluster overlap using the Davies–Bouldin index (DBI) (30), a function of the ratio of the sum of within-cluster scatter to between-cluster separation. Small values of DBI correspond to clusters that are compact and whose centers are far away from each other. The lowest value indicates the best solution. KODAMA achieved the best results compared with other methods, as shown in the box-and-whiskers plot in Fig. 2C.

The results of KODAMA, DM, PCA, RF, SAMMON, SPE, and t-SNE on datasets with different degrees of separation between clusters are also compared. DBI was used to quantify the degree of separation between the original clusters, and the
performances of the methods were evaluated by the DBI of the outputs. KODAMA showed the lowest DBI independently of the DBI (i.e., cluster separation) of the original data. The obtained DBI values for the various methods are reported in SI Appendix, Fig. S8 as a function of the DBI values of the raw data. Moreover, we also show the results of KODAMA when applied to datasets with a continuous distribution of data points (i.e., single multivariate Gaussian distribution, test-1 and test-2 in SI Appendix, Fig. S5). In these tests, KODAMA correctly showed nonstatistically significant results.

Frequently, missing values occur in real-life experiments. KODAMA can handle missing data: A detailed procedure is provided in SI Appendix. SI Appendix, Fig. S9 shows that KODAMA has intermediate performance, behaving somewhat less well with respect to ISOMAP, PCA, and RF, compared with SAMMON and SPE, and significantly better than LLE, DM, and t-SNE.

Comparative Tests of KODAMA on Experimental Datasets. Lymphoma dataset. KODAMA was tested on an experimental dataset (31) that is a popular benchmark for statistical analysis programs. This dataset consists of gene expression profiles of the three most prevalent adult lymphoid malignancies: diffuse large B-cell lymphoma (DLBCL), follicular lymphoma (FL), and B-cell chronic lymphocytic leukemia (B-CLL). The source study produced gene expression data for \( f = 4,682 \) genes in \( n = 62 \) mRNA samples: 42 samples of DLBCL, 9 samples of FL, and 11 samples of B-CLL. In the present work, it is assumed that the lymphoma data are unsupervised (i.e., the number of classes and the class of each sample are not given a priori). We imputed missing values and standardized the data as described in ref. 32.

KODAMA performed with \( k \)-NN identified three classes. A separation between DLBCL and FL/B-CLL is clearly apparent in the first two components, whereas FL and B-CLL can be distinguished in the third component. From this unsupervised KODAMA analysis, we may conclude that the lymphoma data consist primarily of two classes (DLBCL and FL/B-CLL) and that FL and B-CLL are secondary classes, confirming the results obtained in a previous study (5). KODAMA performed with SVM shows a clear separation of the three different malignancies, as does LLE and at variance with PCA and ISOMAP (Fig. 3A). The other methods are shown in SI Appendix, Fig. S10. The DBIs are reported in Fig. 4 and in SI Appendix, Table S6. Whereas KODAMA with either \( k \)-NN or SVM achieved statistically significant results (\( P < 0.01 \)) and comparable values of \( H \) (respectively 8.05 and 8.09), KODAMA with PCA-CA-\( k \)-NN showed a higher and not statistically significant value of 8.25 (\( P = 0.35 \)), compared with an averaged \( H \) of 8.26 obtained on 100 random datasets. In terms of accuracy with respect to the biological classification, LLE and KODAMA with SVM yield no misclassifications, and KODAMA with \( k \)-NN yields one misclassified sample. PCA, ISOMAP, and the other methods tested (SI Appendix, Fig. S10) perform less well.

Metabolomic dataset. The global analysis of metabolites in biological fluids, tissues, or related biological samples is a promising area of research, owing to its potential relevance for human health. To examine KODAMA in this context, we address the task of clustering a dataset of NMR spectra of urines (9). The data belong to a cohort of 22 healthy donors (11 male and 11 female) where each provided about 40 urine samples over the time course of approximately 2 mo, for a total of \( n = 873 \) samples and \( f = 416 \) variables (9). KODAMA was performed with \( k \)-NN, SVM, and PCA-CA-\( k \)-NN. Moreover, KODAMA was initialized with different class-indicator vectors \( W \). We obtained the lowest value of \( H \) in KODAMA with PCA-CA-\( k \)-NN and \( W \) initialized by \( k \)-medoids (with \( k = 22 \)) to the KODAMA dissimilarity matrix \( D_K \); we obtained only 10.7% of misclusterized samples in the urine dataset. The performances of the different unsupervised clustering methods were compared with the adjusted Rand index (ARI) (SI Appendix) (33), a function that measures similarity between two classifications. ARI spans from −1 to 1; perfect agreement is scored 1, whereas 0 corresponds to a random partition. Negative values indicate less agreement than expected by chance. In all cases the presence of 22 clusters was imposed. The ARI value for KODAMA was 0.769. The other methods tested (34–39) (SI Appendix) provided ARI values ranging from 0.439 to 0.212 (SI Appendix, Table S7).

We also applied KODAMA in a semisupervised context, by providing the information regarding sample groupings from each
individual in such a way that the spectra belonging to each individual were forced to maintain the same classification. No information about sex was provided. We performed a PCA-CA analysis as described in ref. 9: The data were projected into their PCA subspace representing 90.0% of the variance, and the resulting PCA scores were projected into the 2D CA subspace. SI Appendix, Fig. S12 shows the result obtained by PCA-CA and KODAMA. We observed that in terms of showing a clear sex separation the semi-supervised KODAMA performed better than PCA-CA.

Knowledge Discovery by KODAMA. Early-type galaxies datasets. We next explored two early-type galaxies (ETGs) datasets, available from the ATLAS3D project (40). The ATLAS3D project combines a multiwavelength survey of a complete set of ETGs. Various parameters are collected, such as the largest equivalent aperture radius ($R_{\text{max}}$), the moment ellipticity measured within one effective radius $Re$ and one-half effective radius $Re/2$ (respectively $e_\text{v}$ and $e_\text{v}/2$), and other photometric and integral-field spectroscopic parameter (i.e., $V/\sigma_\text{v}$, $V/\sigma_\text{v}/2$, $\lambda_{\text{Re}/2}$, and $\lambda_{\text{Re}/2}$ indexes). A dataset contains the $R_{\text{max}}$, $e_\text{v}$, $V/\sigma_\text{v}$, and $\lambda_{\text{Re}/2}$ parameters; the other dataset contains $R_{\text{max}}$, $e_\text{v}/2$, $V/\sigma_\text{v}/2$, and $\lambda_{\text{Re}/2}$. Recent work by Emsellem et al. (41) shows how these parameters can be used to define a refined and optimized criterion for disentangling the so-called fast rotators (FRs) and slow rotators (SRs). SI Appendix, Figs. S10 and S13 show the comparison between KODAMA and other unsupervised feature extraction methods. In both ETGs datasets, KODAMA achieved excellent results, comparable only to those of ISOMAP (Fig. 4 and SI Appendix, Table S6). KODAMA correctly highlighted the differences between FRs and SRs, further suggesting that the SRs may be part of a larger well-defined cluster, with some exceptions falling outside the boundaries.

State of the Union dataset. Sparse data, in which each individual record contains values only for a small fraction of attributes, present a challenge for data mining methods. An interesting case study is offered by the annual addresses presented by the presidents of the United States to the Congress (the “State of the Union” speech), which are available from The American Presidency Project repository. The State of the Union speeches have been the subject of numerous linguistic analyses (ref. 42 and references therein). We selected only the spoken, not written, addresses from 1900 until the sixth address by Barack Obama in 2014. Punctuation characters, numbers, words shorter than three characters, and stop-words (e.g., “that,” “and,” and “which”) were removed from the dataset. This resulted in a dataset of $n = 86$ speeches containing $f = 834$ different meaningful words each. Term frequency-inverse document frequency (TF-IDF) (43) was used to get the feature vectors for the unsupervised analysis. It is often used as a weighting factor in information retrieval and text mining. The TF-IDF value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to control for the fact that some words are generally more common than others.

KODAMA was performed with $k$NN, SVM, and PCA-CA-$k$NN. We found the lowest value of $H$ in KODAMA with $k$NN. The first component of MDS applied to the KODAMA dissimilarity matrix (Fig. 5) shows a single, clear, and abrupt transition over a period of more than 100 y, which occurred during the presidency of Ronald Reagan. The pre-Reagan and post-Reagan speeches are recognized with 100% cross-validated accuracy using the $k$NN classifier. Words such as “labor,” “expenditures,” “employment,” “relations,” “resources,” and “production” suddenly decrease in frequency, whereas words such as “parents,” “students,” “pass,” “children,” “Medicare,” and “reform” suddenly increase in frequency (Fig. 5, inset). It can be noted that the third address of G. Bush (January 29, 1991) is somewhat different from the other Bush’s addresses and marks a partial reversal to the pre-Reagan style. Interestingly, this speech was held in the middle of Operation Desert Storm (January 17, 1991–February 28, 1991) and probably reflects the emotional atmosphere of the nation.

It is noteworthy that if KODAMA is performed in a semi-supervised way, by providing information grouping together the speeches of each president, Reagan seems to represent a “hinge” between past and present rhetorical modes. The performances of the other feature extraction methods are lower when compared on the basis of the pre- and post-Reagan discrimination (SI Appendix, Table S6). The results of each method are shown in SI Appendix, Fig. S14. Interestingly, no distinction between Republicans and Democrats is evident. Although it is widely accepted that Reagan’s rhetoric was unique and that it influenced...
all subsequent speeches—in the words of Obama, Ronald Reagan “changed the trajectory of America in a way that, you know, Richard Nixon did not and in a way that Bill Clinton did not” (44)—our analysis clearly points to a sharp change of Reagan’s rhetoric during his presidency, and more precisely toward the end of his first mandate.

**Discussion**

Some limitations of conventional algorithms, such as PCA (12), stem from the fact that they use imposed distance measures defined in a globally linear space or with limited degrees of freedom. Linear methods are usually not appropriate to modeling curved manifolds, because they focus on preserving the distances between widely separated data points rather than on preserving the distances between nearby data points. Nonlinear dimensionality reduction methods (11, 13) are capable of discovering nonlinear degrees of freedom (11) but are negatively affected by increasing dimensionality of the embedded manifold (the so-called curse of dimensionality) (45) and by the problem of short-circuit edges in the presence of noisy or sparse data (27).

To illustrate another type of drawback, t-SNE reduces the dimensionality of data in a manner dependent on the local properties of that data. This makes t-SNE likewise sensitive to the curse of dimensionality of the data (17). Manifold learners such as ISOMAP and LLE suffer from precisely the same problem (17). Even a single short-circuit error (27) can alter many entries into the neighborhood graph, which in turn can lead to a drastically different and incorrect low-dimensional embedding.

We thus propose KODAMA as a method of performing feature extraction on noisy and high-dimensional data. Thus, we have demonstrated its performance on real datasets chosen for their different structures and properties. Our approach differs from previous methods in that it is based on an integrated procedure of validation of the results through an embedded MC procedure that maximizes cross-validated accuracy. Overall, KODAMA outperformed the existing feature extraction methods that we tested. KODAMA offers a general framework for analyzing any kind of complex data in a broad range of sciences. It also makes it possible to perform analyses in unsupervised or semi-supervised contexts.

Finally, its ability to resolve meaningful clusters within the data makes the KODAMA dissimilarity matrix useful in conjunction with classical clustering algorithms (e.g., k-medoids), because it strongly improves their performances.

**ACKNOWLEDGMENTS.** We thank S. Tyekucheva, A. K. Smilde, E. Saccini, C. Sander, and M. A. Andrade-Navarro for support and several critical discussions. We also thank E. Emselfelt for expressing interest in our analysis of the early-type galaxies datasets, and the anonymous reviewer of the earlier versions of this manuscript for the many constructive criticisms and suggestions that allowed us to significantly improve the paper. This work was partly supported by the European Commission-funded FP7 projects COSMOS (312341), WehMR (261572), CHANCE (266331), BioMedBridges (284209), and by the Programmi di Ricerca Nazionale (2009FAKHZT_001). S.C. was supported by a fellowship from Fondazione Italiana per la Ricerca sul Cancro.

Supporting Information

Use of supervised classifiers in step I-II and cross validation.

$k$NN. Cross-validated predictive accuracies were calculated on data by $k$NN using the R library “knnflex”. Data classification was conducted by applying the $k$NN ($k=10$).

PLS. Projection to Latent Structure (PLS) was applied on processed data for dimension reduction using the classical SIMPLS algorithm (1) as implemented in the R library “plsgenomics”.

SVM. The Support Vector Machines (SVM) method (2) was used for data classification using the “libsvm” module of the R library “e1071”. Data classification was conducted by applying the SVM with linear kernel on the first 5 components of PLS.

PCA-CA-$k$NN. Cross-validated predictive accuracies were calculated on data by combining established methods. Principal Component Analysis (PCA) was applied on data for dimension reduction using the standard R function “prcomp”. Canonical Correlation analysis (CA) was conducted using the standard R function "cancor". The $k$-Nearest Neighbor ($k$NN) method was used for data classification using the R library “knnflex”. The data were projected into a PCA subspace explaining 90.0% of the variance. The resulting PCA score matrix was projected into the CA subspace. Data classification was conducted by applying the $k$NN ($k=10$) on the components of the PCA-CA subspace.

Cross-validation is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model. In typical cross-validation, the training and validation sets must crossover in successive rounds such that each data point has a chance of being validated against. The basic form of cross-validation is $k$-fold cross-validation, and this is the form of cross-validation used in KODAMA.

In $k$-fold cross-validation the data is first partitioned into $k$ equally (or nearly equally) sized segments or folds. Each of these $k$ subsets serves in turn as a test set. For each of these $k$ test sets, a classifier is trained on the remaining $k-1$ folds (the training set). The trained classifier is then used to classify the samples in the test set, and the accuracy is calculated. The combined value of the accuracy over the $k$ test sets, which is based on the prediction of all samples one time each, is the cross-validated estimate of that error. Leave-One-Out Cross-Validation (LOOCV) is a special case of $k$-fold cross-validation where $k$ equals the number of samples in the data. In other words in each iteration, nearly all the data except for a single sample are used for training and the model is tested on that single sample. An accuracy estimate obtained using LOOCV is
known to be almost unbiased but it has high variance, leading to unreliable estimates (3).

A large $k$ is seemingly desirable, since with a larger $k$ there are more performance estimates, and the training set size is closer to the full data size. As $k$ increases, however, the overlap between training sets also increases, leading to less precise, less fine-grained measurements of the performance metric. In the other hand small $k$ speeds up the computation velocity of the cross-validation but the training sets are far to represent the full data.

In KODAMA, we implemented the 10-fold cross-validation because it makes classifications using 90% of the data, making it more likely to be generalizable to the full data; it has been shown to be sufficient to achieve stable values of the accuracy (4). These competing factors have all been considered and $k=10$ seems to be a good compromise.

**Optimization of the $\phi$, $T$, $M$, and $\varepsilon$ parameters.**

Except for the choice of the classifier to use, KODAMA has three different parameters that can slightly affect the analysis: the number of samples $\phi$ to select in the maximization of the cross-validated accuracy part, the number of Monte Carlo (MC) steps, $T$, and the number of iteration of steps I-III, $M$.

We optimized these three parameters on the basis of the results achieved on three different datasets using KODAMA with $k$NN as classifier. The first dataset was generated with 3 clusters and 5 dimensions. The number of data points for each cluster is 50. Each cluster is created from a different multivariate normal distribution with a different covariance matrix of the features (5). Each covariance matrix was randomly generated with values that range between 0 and 1. The second dataset is the Swiss-roll with 1000 data points described by the following parametric equations: $x=ucos(u)$, $z=usin(u)$; where $u$ varies between $1.5\pi$ and $4.5\pi$, and $y$ varies between 0 and 21. The third dataset is a spiral with 200 data points defined by the following parametric equation: $x=cos(u)\times(u+a)$, $y=sin(u)\times(u+a)$; where $u$ varies between $\pi$ and $4\pi$, and $a$ is a value from a Gaussian distribution with mean=0 and standard deviation =0.7.

The MC procedure optimizes the vector $W$ by maximizing $A_W$ through a defined number $T$ of iterations. At each step $A_W$ can increase or at least remain equal. The parameter $T$ defines the number of loops that MC procedures does to optimize the cross-validated accuracy. In Fig. S4, we show how the accuracy $A_W$ evolves during the MC procedure in the three different datasets. After 10 loops, the accuracy $A_W$ achieves the maximum value and the characteristic of the dataset seems to not interfere with the maximization procedure. A good compromise between computational time and quality of maximization is $T=20$. In several cases the
MC procedure reaches 100% accuracy before completing $T$ iterations.

The parameter $M$ is the number of times that maximization of the cross-validated accuracy part is repeated. This part is repeated $M$ times to average effects due to the randomness of the iterative procedure and sample selection. Larger $M$ value provides a better description of the distribution of the data. We calculated the residual variance defined as $r^2(A_{200}, A_X)$, where $A_{200}$ is the KODAMA dissimilarity matrix obtained with $M=200$, and $A_X$ is the KODAMA dissimilarity matrix obtained with the different $M$ value tested. The differences between the results with $M=100$ and $M=200$ are low. We conclude that $M=100$ can be a good approximation a larger $M$ value (Fig. S4).

A subset $N' = \phi N$ of samples is selected at the beginning of the maximization of the cross-validated accuracy part. Each time that this part of the KODAMA is repeated a different subset of samples is used. The choice of the number of samples to be used can affect the analysis. The residual variance between the Euclidean distance of data points in the manifold and the KODAMA dissimilarity matrix is used to analyze the data in the Swiss-roll and the spiral datasets. Davies-Bouldin Index (DBI) was used to analyze the results in the 3-clusters dataset. With low $\phi$ values KODAMA cannot achieve a good representation of the manifold embedded, whilst KODAMA suffers from problems relative to the “short-circuits” in the neighborhood graph for $\phi$ approaching 1. These competing factors have all been considered and $\phi=0.75$ seems to be a good compromise (Fig. S4).

The problem of “short-circuits” emerges also if occasional proximities are taken as meaningful while calculating the final KODAMA dissimilarity matrix using the Floyd’s algorithm. We have found that short circuits can be removed by setting equal to zero all $P_M$ values below a certain threshold $\varepsilon$. Empirically, a value of $\varepsilon$ of 0.05 eliminates “short-circuits” without introducing fragmentation of the manifold.

**Significance of the KODAMA result**

The Shannon Entropy ($H$) (6) can be used to assess the significance of the KODAMA result on a high-dimensional dataset. $H$ is given by:

$$H = -\sum_i \sum_j v(i, j) \times \log v(i, j)$$

where $v(i, j)$ is $p_M(i, j)$ divided by the sum of all the values in the matrix $P_M = \{p_M(i, j)\}$. $H$ values were calculated using the function “entropy” in the R library “entropy”.

In terms of hypothesis testing, we proposed testing the null hypothesis that the available result can be
modeled as coming from a single multivariate Gaussian distribution. Our test statistic is the $H$ of the $P_M$ proximity matrix. Simulation of the test statistic is used to produce percentile-based $p$ values. This test assumes that there is no difference in $H$ value between two groups of $P_M$ proximity matrices obtained from two randomly formed datasets. In this test, KODAMA is performed on data from a single multivariate Gaussian distribution generated with the same number of samples and variables, and the same covariance matrix of the original data. From each proximity matrix, a $H$ value is obtained. By repeating the procedure $G$ times, a null distribution of $H$ values is obtained. $H_0$ is then defined as a distribution of $H$ values that are expected to be insignificant. Statistical significance of KODAMA is then assessed by relating the $H$ value of the KODAMA performed on the original data to the distribution of the $H$ values obtained from KODAMA performed on the multivariate Gaussian distributions. The $p$ value is calculated as the number of $H$ values from the distribution of random data that are smaller or identical to the $H$ value from the original data divided by $G$. The lower limit of the number $G$ is dictated by the required statistical significance: for instance, to attain a $p < 0.01$ at least $G=100$ is necessary but may not be sufficient for a proper sampling of the distributions tails.

We compared the $H$ values obtained on 6 different datasets: Swiss-roll, Helicoid, Dini’s surface, 3-cluster, and 2 different continuous distribution datasets (Test-1 and Test-2). The last two datasets are generated from a single multivariate Gaussian distribution and they differ for the degree of correlation between variables. In the first one, the variables are not correlated to each other, whilst the second one is generated using a covariance matrix of the variables. The covariance matrix was randomly generated with values that range between 0 and 1. Three hundreds samples were simulated with 10 variables for both datasets. In Fig. S5, we show the KODAMA proximity matrix and the visualization with Multi-Dimensional Scaling of the KODAMA dissimilarity matrix obtained for four of these datasets.

In Table S2, for each dataset we report the $H$ values obtained form KODAMA proximity matrices and $H$ values obtained from the simulation on 100 datasets generated from a Gaussian distribution with the same covariance matrix. Correctly KODAMA identify as not significant the results for Test-1 and Test-2 datasets, and significant the results for the other datasets.

**Feature extraction methods**

For each feature extraction methods, the number of dimensions in the output space was chosen a priori equal to number of classes minus one if the number of classes is major than two, two otherwise.

Multi-Dimensional Scaling (MDS) was performed using the function “cmdscale” in the R library “stats”. Diffusion Maps (DM) (7) was performed using the function “diffuse” in the R library “diffusionMap”.

4
The parameter “\textit{eps.val}” controls the degree of localness in the diffusion weight matrix. We used the default function to optimize “\textit{eps.val}”. Isometric Feature Mapping Ordination (ISOMAP) (8) was performed using the function “\textit{ISOMAP}” in the R library “\textit{vegan}”. We performed ISOMAP optimizing the suitable neighborhood size through estimating the “quality” of the corresponding mapping, \textit{i.e.} how well the high-dimensional structure is represented in the embedded space, measured by the residual variance. Principal Component Analysis (PCA) (9) was performed using the function “\textit{prcomp}” in the R library “\textit{stats}”. Locally Linear Embedding (LLE) (10) was performed using the function “\textit{lle}” in the R library “\textit{lle}”. The optimal number of neighbours was calculated by using the algorithm proposed by Kayo (11) described in the function “\textit{calc_k}” in the R library “\textit{lle}”. Random Forest (RF) (12) was performed using the function “\textit{randomForest}” in the R library “\textit{randomForest}”. The number of trees was 2000. A higher number of trees should increase the performance of RF. We optimize this parameter \textit{a posteriori} on the basis of the results obtained. MDS was applied on one minus the proximity matrix achieved by RF (13). Sammon's Non-Linear Mapping (Sammon) (14) was performed using the function “\textit{sammon}” in the R library “\textit{MASS}”. We leave unchanged the default parameters. Stochastic Proximity Embedding (SPE) (15) was performed using the function “\textit{spe}” in the R library “\textit{spe}”. We leave unchanged the default parameters. \textit{t}-Distributed Stochastic Neighbor Embedding (\textit{t}-SNE) (16) was performed using the function “\textit{tsne}” in the R library “\textit{tsne}”. We defined the perplexity parameter on the basis of the number of data points. Tree preserving embedding (TPE) (17) was performed using the function “\textit{tpe}” in the R library “\textit{tpe}”.

The performance of each feature extraction method was analyzed by estimating the relative class overlap using the Davies-Bouldin Index (DBI) (18), a function of the ratio of the sum of within-cluster scatter to between-cluster separation, as implemented in the function “\textit{DBIndex}” in the R library “\textit{RDRToolbox}”. DBI is defined as

\[
DBI = \frac{1}{nc} \sum_{i=1,i \neq j}^{\infty} \max \left[ \frac{\sigma_i + \sigma_j}{d(c_i,c_j)} \right]
\]

where \(nc\) is the number of clusters, \(\sigma_i\) is the average distance of all samples in cluster \(i\) to their cluster center \(c_i\), \(\sigma_j\) is the average distance of all samples in cluster \(j\) to their cluster center \(c_j\), and \(d(c_i,c_j)\) is the distance of cluster centers \(c_i\) and \(c_j\). Small values of DBI correspond to clusters that are compact, and whose centers are far away from each other.

**Clustering methods**

High-Dimensional Data Clustering (HDDC) (19) was performed using the function “\textit{hdde}” in the R library “\textit{HDeClassif}”. Models and parameters were chosen on the basis of the maximization of the BIC criterion.
Spectral Clustering (SC) (20) was performed using the function “specc” in the R library “kernlab”. Radial Basis kernel function was used. Spectral clustering based on k-nearest neighbor graph (SCKNN) (21). Clustering was performed using the function “specClust” in the R library “kknn”. The number of neighbors considered was 15. Hierarchical Clustering (HC) (22) was performed using Euclidean distance and Ward’s method of aggregation. Clustering was performed using the function “hclust” in the R library “stats”. k-medoids clustering (23) was performed using the function “pam” in the R library “cluster”. k-means clustering was performed using the function “kmeans” in the R library “stats”. The maximum number of iterations allowed was 100, and 100 random sets were chosen. Affinity Propagation (AP) (24) was performed using the function “apclusterK” in the R library “apcluster”.

Adjusted Rand Index (ARI) (25) was used to compare the performances of different methods. ARI is frequently used in cluster validation since it is a measure of agreement between two partitions. ARI was calculated using the function “adjustedRandIndex” of the R library “mclust”.

Non-linear datasets

The problem of non-linear dimensionality reduction is illustrated in Fig. 2A and in Fig. S6 for three-dimensional data sampled from two-dimensional manifolds. The Swiss-roll is described by the following parametric equations: \( x = u \times \cos(u) \), \( z = u \times \sin(u) \); where \( u \) varies between \( 1.5\pi \) and \( 4.5\pi \), and \( y \) varies between 0 and 21. The Helicoid is described by the following parametric equation: \( x = p \times \cos(u) \); \( y = p \times \sin(u) \); \( z = u \); where \( u \) varies between \(-\pi\) and \( \pi \), and \( p \) varies between \(-1\) and \( 1\). The Dini’s surface is described by the following parametric equation: \( x = \cos(u) \times \sin(v) \); \( y = \sin(u) \times \sin(v) \); \( z = \cos(v) + \log[\tan(v/2)] + u/5 \); with \( 0 \leq u \leq 4\pi \) and \( 0.01 \leq v \leq 1.00 \) and constants \( a = 1.0 \) and \( b = 0.2 \). The spiral datasets described in Fig. 2B are defined by the following parametric equation: \( x = \cos(u) \times (u+a) \); \( y = \sin(u) \times (u+a) \); where \( u \) varies between \( \pi \) and \( 4\pi \), and \( a \) is a value sampled from a Gaussian distribution with mean=0 and deviation standard between 0 and 2 (21 samplings). For each different Gaussian distribution, 100 different datasets were created. The parameter \( u \) describes the position of a point in the body of the spiral. Lower \( u \) values correspond to data points located in the center of the spiral whilst higher \( u \) values correspond to data points located in the external part. The coefficient of determination, \( r^2 \), between the first component of each method and the \( u \) values was used to evaluate the performance of each method. A higher \( r^2 \) means that the low dimensional embedding provides an accurate description of the original data.

Missing values

Real life experiments can often generate missing values. No really satisfactory solution exists for missing data, which is why it is important to try to maximize data collection. The main ways of handling missing
data in analysis are: i) omitting variables which have many missing values; ii) omitting samples which do not have complete variables; and iii) estimating (imputing) what the missing value were. Obviously, estimating associations using an incomplete dataset remains less efficient (i.e., imprecise), because part of the data is not available. Sometimes data are missing in a predictable way that does not depend on the missing value itself but which can be predicted from other data. KODAMA applies \( k \)NN imputation \((26)\) on missing values in the initial step of the algorithm. For each feature with missing values, \( k \)NN imputation finds the \( k \)NN using a Euclidean metric, confined to the columns for which that feature is not missing. Each candidate neighbor might be missing some of the coordinates used to calculate the distance. In this case, \( k \)NN imputation averages the distance from the non-missing coordinates. Having found the \( k \)NN for a feature, \( k \)NN imputation imputes the missing values by averaging those non-missing elements of its neighbors.

To evaluate the performance of \( k \)NN imputation applied on KODAMA, missing values were randomly generated on simulated datasets. Different degrees of missing values were tested (i.e., 5\%, 10\%, 15\%, 20\%, and 25\%). We simulated 20 datasets for each degree of missing values. The datasets were generated with 3 clusters and 5 dimensions. The number of data points for each cluster is 30. Each cluster is created from a different multivariate normal distribution with a different covariance matrix of the variables. Each covariance matrix was randomly generated with values that range between 0 and 1. The performance of \( k \)NN imputation was analyzed by calculating the residual variance \( r^2(K_i,K) \), where \( K_i \) is the KODAMA dissimilarity matrix of the data with the missing values estimated by \( k \)NN imputation, and \( K \) is the KODAMA matrix of the original data. Fig. S9 shows the results.

**Computational time complexity**

We measured computational time experimentally on a desktop machine with a 3.06 GHz Intel Core 2 Duo and with 4 GB of 1067 MHz RAM. We used the R version 3.0.2 (2013-09-25) -- “Frisbee Sailing”.

The computational complexity of the KODAMA is dominated by the multiple iterations of the cross-validation procedure and, thus, it depends proportionally on the product of the number of cross-validations performed by the time complexity of the classifier used. Among the classifiers tested, the \( k \)NN classifier has the lowest time complexity. The simplest \( k \)NN algorithm has a time complexity of \( O(n \times m \times f) \), where \( n \) and \( m \) are the number of data points of the training and test set, respectively, and \( f \) is the dimensionality of the dataset. In the case of a 10-fold cross-validation \( n=0.9 \times N' \) and \( m=0.1 \times N' \), where \( N' \) is the overall number of data points \((N'=\varphi N)\), and the time complexity of the \( k \)NN classifier is therefore \( O(0.09 \times N'^2 \times f) \). A 10-fold cross-validation performed with \( k \)NN classifier has thus a time complexity of \( O(0.9 \times N'^2 \times f) \). KODAMA consequently has a time complexity at most of \( O(0.9 \times M \times T \times N'^2 \times f) \), where \( M \) is the number of times that the
maximization of the cross-validated accuracy is repeated, and \( T \) is the maximum number of MC iterations.

Running time (in seconds) of each classifier implemented in KODAMA (\( i.e. \), \( k \)NN, SVM, and PCA-CA-\( k \)NN) was provided for different datasets, varying the number of samples and variables.

The time complexity of the SVM and PCA-CA-\( k \)NN classifiers is \( O(N^{2sf}xnc) \), where \( nc \) is the number of classes present in the cross-validation.

We experimentally compare the time complexity of KODAMA with the other methods on datasets with 5 variables and different number of data points (\( i.e. \), 50, 100, 200, 500, and 1000) and on datasets with 50 data points and a different number of variables (\( i.e. \), 50, 100, 200, 500, and 1000), respectively, as shown in Table S3 and S4.

Running time of each classifier was also calculated for some of datasets tested (\( i.e. \), Swiss-roll, Lymphoma, Metabolomic, Early-Type Galaxies, and The State of the Union). The results are shown in Table S5.
References

Table S1. KODAMA adjustable parameters with default values.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi$</td>
<td>Fraction of samples randomly taken at the beginning of step I</td>
<td>0.75</td>
</tr>
<tr>
<td>$T$</td>
<td>Maximum number of MC iterations</td>
<td>20</td>
</tr>
<tr>
<td>$M$</td>
<td>Repetitions of the iterative procedure in steps I-II-III</td>
<td>100</td>
</tr>
<tr>
<td>$\varepsilon$</td>
<td>Threshold for not significant proximities</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Table S2. $H$ value obtained with KODAMA performed with kNN on six different datasets: Swiss-roll, Helicoid, Dini’s surface, 3-clusters datasets, and two different homogenous datasets with correlation and without correlation among the variables. Mean, standard deviation, and range of $H$ values obtained with KODAMA performed with kNN on 100 homogenous datasets created with the same covariance matrix are reported. $p$ values are also reported.

<table>
<thead>
<tr>
<th></th>
<th>$H$ value</th>
<th>Mean</th>
<th>St. dev.</th>
<th>Min</th>
<th>Max</th>
<th>$p$ value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swiss-roll</td>
<td>11.391174</td>
<td>13.802700</td>
<td>13.799552</td>
<td>13.805183</td>
<td>&lt;0.01</td>
<td></td>
</tr>
<tr>
<td>Helicoid</td>
<td>10.942682</td>
<td>11.683730</td>
<td>11.683730</td>
<td>11.683750</td>
<td>&lt;0.01</td>
<td></td>
</tr>
<tr>
<td>Dini’s surface</td>
<td>10.921598</td>
<td>11.654730</td>
<td>11.654720</td>
<td>11.654740</td>
<td>&lt;0.01</td>
<td></td>
</tr>
<tr>
<td>3-clusters</td>
<td>11.260296</td>
<td>11.369976</td>
<td>11.359756</td>
<td>11.384299</td>
<td>&lt;0.01</td>
<td></td>
</tr>
<tr>
<td>Test-1</td>
<td>11.384031</td>
<td>11.376065</td>
<td>11.359756</td>
<td>11.396324</td>
<td>0.73</td>
<td></td>
</tr>
<tr>
<td>Test-2</td>
<td>11.397000</td>
<td>11.394322</td>
<td>11.380627</td>
<td>11.405440</td>
<td>0.69</td>
<td></td>
</tr>
</tbody>
</table>

Table S3. Running times (seconds) of each method performed on homogenous Gaussian datasets with 5 variables and different numbers of data points.

<table>
<thead>
<tr>
<th></th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>500</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>KODAMA (kNN classifier)</td>
<td></td>
<td>109.199</td>
<td>244.892</td>
<td>512.025</td>
<td>1434.554</td>
</tr>
<tr>
<td>KODAMA (SVM classifier)</td>
<td></td>
<td>152.425</td>
<td>179.740</td>
<td>276.022</td>
<td>605.145</td>
</tr>
<tr>
<td>KODAMA (PCA-CA-kNN classifier)</td>
<td></td>
<td>120.799</td>
<td>246.311</td>
<td>497.037</td>
<td>1327.146</td>
</tr>
<tr>
<td>DM</td>
<td>0.099</td>
<td>0.121</td>
<td>0.153</td>
<td>0.387</td>
<td>1.738</td>
</tr>
<tr>
<td>ISOMAP</td>
<td>0.451</td>
<td>1.289</td>
<td>6.713</td>
<td>85.160</td>
<td>783.752</td>
</tr>
<tr>
<td>PCA</td>
<td>0.002</td>
<td>0.001</td>
<td>0.004</td>
<td>0.002</td>
<td>0.001</td>
</tr>
<tr>
<td>LLE</td>
<td>2.421</td>
<td>5.441</td>
<td>12.950</td>
<td>62.539</td>
<td>309.378</td>
</tr>
<tr>
<td>RF</td>
<td>0.159</td>
<td>0.362</td>
<td>0.863</td>
<td>3.178</td>
<td>10.428</td>
</tr>
<tr>
<td>SAMMON</td>
<td>0.007</td>
<td>0.030</td>
<td>0.102</td>
<td>0.394</td>
<td>3.751</td>
</tr>
</tbody>
</table>
Table S4. Running times (seconds) of each method performed on homogenous Gaussian datasets with 50 data points and different numbers of variables.

<table>
<thead>
<tr>
<th>Number of variables</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>500</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>KODAMA (kNN classifier)</td>
<td>64.388</td>
<td>74.547</td>
<td>64.607</td>
<td>62.373</td>
<td>66.270</td>
</tr>
<tr>
<td>KODAMA (SVM classifier)</td>
<td>162.551</td>
<td>182.134</td>
<td>194.097</td>
<td>254.174</td>
<td>264.176</td>
</tr>
<tr>
<td>KODAMA (PCA-CA-kNN classifier)</td>
<td>102.248</td>
<td>79.182</td>
<td>76.313</td>
<td>62.370</td>
<td>98.220</td>
</tr>
<tr>
<td>DM</td>
<td>0.096</td>
<td>0.097</td>
<td>0.099</td>
<td>0.101</td>
<td>0.102</td>
</tr>
<tr>
<td>ISOMAP</td>
<td>0.318</td>
<td>0.321</td>
<td>0.318</td>
<td>0.324</td>
<td>0.324</td>
</tr>
<tr>
<td>PCA</td>
<td>0.003</td>
<td>0.004</td>
<td>0.006</td>
<td>0.011</td>
<td>0.019</td>
</tr>
<tr>
<td>LLE</td>
<td>3.342</td>
<td>3.832</td>
<td>4.701</td>
<td>7.715</td>
<td>11.898</td>
</tr>
<tr>
<td>RF</td>
<td>0.661</td>
<td>1.173</td>
<td>2.173</td>
<td>5.285</td>
<td>10.490</td>
</tr>
<tr>
<td>SAMMON</td>
<td>0.005</td>
<td>0.014</td>
<td>0.011</td>
<td>0.014</td>
<td>0.004</td>
</tr>
<tr>
<td>SPE</td>
<td>13.412</td>
<td>23.446</td>
<td>53.697</td>
<td>163.317</td>
<td>319.925</td>
</tr>
</tbody>
</table>

Table S5. Running times (seconds) of each method applied to Swiss-roll, Lymphoma, Metabolomic, ETGs and State of the Union datasets.

<table>
<thead>
<tr>
<th>Data points</th>
<th>Swiss-roll</th>
<th>Lymphoma</th>
<th>Metabolomic</th>
<th>ETGs</th>
<th>State of the Union</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
<td>1000</td>
<td>62</td>
<td>873</td>
<td>260</td>
<td>84</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>123.787</td>
<td>4133.893</td>
<td>690.069</td>
<td>164.217</td>
</tr>
<tr>
<td>KODAMA (kNN classifier)</td>
<td>3662.340</td>
<td>1224.028</td>
<td>1922.688</td>
<td>297.947</td>
<td>421.516</td>
</tr>
<tr>
<td>KODAMA (SVM classifier)</td>
<td>1019.067</td>
<td>193.041</td>
<td>297.947</td>
<td>421.516</td>
<td></td>
</tr>
<tr>
<td>KODAMA (PCA-CA-kNN classifier)</td>
<td>3144.494</td>
<td>176.856</td>
<td>4560.366</td>
<td>659.273</td>
<td>193.041</td>
</tr>
<tr>
<td>DM</td>
<td>1.600</td>
<td>0.064</td>
<td>277.738</td>
<td>0.304</td>
<td>0.069</td>
</tr>
<tr>
<td>ISOMAP</td>
<td>519.844</td>
<td>0.486</td>
<td>414.901</td>
<td>12.816</td>
<td>0.871</td>
</tr>
<tr>
<td>PCA</td>
<td>0.006</td>
<td>0.113</td>
<td>1.261</td>
<td>0.001</td>
<td>0.048</td>
</tr>
<tr>
<td>LLE</td>
<td>311.742</td>
<td>57.809</td>
<td>1015.864</td>
<td>24.812</td>
<td>23.301</td>
</tr>
<tr>
<td>RF</td>
<td>8.111</td>
<td>61.781</td>
<td>229.911</td>
<td>1.131</td>
<td>18.357</td>
</tr>
<tr>
<td>SAMMON</td>
<td>2.850</td>
<td>0.008</td>
<td>2.295</td>
<td>0.133</td>
<td>0.024</td>
</tr>
<tr>
<td>SPE</td>
<td>6.439</td>
<td>1394.867</td>
<td>382.616</td>
<td>6.343</td>
<td>270.833</td>
</tr>
<tr>
<td>t-SNE</td>
<td>477.211</td>
<td>407.68</td>
<td>391.755</td>
<td>47.790</td>
<td>15.816</td>
</tr>
</tbody>
</table>
**Table S6.** Comparison among DBIs obtained with different feature extraction methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Lymphoma</th>
<th>Metabolomics</th>
<th>ETGs (εₐ)</th>
<th>ETGs (εₑ)</th>
<th>State of the Union</th>
</tr>
</thead>
<tbody>
<tr>
<td>KODAMA</td>
<td>0.142</td>
<td>0.131</td>
<td>0.913</td>
<td>0.981</td>
<td>0.237</td>
</tr>
<tr>
<td>DM</td>
<td>0.246</td>
<td>0.612</td>
<td>0.934</td>
<td>0.993</td>
<td>0.580</td>
</tr>
<tr>
<td>ISOMAP</td>
<td>0.784</td>
<td>0.408</td>
<td>0.902</td>
<td>0.988</td>
<td>0.356</td>
</tr>
<tr>
<td>PCA</td>
<td>0.307</td>
<td>0.521</td>
<td>0.942</td>
<td>1.134</td>
<td>2.260</td>
</tr>
<tr>
<td>LLE</td>
<td>0.186</td>
<td>0.349</td>
<td>1.256</td>
<td>1.247</td>
<td>0.750</td>
</tr>
<tr>
<td>RF</td>
<td>0.370</td>
<td>0.411</td>
<td>1.475</td>
<td>1.825</td>
<td>0.688</td>
</tr>
<tr>
<td>SAMMON</td>
<td>0.570</td>
<td>0.522</td>
<td>0.947</td>
<td>1.139</td>
<td>1.090</td>
</tr>
<tr>
<td>SPE</td>
<td>0.531</td>
<td>0.529</td>
<td>0.959</td>
<td>1.142</td>
<td>1.090</td>
</tr>
<tr>
<td>t-SNE</td>
<td>1.367</td>
<td>0.331</td>
<td>3.584</td>
<td>3.112</td>
<td>2.423</td>
</tr>
</tbody>
</table>

**Table S7.** ARI values for different clustering methods applied to the Metabolomic data set.

<table>
<thead>
<tr>
<th>Method</th>
<th>ARI</th>
</tr>
</thead>
<tbody>
<tr>
<td>KODAMA</td>
<td>0.769</td>
</tr>
<tr>
<td>k-means</td>
<td>0.330</td>
</tr>
<tr>
<td>high-Dimensional Data Clustering</td>
<td>0.358</td>
</tr>
<tr>
<td>Spectral Clustering</td>
<td>0.336</td>
</tr>
<tr>
<td>Spectral clustering (k-nearest neighbor graph based)</td>
<td>0.439</td>
</tr>
<tr>
<td>Hierarchical Clustering</td>
<td>0.305</td>
</tr>
<tr>
<td>k-medoids</td>
<td>0.317</td>
</tr>
<tr>
<td>Affinity Propagation</td>
<td>0.212</td>
</tr>
</tbody>
</table>
Fig. S1. Flowchart of the first part (steps I-III) of KODAMA.
Fig. S2. The KODAMA accuracy maximization procedure. Each point is colored according to the cluster it belongs to; the circle represents the distance to the second nearest neighbor. $A_W$ values show how the relative cross-validated accuracy increases during the iterative step. Vector $W$ indicates the class. Vector $Z_W$ indicates the predicted values of the classifier built on the base of the vector $W$. 
Fig. S3. The averaging of each element $p(i,j)$ of the $M$ proximity matrices $P$ is performed with the following formula $P_M (i,j) = \frac{\sum_{g=1}^{M} p_g(i,j)}{m(i,j)}$ where $m(i,j)$ indicates the number of times that samples $x_i$ and $x_j$ are present together in the same subdataset generated in step I. Thus, the resulting elements of the matrix $P_M\{p_M(i,j)\}$ ($N \times N$) are averages ranging from 0 to 1.
**Fig. S4.** Parameter optimization. KODAMA was tested on three different datasets to optimize the values of $T$, $M$ and $\varphi$ parameters. The increase in accuracy with increasing $T$ or $M$, respectively, is shown. The best value of the $\varphi$ parameter is more dependent on the type of data.
**Fig. S5.** KODAMA was tested on four different datasets. The first two datasets (*i.e.*, Swiss-roll and 3-clusters) present a clear “organization” in the distribution of the data points. The last two datasets are continuous distributions with correlations (*i.e.*, Test-1) and without correlations (*i.e.*, Test-2) among the variables. In the first column the KODAMA proximity matrices obtained on the respective dataset are reported. In the second column the MDS plot of the KODAMA dissimilarity matrix is shown.
Fig. S6. Comparison between different feature extraction methods on the Swiss-roll, Helicoid, and Dini’s surface. The methods shown are DM, ISOMAP, PCA, LLE, RF, Sammon, SPE, and t-SNE. The color-coding reveals how the data are embedded in two dimensions.
Fig. S7. Performance of achieving a low-dimensional representation from a manifold embedded in high dimensional space as a function of the noise in the Swiss-roll and Helicoid datasets of 500 data points each. KODAMA (in blue), ISOMAP (in yellow), and LLE (in red) were applied to Swiss-roll and Helicoid datasets. The Swiss-roll is described by the following parametric equations in three dimensions: 
\[ x = (u + a_1) \times \cos(u), \quad z = (u + a_2) \times \sin(u); \] 
where \( u \) varies between \( 1.5\pi \) and \( 4.5\pi \), and \( y \) varies between 0 and 10. The values \( a_1 \) and \( a_2 \) are from a Gaussian distribution with mean=0 and standard deviation between 0 and 2 (21 samplings). For each Gaussian distribution, we created 100 different datasets. 

The Helicoid is described by the following parametric equation: 
\[ x = p \times \cos(u); \quad y = p \times \sin(u); \quad z = u + a; \] 
where \( u \) varies between \(-\pi\) and \(\pi\), and \( p \) varies between \(-1\) and \(1\). The value \( a \) is from a Gaussian distribution with mean=0 and standard deviation between 0 and 1 (21 samplings). For each Gaussian distribution, we created 100 different datasets.
Fig. S8. Results of different methods applied to datasets with different degrees of separation among clusters.
Fig. S9. Results of different methods applied to datasets with different degrees of missing values.
Fig. S10. Comparison between DM, RF, Sammon, SPE, and \( t \)-SNE on the Lymphoma, Metabolomic, and ETGs datasets. Data points are colored by their class.
Fig. S11. Metabolomic dataset. Different visualization of KODAMA dissimilarity matrix with MDS, $t$-SNE, and TPE compared to visualization with Euclidean distance matrix with MDS, $t$-SNE, and TPE.
Fig. S12. Metabolomic dataset. Semi-supervised PCA-CA and KODAMA showing unsupervised gender discrimination. PCA-CA-\textit{k}NN classifier for KODAMA was selected by minimizing the $H$ value. MDS was used to visualize the results of KODAMA dissimilarity matrix.
Fig. S13. ETGs dataset. Comparison between KODAMA performed with $k$NN, PCA, ISOMAP, and LLE. Color-coding indicates samples from the same class. The results of the other methods are shown in Fig. S10.
Fig. S14. First component of DM, ISOMAP, PCA, LLE, RF, Sammon, SPE, and $t$-SNE applied to the selected addresses of American presidents, in chronological order.