Diffusion Maps for Dimensionality Reduction and Visualization of Meteorological Data

Ángela Fernández[a,*], Ana M. González[a], Julia Díaz[b], José R. Dorronsoro[a]

[a] *Departamento de Ingeniería Informática, Universidad Autónoma de Madrid, 28049, Spain*
[b] *Instituto de Ingeniería del Conocimiento, 28049, Madrid, Spain.*

Abstract

The growing interest in big data problems implies the need for unsupervised methods for data visualization and dimensionality reduction. Diffusion Maps (DM) is a recent technique that can capture the lower dimensional geometric structure underlying the sample patterns in a way which can be made to be independent of the sampling distribution. Moreover, DM allows to define an embedding whose Euclidean metric relates to the sample's intrinsic one which, in turn, enables a principled application of k-means clustering. In this work we give a self-contained review of DM and discuss two methods to compute the DM embedding coordinates to new out-of-sample data. Then, we will apply them on two meteorological data problems that involve respectively time and spatial compression of numerical weather forecasts and show how DM is capable to, first, greatly reduce the initial dimension while still capturing relevant information in the original data and, also, how the sample-derived DM embedding coordinates can be extended to new patterns.

Keywords: Natural Clustering, Compressed Data, Diffusion Maps, Nyström formula, Laplacian Pyramids

1. Introduction

Methods for dimensionality reduction and data compression, visualization and analysis that preserve the original information of high dimensional patterns are highly valued in data mining and machine learning. The classical example is Principal Component Analysis (PCA) [1] but in recent years methods based on manifold learning such as Multidimensional Scaling [2], Local Linear Embedding [3], Isomap [4], Laplacian Eigenmaps [5] or Hessian Eigenmaps [6] have received a great deal of attention. The common assumption in these methods is that sample data lie in a low dimensional manifold and their goal is to identify the metric on the underlying manifolds from which a suitable low dimensional representation is derived that allows to adequately approximate the original manifold metric with the natural one in the low dimensional representation. Several of these methods rely on the spectral analysis of a data similarity matrix and this is also the case of Diffusion Maps (DM) [7, 8] which we consider here.

As it is the case in other manifold learning methods, DM relies on a graph representation of the sample, where the weight matrix is defined from a suitable similarity matrix of the data points. This approach was pioneered in the several approaches to Spectral Clustering (SC) [9, 10, 11] and their various but essentially equivalent eigenanalysis of the similarity matrix that, in turn, can be connected [5] with the Riemannian geometry of the manifold where the sample is assumed to lie. The main assumption in DM is that the manifold metric can be approximated by the diffusion distance of a Markov process [7] whose transition matrix P is defined by an adequate normalization of the similarity matrix. In turn, this allows the construction of a set of embedding functions, the Diffusion Maps, that transforms the original space into a new one in

*Corresponding author: C/Francisco Tomás y Valiente, 11, E.P.S., Edificio B, 4 planta, UAM-Cantoblanco, 28049 Madrid, Spain; Tel: +34 914972349; Fax: +34 4972235.

Email addresses: a.fernandez@uam.es (Ángela Fernández), ana.marcos@uam.es (Ana M. González), julia.diaz@iic.uam.es (Julia Díaz), jose.dorronsoro@uam.es (José R. Dorronsoro)

which Euclidean distance corresponds to the original manifold metric of the sample data. This means that clustering methods that rely on the Euclidean metric, as is the case of k-means, can be properly applied in the reduced space. In other words, DM can be seen as an intelligent data analysis technique for both dimensionality reduction and clustering (see [12, 13] for other examples of these techniques).

While elegant and powerful, Diffusion Maps (and, in fact, other manifold learning methods, such as Spectral Clustering) have a first drawback on the potentially quite costly eigenanalysis of the transition matrix P that they require. Observe that in principle, sample size coincides with the dimension of the similarity and, hence, transition matrices, which may make its eigenanalysis computationally unfeasible for very large data sets. We will not deal with the issue of the complexity of DM's eigenanalysis in this work but point out that the usual approach is to apply an adequate subsampling of the original data [14]. In turn, this requires a mechanism to extend the embedding computed on that subsample to the other data points not considered or, more generally, new, unseen patterns. This extension is the second important challenge in DM and other manifold learning methods and we will address it in this work through two different algorithms that extend the diffusion coordinates for new out-of-sample points. The first one is an extension to the non-symmetric transition matrix P of the classical Nyström formula [15] for symmetric, positive semidefinite matrices derived from a kernel that extends the eigenvectors of the sample kernel matrix to the eigenfunctions of the underlying integral operator. The second one, the Laplacian Pyramids algorithm [16], also relies on a kernel representation but starts from the discrete sample values $f(x_i)$ (in our case, the eigenvectors of the sample based Markov transition matrix P) of a certain function f (in our case, the general eigenfunctions), and seeks a multiscale representation of f that allows to approximate the values $f(x)$ from an appropriate multiscale combination of the sample values $f(x_i)$.

Even if the ultimate goal is to build a supervised classifier or a predictive model, dimensionality reduction and clustering methods are most often applied in an unsupervised setting, where a first objective is to acquire a knowledge of the underlying data that is simply impossible to achieve under their original, high dimensionality representations. This is a common situation in many of the modern applications of big data analytics that have to deal with large samples whose also large dimensions make this very difficult and even precludes the meaningful use of plain data understanding or visualization tools. The number of fields to which this situation applies keeps constantly growing as the continuous advances in data acquisition and integration make big data settings an ubiquitous issue. This is in particular the case for meteorological data, either by themselves or as inputs to modeling systems, such as those applied, for instance, in renewable energy.

General weather forecasts, or more precisely, Numerical Weather Prediction (NWP) systems, usually involve a large number of variables over relatively fine resolution three-dimensional spatial grids as the meteorological agencies provide forecasts for several pressure layers and for several future horizons. For example, the European Center for Medium-Range Weather Forecasts (ECMWF[1]) offers every three hours about 70 different variables, in 25 pressure layers for 75 future horizons at a 0.125 degree resolution (about 12.5 km). However, the advances in high speed computing imply that spatial and temporal resolutions are constantly increasing and, for instance, 1 km resolutions are already available in some mesoscale models. Even if applied only at horizontal scales, this implies that data dimension will increase by a 150 factor. This relates to forecasts for a given hour but, of course, the selection of the most appropriate data to be used largely depends on the problem at hand, something that determines the dimensionality of the problem to solve and that can result in even larger dimensions.

Thus, the direct analysis of weather forecasts and their application to prediction problems are natural fields where dimensionality reduction and data visualization techniques can be successfully applied. In this work we will consider the application of Diffusion Maps to two examples in this field. In the first one we will illustrate how DM is able to capture in a meaningful way the one year evolution of the forecasts of five weather variables given every three hours for a given point. More precisely, we will show how DM is able to reduce an initial dimension of $14,600$ to a much lower dimension and how Euclidean k-means with $k = 4$ applied to the projected data naturally corresponds to the geopotential height of the points

[1]European Center for Medium-Range Weather Forecasts, `http://www.ecmwf.int/`

considered. This example illustrates how DM can be applied in a time compression setting, as it compress a full year data evolution into single point DM representations. Our second example is concerned with the recent solar radiation forecasting competition set up by the American Meteorological Society [2] and can be considered as a spatial compression problem. More precisely, a set of 15-variable 3-hour forecasts (5 a day) are given for 144 grid points that encompass 98 weather stations in Oklahoma for which accumulated daily radiation average is to be predicted. Data dimension is thus $144 \times 15 \times 5 = 10,800$ and the entire NWP forecast for a day is to be projected in a meaningful low dimensional representation that should be useful for the analysis and eventual forecasts of the aggregated 98 stations daily radiation average. As such, this is ultimately a prediction problem, although we will not put our emphasis on this. However we will use it to illustrate how DM can give meaningful, low dimensional representations that can be understood as a the spatial compression of NWP-based patterns given over a large geographical area. Moreover, in both examples we will also present a comparison of the performances of the previously mentioned two methods to extend DM-embeddings to out-of-sample points that will show that both give similar and good results.

The paper is organized as follows. In Section 2, we shall briefly review Diffusion Maps while in Section 3 the Nyström and Laplacian Pyramids algorithms for extending DM to out-of-sample data are described. In Subsection 4.1 we will illustrate the application of DM to the time compression of one year evolution of NWP weather forecasts for individual grid points and their out-of-sample extension, while in Subsection 4.2 we will apply DM to achieve spatial compression of $10,800$-dimensional patterns that describe the daily NWP forecasts of radiation-related variables for a large area that encompasses the state of Oklahoma; we will also discuss the DM embedding's relationship with the average of the accumulated daily radiation values measured at 98 weather stations. Finally, Section 5 ends this paper with a brief discussion and conclusions.

2. Diffusion Maps

The primary goal of statistical dimensionality reduction techniques such as PCA is to obtain a low dimensional representation of sample data that captures as much of the variance of the original data as it is possible. While these methods lead to a lower dimensional representation of the original patterns, there is essentially no assumption on the underlying geometry of a sample, that they do not try to find out. In contrast, the starting assumption in manifold learning methods is precisely that the sample lies in a low dimensional manifold whose metric they try to identify and exploit. In the case of Diffusion Maps (and also of Spectral Clustering) this is done after the eigenanalysis of a sample-defined similarity matrix.

The construction of a graph is thus the first step in both Spectral Clustering and Diffusion Maps. Given a sample $\mathcal{S} = \{x_1, \ldots, x_n\}$ in the initial feature space, the graph nodes are the sample points x_i and the similarity weights $W_{ij} = w(x_i, x_j)$ are taken to be symmetric and point-wise positive. Different choices are possible for the weight matrix but a frequent one is to define W_{ij} by a Gaussian kernel $w(x_i, x_j) = \exp\left(-\|x_i - x_j\|_2^2/\sigma^2\right)$. The parameter σ implicitly determines the radius of the relevant neighborhood around x_i (see [17] for some directions on how to choose σ). In the case of Spectral Clustering and Diffusion Maps this kernel choice allows a natural connection to the Riemannian structure of the underlying manifold [5] and, moreover, makes possible to use algorithms like the Nyström formula to compute the diffusion coordinates of new, unseen points, as we shall see in Section 3.

It is clear that the distribution of the sample data is an important factor that will affect how well the similarity matrix captures the local geometry of the data. It is thus important to take this distribution into account and following the discussion in [7], a new parameter α, with values between 0 and 1, is introduced and, as we will see, makes explicit the influence of the sample density q. To do so, we will not work directly with W but instead with

$$w_\alpha(x_i, x_j) = \frac{w(x_i, x_j)}{q(x_i)^\alpha q(x_j)^\alpha}$$

where $q(x_i) = \sum_{j=1}^{n} w(x_i, x_j)$ is the degree of each vertex x_i in W.

[2] Solar Energy Prediction Contest (2013-2014), `https://www.kaggle.com/c/ams-2014-solar-energy-prediction-contest`.

Working with this new matrix W_α, the degree of a vertex x_i becomes now

$$g_\alpha(x_i) = \sum_{j=1}^{n} \frac{w(x_i, x_j)}{q(x_i)^\alpha q(x_j)^\alpha} = \sum_{j=1}^{n} w_\alpha(x_i, x_j),$$

and we define a Markov chain over the graph whose transition probability matrix P_α is given by

$$(P_\alpha)_{ij} = p_\alpha(x_i, x_j) = \frac{w_\alpha(x_i, x_j)}{g_\alpha(x_i)}. \tag{1}$$

We denote by π the stationary distribution of this Markov process [18], which is given by $\pi(x) = \frac{g_\alpha(x_i)}{\sum_j g_\alpha(x_j)}$.

We can now consider one-step Markov neighborhoods or, more generally, larger t-step neighborhoods, i.e., those points accessible from a given x_i in t Markov steps. As it is well known, the corresponding probability matrices P^t are given by the powers of P_α, i.e., $p_{\alpha,t}(x_i, x_j) = (P_\alpha^t)_{ij}$. This makes possible to define for each t the diffusion distance

$$D_t^2(x, z) = \|p_{\alpha,t}(x, \cdot) - p_{\alpha,t}(z, \cdot)\|^2_{L^2(\frac{1}{\pi})} = \sum_{y \in \mathcal{S}} \frac{(p_{\alpha,t}(x, y) - p_{\alpha,t}(z, y))^2}{\pi(y)}, \tag{2}$$

where $L^2(\frac{1}{\pi})$ represents the L^2-norm weighted by the inverse of the stationary distribution π. In a sense, $D_t^2(x, z)$ considers x, z to be close if they are connected in the graph by many short paths of length t.

We can rewrite $D_t^2(x, z)$ based on the spectral analysis of the P_α-defined graph [5, 9], which allows for an alternative formulation of the diffusion distance, namely

$$D_t^2(x, z) = \sum_{j=1}^{n-1} \lambda_j^{2t} (\psi_j(x) - \psi_j(z))^2, \tag{3}$$

where λ_j and ψ_j are respectively the eigenvalues and eigenvectors of P_α and we disregard the trivial eigenvalue $\lambda_0 = 1$ and eigenfunction $\psi_0 \equiv 1$. We observe that (3) is simply the Euclidean distance between the points $\hat{\Psi}_t(x)$ and $\hat{\Psi}_t(s)$, where

$$\hat{\Psi}_t(x) = (\lambda_1^t \psi_1(x), \ldots, \lambda_{n-1}^t \psi_{n-1}(x))^\top.$$

This suggests a natural way for dimensionality reduction by simply cutting the number of terms in $\hat{\Psi}_t(x)$ to a number $d(t)$ that, fixing a given precision δ, can be simply chosen as $d(t) = \max\{\ell : \lambda_\ell^t > \delta \lambda_1^t\}$. In other words, we retain those eigenvalues larger than the fraction δ of the power λ_1^t. We approximate (3) now as $D_t^2(x, z) \sim \sum_{j=1}^{d(t)} \lambda_j^{2t} (\psi_j(x) - \psi_j(z))^2$, and, if we define the projection

$$\Psi_t(x) = \begin{pmatrix} \lambda_1^t \psi_1(x) \\ \vdots \\ \lambda_{d(t)}^t \psi_{d(t)}(x) \end{pmatrix}$$

of the original points x_i into the $d(t)$ dimensional space $\mathbb{R}^{d(t)}$, the diffusion distance on the original space can be approximated by the Euclidean distance of the $\Psi_t(x)$ projections in $\mathbb{R}^{d(t)}$:

$$D_t^2(x, z) \sim \sum_{j=1}^{d(t)} \lambda_j^{2t} (\psi_j(x) - \psi_j(z))^2 = \|\Psi_t(x) - \Psi_t(z)\|_2^2.$$

The previous steps are summarized in Algorithm 1.

The diffusion projections $\Psi_t(x)$ lend themselves in a principled way to clustering applications. More precisely, if the parameters that we have chosen are such that the diffusion distance adequately captures the

Algorithm 1 Diffusion Maps Algorithm.

Input: $\mathcal{S} = \{x_1, \ldots, x_n\}$, the original dataset.
Output: $\{\Psi(x_1), \ldots, \Psi(x_n)\}$, the embedded dataset.

1: $(\mathcal{S}, W)$ with $W_{ij} = w(x_i, x_j) = e^{\frac{-\|x_i - x_j\|_2^2}{\sigma^2}}$.
2: $q(x_i) = \sum_{j=1}^{n} w(x_i, x_j)$. % Initial density function.
3: $w_\alpha(x, y) = \frac{w(x,y)}{q(x)^\alpha q(y)^\alpha}$. % Normalized weights.
4: $P_{ij} = p_\alpha(x_i, x_j) = \frac{w_\alpha(x_i, x_j)}{g_\alpha(x_i)}$, with $g_\alpha(x_i) = \sum_{j=1}^{n} w_\alpha(x_i, x_j)$ the graph degree. % Transition Probability.
5: Get eigenvalues $\{\lambda_r\}_{r \geqslant 0}$ and eigenfunctions $\{\psi_r\}_{r \geqslant 0}$ of P such that

$$\left\{ \begin{array}{rcl} 1 & = & \lambda_0 > \lambda_1 \geqslant \cdots \\ P\psi_r & = & \lambda_r \psi_r. \end{array} \right.$$

6: Choose $d(t) = \max\{\ell : \lambda_\ell^t > \delta \lambda_1^t\}$. % Embedding Dimension.
7: Define $\Psi = \begin{pmatrix} \lambda_1^t \psi_1(x) \\ \vdots \\ \lambda_{d(t)}^t \psi_{d(t)}(x) \end{pmatrix}$. % Diffusion Coordinates.

sample's underlying geometry, the $\Psi_t(x)$ embed the original data in a lower dimension space in such a way that the metric of the sample's underlying geometry becomes Euclidean distance on the embedding coordinates. In turn, if we are interested in clustering the sample, it is now quite natural to apply the standard Euclidean k-means algorithm on the embedding coordinates. We will thus obtain k clusters $C_1, \cdots, C_k$ that can be directly related with the clusters $A_1, \cdots, A_k$ in the original sample $\mathcal{S}$ as $A_i = \{x_j | \Psi_t(x_j) \in C_i\}$.

The last remaining question is how to choose the parameter α. It can be seen [7] that the infinitesimal generator L_α of the diffusion process P_α acts on a function f as

$$L_\alpha f = \frac{\Delta(f q^{1-\alpha})}{q^{1-\alpha}} - \frac{\Delta(q^{1-\alpha})}{q^{1-\alpha}} f,$$

where Δ is the Laplace–Beltrami of the underlying manifold. Notice that if $\alpha = 1$, L_1 coincides with Δ and we can expect the diffusion projection to capture the underlying geometry without any interference from the sample's density q. On the other hand, when $\alpha = 0$, we have

$$L_0 f = \frac{\Delta(fq)}{q} - \frac{\Delta(q)}{q} f$$

and the density q will influence how the diffusion coordinates capture the underlying geometry, unless, of course, q is uniform in which case we arrive again at $L_0 = \Delta$. Because of this we will consider the case $\alpha = 1$ in what follows.

In summary, Diffusion Maps provide a simple and elegant way to capture the intrinsic geometry of the sample although with two drawbacks, the possibly costly eigenanalysis that is required to obtain the DM coordinates and the need to avoid repeating this analysis when the projections of new patterns have to be computed. The eigenanalysis of the similarity matrix is unavoidable if DM are to be applied but we describe next two methods to compute the diffusion coordinates of new, unseen patterns.

3. Extending Diffusion Maps to Out-Of-Sample Patterns

The models built in supervised Machine Learning are usually easy to apply on new data points without repeating the whole training algorithms. However, the unsupervised nature of DM and the lack of a explicit function that could be used to compute the projection algorithm make it difficult to obtain the DM embedding of new points. In other words, in DM we find the similarity matrix eigenvector coordinates $\psi_i(x_j)$ of the sample patterns x_j but we do not learn general eigenfunctions $\psi_i(x)$ that give these coordinates for a new x. Thus, it is very important to have relatively simple and efficient ways to extend these $\psi_i(x_j)$ values

to out-of-sample patterns and, hence, to be able to compute the DM embedding of a new pattern x. In this section we will review two different approaches to obtain approximate embedding coordinates for new points.

3.1. Nyström Formula

The Nyström formula [15] is a general method to approximate the eigenfunctions $\psi_j(x)$ of a kernel from the eigenvectors $\psi_j(x_i)$ of a sample-based kernel matrix. As we shall see, in the case of DM, the formula enables to approximately compute the embedding of new patterns without computing again the eigenvalues and eigenvectors of the similarity matrix of the training sample.

Assume $k(x, y)$ is a symmetric, positive semi-definite and bounded kernel; then it has an eigendecomposition [7]

$$k(x, y) = \sum_{l \geq 0} \lambda_l u_l(x) u_l(y).$$

Now, given a sample $\mathcal{S} = \{x_1, \ldots, x_n\}$, let $\{\ell_j\}$, $\{v_j\}$ be the eigenvalues and eigenvectors of the sample-restricted kernel matrix $k_{ij} = k(x_i, x_j)$. Then, the general Nyström method [15] enables us to approximate the u_l eigenfunctions by the following expression that extends the matrix eigenvectors v_j to a new y as

$$v_j(y) = \frac{1}{\ell_j} \sum_{i=1}^{n} v_j(x_i) k(y, x_i). \tag{4}$$

When dealing with the DM setting, notice that, by construction, the Markov transition matrix P in (1) cannot be associated to a symmetric kernel; however, we can apply the approach in Appendix A in [7] and define first a symmetric kernel as

$$a(x, y) = \frac{\sqrt{\pi(x)}}{\sqrt{\pi(y)}} p(x, y),$$

where π is the stationary distribution of the Markov process, as explained in Section 2. It is easy to see that the kernel a is symmetric if we take into account the definition of P in (1):

$$a(x, y) = \frac{\sqrt{\pi(x)}}{\sqrt{\pi(y)}} \frac{w_\alpha(x, x)}{g_\alpha(x)} = \frac{\sqrt{g_\alpha(x)}}{\sqrt{g_\alpha(y)}} \frac{w_\alpha(x, y)}{g_\alpha(x)} = \frac{1}{\sqrt{g_\alpha(y)}\sqrt{g_\alpha(x)}} w_\alpha(y, x) = a(y, x).$$

Moreover, a is also positive semi-definite and bounded [7] and, thus, it has an eigendecomposition

$$a(x, y) = \sum_{l \geq 0} \lambda_l \varphi_l(x) \varphi_l(y),$$

from which the corresponding eigendecomposition of P can be easily derived as

$$p(x, y) = \sum_{l \geq 0} \lambda_l \Psi_l(x) \Phi_l(y),$$

where the eigenvalues λ_l are those of a and Ψ_l and Φ_l are obtained from the eigenfunctions φ_l of the kernel $a(x, y)$ as

$$\Psi_l(x) = \frac{\varphi_l(x)}{\sqrt{\pi(x)}}; \quad \Phi_l(y) = \varphi_l(y)\sqrt{\pi(y)}.$$

Now, we can apply first the Nyström formula (4) to the symmetric kernel a to obtain the approximations

$$\phi_j(x) = \frac{1}{\lambda_j} \sum_{i=1}^{n} \phi_j(x_i) a(x, x_i),$$

to the φ_j eigenvectors from the eigenvalues λ_j and eigenvectors ϕ_j of the sample-based kernel matrix $a_{ij} = a(x_i, x_j)$. If we use the above relationships between Ψ_l and φ_l on the one hand, and a and p on the other hand, we arrive at

$$\begin{aligned}\psi_j(x) = \frac{\phi_j(x)}{\sqrt{\pi(x)}} &= \frac{1}{\sqrt{\pi(x)}}\left(\frac{1}{\lambda_j}\sum_{i=1}^{n}\phi_j(x_i)a(x,x_i)\right)\\ &= \frac{1}{\lambda_j}\sum_{i=1}^{n}\phi_j(x_i)\frac{a(x,x_i)}{\sqrt{\pi(x)}} = \frac{1}{\lambda_j}\sum_{i=1}^{n}\phi_j(x_i)\frac{p(x,x_i)}{\sqrt{\pi(x_i)}}\\ &= \frac{1}{\lambda_j}\sum_{i=1}^{n}\psi_j(x_i)p(x,x_i). \end{aligned} \tag{5}$$

In other words, we can extend Nyström's formula to approximate the values of the DM kernel eigenvectors over new patterns x from those of the matrix eigenvectors $\psi_j(x_i)$ and, thus, to compute the DM coordinates of these x.

The complexity analysis of Nyström's method is easy to make. Observe that to compute (5) for a new pattern x has an $O(N)$ cost for each of the embedding coordinates to be extended. Of course, (5) requires the knowledge of the eigenvalues λ_j and eigenvectors $\psi_j(x_i)$, but these come from the general DM analysis and, thus, do not suppose an extra cost when applying (5). In summary, if we use a training sample of size N and work with D embedding coordinates, computing these coordinates for a new pattern has a cost $O(D\ N)$. We also observe that a nice property of Nyström's method is that it does not need the selection of any algorithm parameter.

3.2. Laplacian Pyramids

The Nyström formula can be seen as a procedure to interpolate the $\psi_j(x_i)$ values to new points x. Laplacian Pyramids (LP), first proposed by Burt and Adelson [19], have been widely used for this purpose, particularly in image coding, low-pass filtering and down-sampling. From a general point of view LP is a multiscale algorithm for extending sample-based function values (in our case, the eigenvectors $\psi_j(x_i)$) that uses different scalings for different resolutions. More precisely, we can apply them [16] to approximate a function f from its values $f(x_k)$ on a sample $\mathcal{S} = \{x_1, \ldots, x_n\}$ using Gaussian kernels of decreasing widths so that we obtain an approximation $\hat{f}(x)$ to this function's extension to new points x, i.e.,

$$f(x) \approx \hat{f}(x) = \hat{f}^{(0)}(x) + \hat{f}^{(1)}(x) + \hat{f}^{(2)}(x) + \cdots + \hat{f}^{(H)}(x).$$

We refer to the construction of these $\hat{f}^{(h)}$ as the training step, where we start with a first approximation $\hat{f}^{(0)}$ that is built starting with a Gaussian kernel G_0 with a wide, initial scale σ_0

$$G_0(x, x') = e^{\frac{-\|x-x'\|_2^2}{\sigma_0^2}},$$

that, in our Markov context, we normalize as

$$\mathcal{G}_0(x, x_p) = \frac{G_0(x, x_p)}{\sum_q G_0(x, x_q)},$$

to ensure that $\sum_p \mathcal{G}_0(x, x_p) = 1$. We then define

$$\hat{f}^{(0)}(x_k) = \sum_{i=1}^{n}\mathcal{G}_0(x_k, x_i)f(x_i), \tag{6}$$

i.e., we use the kernel $\mathcal{G}_0$ as a smoothing operator that, applied to f in the training points, gives us a first approximation $\hat{f}^{(0)}$ for each point x_k in the training set. We then improve this approximation to the

unknown f in an iterative way, working at each step with a finer scale. More precisely, at each step h we define again a new Gaussian Kernel

$$G_h(x, x') = e^{\frac{-\|x-x'\|_2^2}{(\sigma_0/\mu^h)^2}}$$

that uses a sharper scaling σ_0/μ^h and normalize it as before, i.e., $\mathcal{G}_h(x, x_p) = \frac{G_h(x,x_p)}{\sum_q G_h(x,x_q)}$. The concrete value of scaling parameter μ is relatively unimportant and is usually chosen to be 2. Denoting by $d_h(x_i)$ the residual at step h over the pattern x_i, and defined it as

$$d_h(x_i) = f(x_i) - \sum_{l=0}^{h-1} \hat{f}^{(l)}(x_i), \tag{7}$$

we arrive at a new term $\hat{f}^{(h)}(x_k)$, which is given by

$$\hat{f}^{(h)}(x_k) = \sum_{i=1}^{n} \mathcal{G}_h(x_k, x_i) d_h(x_i), \tag{8}$$

and it is added to the approximation $\hat{f}(x)$. This iterative process is applied while the approximation error at each step, computed as $\|d_h\|/N$, is bigger than a prefixed quantity. When these iterations stop we have a multiscale representation of f that we can apply to new, unseen points x as

$$\hat{f}(x) = \sum_{h=0}^{H} \hat{f}^{(h)}(x) = \sum_{i=1}^{n} \mathcal{G}_0(x, x_i) f(x_i) + \sum_{h=1}^{H} \sum_{i=1}^{n} \mathcal{G}_h(x, x_i) d_h(x_i), \tag{9}$$

where $\hat{f}^{(0)}$ and the different $\hat{f}^{(h)}$ are given by (6) and (8) respectively.

It is now straightforward to apply (9) to each one of the eigenvectors ψ_j of the Markov matrix, extending their values to a new x by

$$\hat{\psi}_j(y) = \sum_{h=0}^{H} \hat{\psi}_j^{(h)}(y) = \sum_{i=1}^{n} \mathcal{G}_0(y, x_i) \psi_j(x_i) + \sum_{l=1}^{H} \sum_{i=1}^{n} \mathcal{G}_h(y, x_i) d_{jh}(x_i), \tag{10}$$

with now $d_{jh}(x_i) = \psi_j(x_i) - \sum_{l=0}^{h-1} \hat{\psi}_j^{(l)}(x_i)$. All the steps for the diffusion coordinates extension are outlined in Algorithms 2 and 3.

Algorithm 2 Building the LP Approximation

Input: $\mathcal{S} = \{x_i\}_i$, the sample dataset; ψ_j, the DM eigenvectors; σ_0, the initial width parameter; μ, the σ-reduction factor.
Output: $(\{d_i\}, k)$, the training model formed by the residuals and the number of steps needed.

$\sigma \leftarrow \sigma_0$, $d_0 = \psi_j$, $h = 1$. % Initialization steps.
while $\text{err}_h > \text{err}$ **do**
  $G_h(x_p, x_q) = \exp(-\|x_p - x_q\|_2^2/\sigma^2)$. % Gaussian kernel.
  $\mathcal{G}_h(x_p, x_q) = G_h(x_p, x_q)/\sum_k G_h(x_p, x_k)$. % Normalized kernel.
  $\hat{\psi}_j^{(h)}(x_k) = \sum_p d_{h-1}(x_p)\mathcal{G}_h(x_p, x_k)$. % Approximation to the residual error left at step $h-1$.
  $d_h = d_{h-1} - \hat{\psi}_j^{(h)}$. % New residual computation.
  $\hat{\psi}_j = \hat{\psi}_j + \hat{\psi}_j^{(h)}$. % Approximation to the j-eigenvector at step h.
  $\text{err}_h = {}^{d_h}/_N$.
  $\sigma = {}^{\sigma}/_{\mu}$. % Sharper width parameter.
  $h = h + 1$.
end while
$k = h - 1$.

Finally, concerning the cost of applying LP, we have to take into account the cost of computing the embedding coordinates of a new pattern, but also that of building the underlying LP model that is used to

Algorithm 3 Applying the LP Approximation

Input: $\mathcal{S}$, the sample dataset; y, the new point; $(\{d_i\}, k)$, the ALP model.
Output: $\hat{\psi}(y)$, the extension for the new point.

$\hat{\psi}_0(y) = 0$, $\sigma = \sigma_0$.
for $h = 0$ **to** $k - 1$ **do**
  $G_h(x_p, y) = \exp(-\|x_p - y\|_2^2/\sigma^2)$. % Gaussian kernel.
  $\mathcal{G}_h(x_p, y) = {}^{G_h(x_p,y)}/_{\sum_p G_h(x_p,y)}$. % Normalized kernel.
  $\hat{\psi}_j^{(h)}(y) = \sum_p d_{h-1}(x_p)\mathcal{G}_h(x_p, y)$.
  $\hat{\psi}_j(y) = \hat{\psi}(y) + \hat{\psi}_j^{(h)}(y)$. % Approximation to the j-eigenvector at step h.
  $\sigma = {}^{\sigma}/_{\mu}$.
end for

do so. Assuming a training sample with size N and H resolution levels for the LP model, the cost of applying (10) to a new x is clearly $O(N(H+1))$ per embedding coordinate, i.e., $O(N\ H\ D)$ for a D-dimensional embedding. Contrary to what happened in Nyström's case, LP extensions require to build a model before they can be applied, which means that we have to compute the $(H+1)N$ residuals $d_h(x_i)$ in (7). In turn, each of them requires that the previous $\hat{f}^{(h-1)}(x_k)$ terms in (8) be available, at a $O(N^2)$ cost per iteration. Moreover, we have to decide on the number H of LP iterations using a validation subset S_V to compute the errors of the LP approximations $\hat{f}_H(x_q)$ to the true validation values $f(x_q)$, $x_q \in S_V$, and stopping LP training when these errors start to grow. Doing so adds an extra $O(N\ |S_V|)$ cost per LP iteration that, as $|S_V| \leq N$, is dominated by the previous $O(HN^2)$ for a size N training sample and a LP model with $H+1$ terms, which is thus the main cost in LP. We point out that this is quite less that the roughly $O(N^3)$ cost of the straight DM analysis but, even if we restrict ourselves to consider the Nyström and LP costs only over new patterns, the LP cost is still $O(H)$ times bigger than that of Nyström's. Thus, if only complexity is considered, Nyström's method is more efficient than LP.

4. Diffusion Maps for the Analysis of Meteorological Data

Meteorological data can be seen from two different points of view. In the first one we can concentrate at a given space point and consider weather evolution at that point over a certain large time period. Of course, weather will change along time but we could expect that this evolution is dependent on the point in question, which somehow captures, or compresses, that evolution. In other words, it is conceivable that weather evolution at a concrete point which will make up a very high dimensional feature vector, can be *time compressed* into a lower dimensional representation that somehow characterizes that point.

Alternatively, we can consider the reciprocal of the preceding situation, where for a concrete time moment we collect the weather measurements at a large number of space points over a large area. These measures form also a very large dimensional pattern that we can expect to capture in an abstract way weather behavior at that concrete time. Now it is natural to ask ourselves whether we can *spatially compress* the large weather measurements area in a lower dimensional representation that now is prototypical of that precise time moment.

In the following subsections we consider the application of DM to the time and spatial compression of weather values. We will not work with actual weather measurements but use instead as a proxy NWP values over very large grids.

4.1. Diffusion Maps for time compression

We first illustrate how Diffusion Maps can be applied to compress an entire year evolution of five meteorological variables for points of the ECMWF grid for the Iberian peninsula. More precisely, and as just mentioned, we will not work with actual weather measurements but, instead, with their surface predictions provided by the ECMWF. These predictions are in general close to the actual atmospheric conditions, and they have the advantage of providing values over an uniform large scale grid. To cover the Iberian peninsula

we have selected a $1,995$-point grid with a resolution of 0.25° (i.e., a grid square corresponds to a land square with about a 27 km side).

We will use meteorological forecasts for a whole year, from March 2009 to February 2010, working with five surface variables: wind velocity and x and y direction (given by sine and cosine values), pressure and temperature, that are available every three hours. In this way we have for each day eight snapshots of the meteorological conditions at each one of the $1,995$ nodes in the ECMWF grid for the Iberian peninsula; we take those $1,995$ grid points as our sample $\mathcal{S}$ here. Thus, we assign to each grid node a vector with dimension $5 \times 8 \times 365 = 14,600$ that describes weather evolution at that node over an entire year, and that we seek to compress into another vector of a much lower dimension in a way that still provides meaningful information about each one of the grid nodes. We normalize this data matrix column-wise to have 0-mean and standard deviation 1 variables.

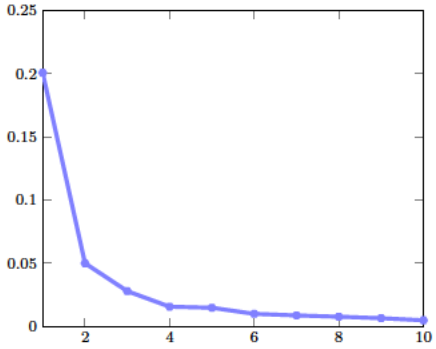
Notice that we are essentially working in an unsupervised setting; nevertheless, we would like to compare the DM results with those obtained from PCA and SC, for which we need a comparison criterion. It is logical to expect that points that are close in the embedded space should also be close under some criterion in the original space. For instance, since we are dealing with surface points, a first notion of closeness could be just geographic proximity; however, this may be just too narrow, as we would also expect that far away points may share similar weather. On the other hand, geographical altitude could also determine similar weather evolution patterns and can bring together distant points. In this experiment, we shall see that this is indeed the case, in the sense that by clustering over the reduced coordinates we are able to describe the Iberian peninsula in terms of the climate in a way that is directly related to the altitude.

We begin with the parameter selection, choosing first the width σ of the Gaussian kernels we will work with. As this parameter defines the neighborhood size, a reasonable idea is fixing it as the median of the Euclidean distances between points x_i and x_j for every pair of points in the grid sample $\mathcal{S}$. The main reason to select this measure is its robustness to outliers. For this concrete problem the σ obtained has a value of 159.18, where the average distance is 166.71 with a standard deviation of 35.34. Recall that those values correspond to normalized data.

The next choices are the diffusion step t and the dimension $d(t)$ of the embedded space. We first apply the threshold-based dimension selection method proposed in [7] and already explained in Section 2. Recall that for a given t, we can select $d(t)$ as $d(t) = \max\{\ell \in N \ : \ \lambda_\ell^t > \delta\lambda_1^t\}$. The precision parameter δ is fixed in our experiments either at 0.1 or at 0.01, which actually means keeping those eigenvalues whose t-th power is 10% or 1% bigger than that of the first relevant eigenvalue (recall that we discard the $\lambda_0 = 1$ eigenvalue). The right hand table in Figure 1 shows $d(t)$ values for both precisions and different values of t. The figure at left shows the eigenvalue decay. Notice that eigenvalues are smaller than 1, and higher t values thus result in smaller λ_ℓ^t values. Seeing the table we discard $d(t)$ values of 1, as probably yielding a too drastic dimension reduction. We also discard the probably too high value of 19 obtained for $\delta = 0.01$ and $t = 1$. This leaves us with the options $t = 1$, $d = 3$ for the 10% precision and $t = 2$, $d = 3$ or $t = 3$, $d = 2$ for the 1% precision. For a more homogeneous comparison we will work with $d = 3$ and t equals 1 and 2. These choices also make sense if we observe the eigenvalue decay at the figure's left, that seems to become linear and close to 0 after the fourth eigenvalue (i.e., $\lambda_d \simeq 0$ for $d \geq 4$).

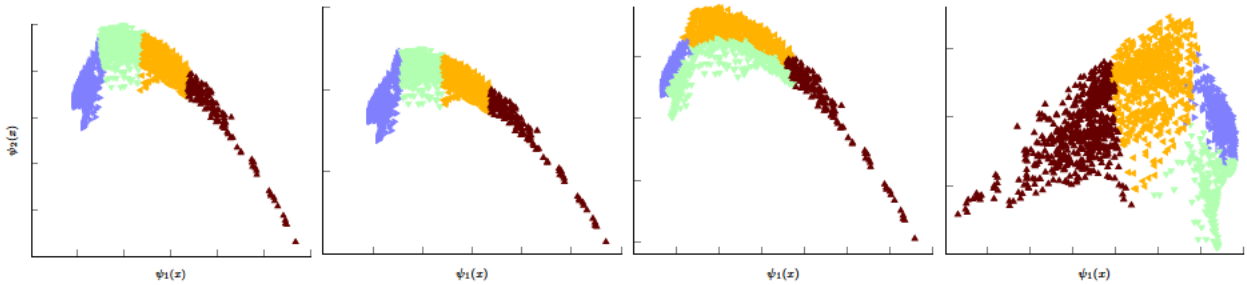
For a homogeneous comparison we will work with a dimension $d = 3$ for Spectral Clustering, that we compute just as DM with $t = 0$. Regarding PCA, although the covariance matrix would be $14,600 \times 14,600$, the data matrix has rank at most $1,995$, i.e., the number of patterns considered. The standard way to decide on the PCA embedding dimension is to retain a certain percentage of variance; however, and again for homogeneity, we will apply PCA also with a 3-dimensional projection, for which the variance explained is 44.18%. In summary, we will compare four different dimensionality reduction models: the classical SC algorithm, PCA, and DM with $t = 1$ and $t = 2$, with the final reduced dimension being 3 in all cases.

As we have mentioned and is the case in any unsupervised problem, the quality of an embedding has to be established by somewhat indirect means. Here we will visualize first the geometric structure of the embeddings and analyze then the clusters obtained after applying Euclidean k-means on the embedding coordinates provided by each method; we choose $k = 4$ in order to achieve an easy visualization. Figure 2 depicts over the first two embedding coordinates the corresponding DM, SC and PCA projections as well as the four cluster structure obtained for each model. In all cases the initial k-means centroids have been



t	$\delta = 0.1$	$\delta = 0.01$
1	3	19
2	1	3
3	1	2
4	1	1

Figure 1: Parameter setting for time compression. The left hand image presents the first 10 eigenvalues (without λ_0) of the probability matrix. The right hand image shows the reduced dimension for some different δ and t values.



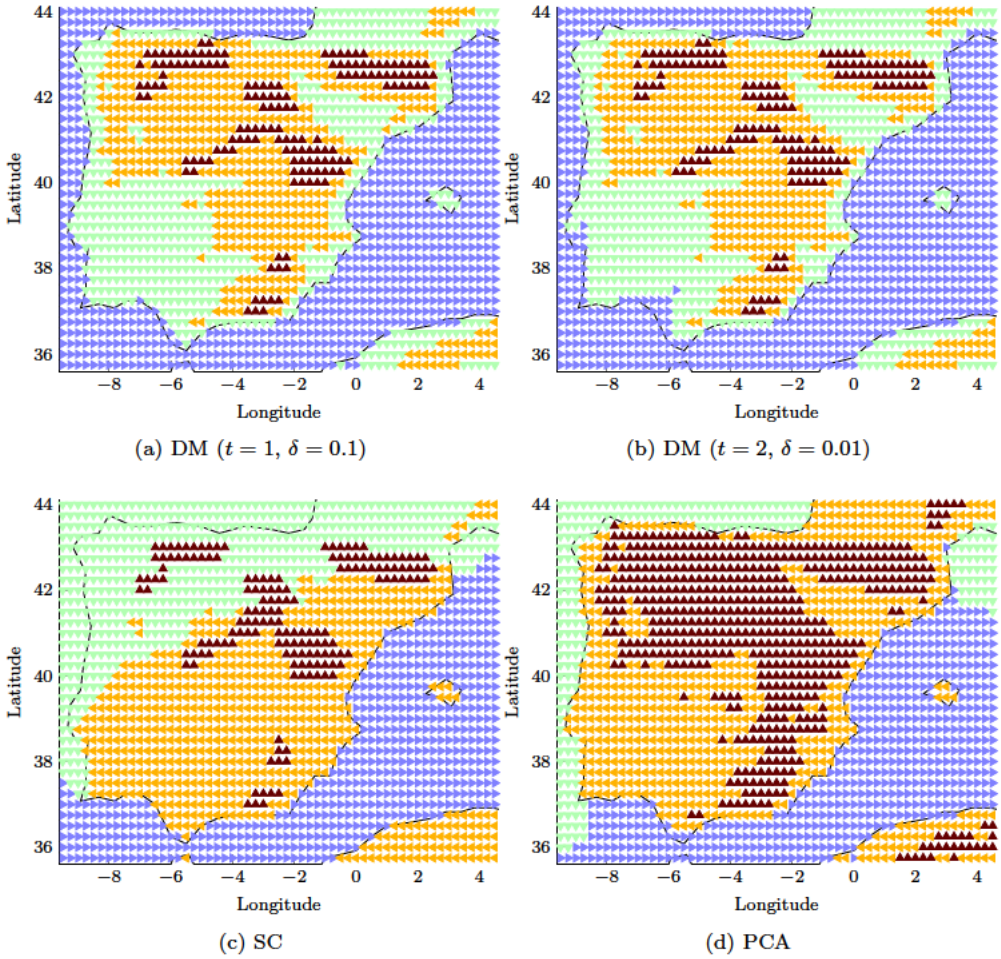
(a) DM ($t = 1$, $\delta = 0.1$) (b) DM ($t = 2$, $\delta = 0.01$) (c) SC (d) PCA

Figure 2: Embeddings resulting from time compression. The images represent the first two embedded coordinates for four different models.

the same for each model; in fact the four model final clusters seem to be quite robust with respect to this initialization, as the results are almost always very similar after different initializations. As it can be seen, the SC and DM coordinates follow a similar pattern, as their embedding coordinates share the eigenvector information and differ only on the λ_i^t eigenvalue dilations. The PCA coordinate structure, while reminiscent of the others, looks nevertheless fuzzier and less crisp. The SC and PCA clusters are also different from the DM ones. These are determined by the first coordinate while the ones for SC and PCA have a less clear structure.

In order to get a more meaningful representation of the different embeddings we have depicted each grid point of the Iberian peninsula according to the color of the cluster it belongs to. The results are shown in Figure 3. All models except SC are able to identify the contours of the Iberian peninsula and North Africa. Seas are depicted by a single cluster in both DM embeddings and the other three DM clusters could be roughly assigned to coasts and lower valleys, the central Spain plateaus and the mountains respectively. The SC and PCA embeddings yield two different sea clusters, that in the case of the SC one also extends to about half of the peninsula. PCA defines two land clusters, one that roughly coincides with the lower altitude DM clusters while the other seems to encompass the other two DM clusters. For SC, one of the land clusters can be associated to the mountain regions and the other two separate the peninsula diagonally into two halves with one of them extending, as already mentioned to the sea.

The preceding discussion suggests that the DM clusters may reflect a height based representation of the NWP grid. Each grid point has associated its geopotential in the underlying NWP orography model and since we are working with surface forecasts, we have represented in Figure 4 box plot diagrams of the geopotential distribution of each cluster after normalizing geopotentials to zero mean and 1 standard deviation. The box plots show the sample minimum and maximum, the median and the lower and upper



(a) DM ($t = 1$, $\delta = 0.1$)

(b) DM ($t = 2$, $\delta = 0.01$)

(c) SC

(d) PCA

Figure 3: Maps resulting for time compression.

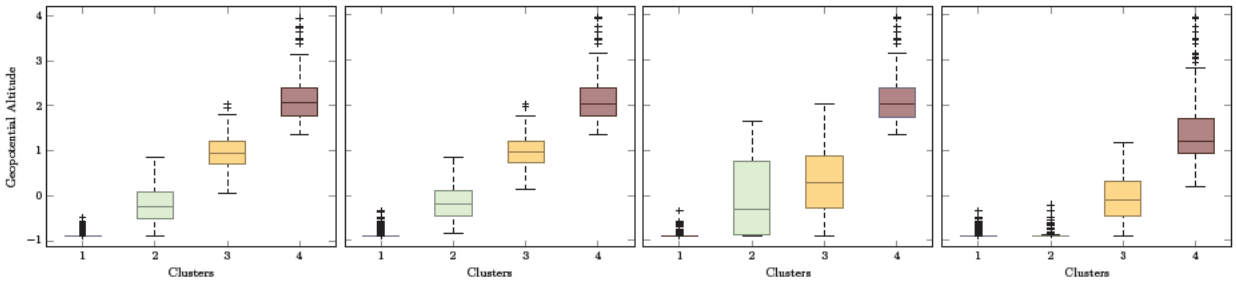
quartiles. They also indicate which observations, if any, might be considered outliers.

Clearly, the DM clusters are separated according to the geopotential altitude and a Mann–Whitney–Wilcoxon test shows each cluster distribution to be statistically different. The box plots of the SC and PCA clusters have a much less defined structure. There is a big SC cluster that mixes different geopotentials that, as we have seen above, corresponds to the North half of the peninsula and the adjacent sea. Also, the box plots of the two PCA sea clusters essentially overlap while the other two box plots, although different, have a less marked structure than their DM counterparts. In summary, we can conclude that DM achieve a meaningful time compression, being able to reduce the original $14,600$ dimension to a much lower of 3 while capturing relevant information on the initial patterns in a better way than achieved by either SC or PCA.

4.1.1. Out-of-sample extensions for the time compression problem

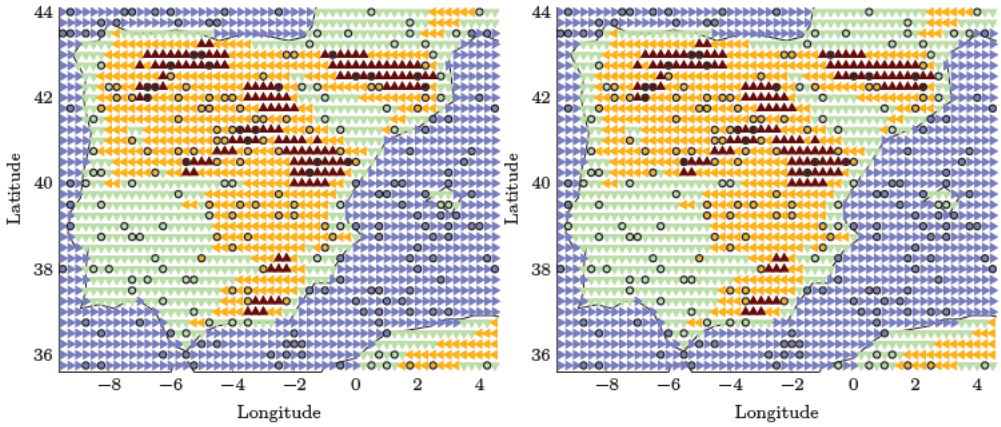
We turn next to the issue of computing the embedding coordinates of new points under the time compression scheme of the previous section. As explained in Section 3, one of the biggest challenges when working with Diffusion Methods is to apply them to new patterns without having to recompute the eigenvalue and eigenvector structure of the Markov matrix over which DM rely. We study here the application of Nyström's formula and Laplacian Pyramids with this goal.

For this purpose we will randomly take out of the sample one year patterns associated to 100, 250 and 500 grid nodes respectively (i.e., we take, approximately, between 5% and 25% of the original sample) that,



(a) DM ($t = 1$, $\delta = 0.1$) (b) DM ($t = 2$, $\delta = 0.01$) (c) SC (d) PCA

Figure 4: Box plots of the geopotential altitude for time compression.



(a) Nyström (b) LP

Figure 5: Extension for new, unseen points.

in turn, will form the test datasets. We have then computed the DM eigenstructure over the new training set with parameter $t = 1$ and, finally, we have obtained the embedding coordinates of the test patterns using both out-of-sample methods. Recall that the LP training algorithm has a stopping criterion based on an error threshold that has to be prefixed. For these experiments we have computed a 5-fold cross-validation to find its optimal value.

We will get a first visual measure of the quality of the out-of-sample extension applying k-means with $k = 4$ to the embedded coordinates of the training points, and associating then the embeddings of the test patterns to the closest training cluster. If the out-of-sample embeddings are correct, the test patterns should be assigned to the same clusters of their neighbors. We can visually appreciate whether this is the case checking that the cluster color of the test points coincide with the cluster color of their training neighbors. The results for both methods in the 250 test point case are shown in Figure 5 where we have marked test points as circles. As we can see, this coincidence happens most of the time and when this is not the case, the errors can be considered as relatively minor as the cluster assignment is still coherent with a node's neighbors; for example, a sea point is never taken as a mountain one.

A better, more precise way of measuring the quality of the Nyström and LP projections can be to compare the cluster assignments of the test points after applying k-means, again with $k = 4$, on the training subgrid (the "predicted" assignments) with those made when projections are computed after applying DM to the entire grid sample without removing any nodes (the "real" assignments). In other words, we transform the out-of-sample computations into a classification problem, where a test point is correctly classified if the cluster to which it is assigned according to its Nyström or LP embedded coordinates coincides with

the one it belongs when k-means is applied over the DM coordinates computed over the entire grid. The classification accuracy, that is, the percentage of test nodes assigned to their "real" clusters, will now be the quality measure. Table 1 includes the confusion matrices of both methods for the different test sets. These matrices confirm the overall good performance, and also that the few missclassifications are always between near clusters. Notice that the clusters will be numbered accordingly to their mean geopotential altitude, thus, cluster 1 will represent the sea and cluster 4, the mountains. It can be also seen on this table how both methods offer a very similar high accuracy for each dataset, being LP slightly better than the Nyström method in the case of the experiments with 100 and 250 test points. For the most complicated example, the subset with 500 test points—around the 25% of the original sample—the Nyström method is slightly better than the other ones. with Nyström's method having a slight advantage over LP as the number of test patterns grow. We point out that these comparisons have been repeated in 100 experiments where we randomly select the test subsets. The results in Table 1 correspond to the averages computed over the 100 test sets.

	P_1	P_2	P_3	P_4	$\sum$
R_1	39.93	0.49	0.00	0.00	40.42
R_2	0.56	25.07	0.97	0.00	26.60
R_3	0.00	1.07	24.25	0.50	25.82
R_4	0.00	0.00	0.54	6.62	7.16
$\sum$	40.49	27.17	25.76	7.12	100.00

(a) Nyström extension for 100 test points.
Accuracy: 95.87%

	P_1	P_2	P_3	P_4	$\sum$
R_1	39.98	0.44	0.00	0.00	40.42
R_2	0.15	26.17	0.28	0.00	26.60
R_3	0.00	0.03	25.79	0.00	25.82
R_4	0.00	0.00	0.56	6.60	7.16
$\sum$	40.13	27.20	26.63	6.60	100.00

(b) LP extension for 100 test points.
Accuracy: 98.54%

	P_1	P_2	P_3	P_4	$\sum$
R_1	98.55	0.78	0.00	0.00	99.33
R_2	1.48	65.94	1.79	0.00	69.21
R_3	0.00	1.65	60.83	0.83	63.31
R_4	0.00	0.00	0.90	17.25	18.15
$\sum$	100.03	69.27	63.52	18.08	250.00

(c) Nyström extension for 250 test points.
Accuracy: 97.03%

	P_1	P_2	P_3	P_4	$\sum$
R_1	99.16	0.77	0.00	0.00	99.93
R_2	0.74	67.97	0.46	0.00	69.17
R_3	0.00	0.07	63.00	0.01	63.08
R_4	0.00	0.00	1.33	16.49	17.82
$\sum$	99.90	70.14	64.79	16.50	250.00

(d) LP extension for 250 test points.
Accuracy: 98.65%

	P_1	P_2	P_3	P_4	$\sum$
R_1	198.34	1.25	0.00	0.00	199.59
R_2	1.89	132.56	2.59	0.00	137.04
R_3	0.00	2.36	122.53	1.53	126.42
R_4	0.00	0.00	2.68	34.27	36.95
$\sum$	200.23	138.85	127.80	35.80	500.00

(e) Nyström extension for 500 test points.
Accuracy: 97.54%

	P_1	P_2	P_3	P_4	$\sum$
R_1	193.07	6.52	0.00	0.00	199.59
R_2	1.49	128.46	7.09	0.00	137.04
R_3	0.00	0.69	122.40	3.33	126.42
R_4	0.00	0.00	4.02	32.93	36.95
$\sum$	194.56	139.69	133.51	36.26	500.00

(f) LP extension for 500 test points.
Accuracy: 95.37%

Table 1: Confusion matrices for Nyström and LP methods for the three test datasets for the average over 100 experiments.

The preceding discussion shows that we can obtain good clustering accuracies when Nyström or LP are used to extend diffusion coordinates. This is a somewhat indirect measure but we show next that both methods also give good intrinsic approximated embeddings for new points. A simple way of doing so is to compute what we may call the Frobenius distance, that is, the Frobenius norm of the data matrix with the differences between the exact, i.e., the "true" embedding coordinates of the test points and their

test coordinates computed using Nyström or LP. We recall that the Frobenius norm of a matrix is just the Euclidean norm of the vectorized matrix entries, i.e. $\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$. We used a relative Frobenius distance defined as $\text{dis}_F = \|\text{DM}_{\text{full}} - \text{DM}_{\text{ext}}\|_F / \|\text{DM}_{\text{full}}\|_F$, where DM_{full} represents the DM coordinates of the test points obtained applying DM over the training and test sets together, and DM_{ext} represents the coordinates where DM has been computed over the training set and those of the test points has been extended via Nyström or LP.

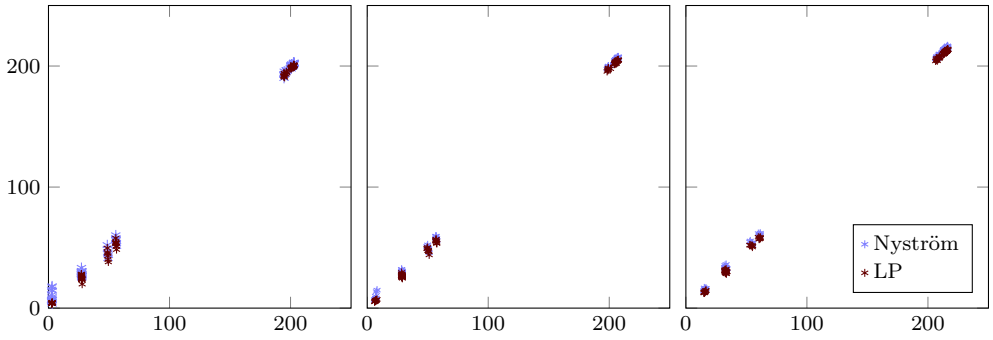
Both sets can be measured over the training points, over the test points or over both together. Of course, this Frobenius distance will depend on the number of test points considered. We can compare three different matrices: the full embedding matrix which contains the corresponding coordinates of the entire sample, the embedding matrix corresponding only to the training subsample and the embedding matrix corresponding to the test subsample. Recall that embeddings are computed after a suitable eigenanalysis of the training subsample, which is quite influenced by the selection made of the test subset, particularly when, as it will be the case in some examples, it is a sizable part of the entire sample. Notice that if the embeddings of the training subsample are quite different when computed over the full matrix or over the training submatrix, they will also differ markedly over the test subsample. This fact has to be taken into account when comparing the approximate embeddings computed for test patterns with their full sample counterparts.

These three different distance values are presented in Table 2, that shows the medians of the corresponding relative distances given as percentages when they are computed over the 100 different train-test partitions. Because of the preceding discussion, we use the median as it is more robust than the mean in a case where we obtain DM_{ext} embeddings very different from DM_{full}.

		100	250	500
Training		56.17%	51.01%	207.53%
Test	Nyström	59.08%	51.64%	207.47%
	LP	56.22%	49.75%	205.87%

Table 2: Median relative Frobenius distances between exact and approximated embedding coordinates computed for 100 experiments.

These Frobenius distances can be considered as the relative reconstruction distances, and we can observe how they grow, as expected, with the number of test points. And notice how, when we take 500 points out of the sample set for testing, the median of the relative Frobenius distance may be above 100%, which means that the training and test embeddings obtained are totally different from the ones computed with the full DM. This is possibly due to the large size of the test subset relative to the training one (recall that 500 points are about 25% of the sample set). However, we point out that the median result can be further refined in terms of the actual distribution of reconstruction distances that is to a great extent determined by the distance between the DM embeddings of the training subsample when they are computed over the entire sample or over the training subset only. As mentioned above, we cannot expect the "real" and extended (i.e., after applying Nyström and LP) embeddings of the test subset to be close when this is not the case for the original embeddings computed directly by a direct eigenanalysis over the training patterns. In fact, a low train distance almost always correspond to a low test distance and the table exemplifies this, as test and train distances are rather similar in all cases. This effect can be checked over Figure 6, where we present for the different test set sizes the relative Frobenius distances of the direct training subsample embeddings versus the relative distances over the extended testing subsample embeddings for the 100 experiments. Observe that the points essentially appear in the diagonal of the graph for both extended embeddings. Moreover, there are two clearly different regions, a left bottom one of the smaller differences and a top right, far away region, where both train and test embeddings are simultaneously large. The figure thus supports our previous hypothesis that the Nyström and LP methods behave well when the "real" training embedding differences are smaller.



(a) 100-points test set. (b) 250-points test set. (c) 500-points test set.

Figure 6: Relative Frobenius distances over the training subsample embedding versus the relative Frobenius distances over the testing subsample embedding for the 100 experiments done for each different test set size.

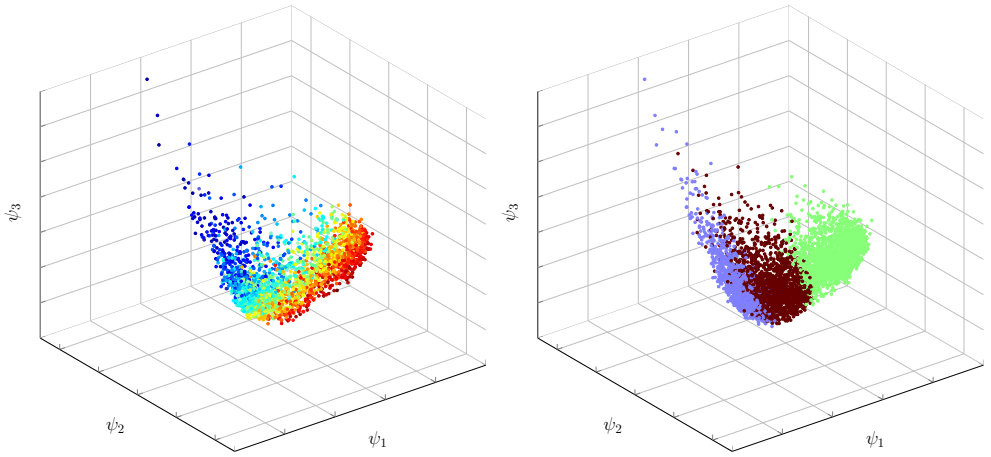
4.2. Diffusion Maps for spatial compression

We turn now our attention to the ability of DM to spatially compress weather forecasts. We will consider a solar radiation problem where we have data from the 2013–2014 Solar Energy Prediction Contest organized by the American Meteorological Society (AMS) and hosted by the company Kaggle. The ultimate goal in this competition was to predict the total daily incoming solar radiation in 98 meteorological stations located in Oklahoma using NWP forecasts of 15 variables for each one of the nodes of a grid that encompasses the state. Here, instead, we will focus on using DM (we will not deal now with SC or PCA) to compress for any given day the spatial feature vector made up of the weather predictions for the entire grid and to obtain thus a meaningful low dimensional embedding of the daily NWP patterns that could help us to understand the relationship between them and the average radiation computed over all stations. We will also consider the extension of the coordinates that result from the spatial compression to new patterns, using the Nyström and LP methods described previously.

The complete input data of the contest are, on the one hand, the numerical weather predictions (NWP) given as eleven ensembles from the NOAA/ESRL Global Ensemble Forecast System (GEFS) and, on the other hand, daily aggregated radiation readings from the 98 stations. For this experiment setting we have used only the NWP forecasts in the main ensemble. For every day, it contains predictions for five time steps (12 to 24 UTC-hours in three hour increments), giving for each one 15 variables related to radiation, temperature, precipitation, cloud clover and pressure. These NWP forecasts are given over a grid with $16 \times 9 = 144$ points. Assuming a pattern for each day, pattern dimension is thus $144 \times 15 \times 5 = 10,800$, with spatial spread (144 points) dominating over the time (5) or variable (15) dimensions. These data are available for the years between 1994 and 2007, yielding a total of $5,113$ days and, hence, patterns and our first goal is to compress for each day these daily $10,800$-dimensional spatial features.

We have applied first the DM algorithm to obtain a low dimensional embedding. In this case, the parameter selection has been done as in Section 4.1, selecting a σ value of 132.61 as the median of the Euclidean distances between points (the mean distance is now 138.03 and 48.73 the standard deviation) and we have fixed $\alpha = 1$ and $t = 1$. To decide on the embedding dimension, we have used a precision of $\delta = 0.1$ (i.e., we discard dimensions whose eigenvalues are smaller than 10% of the first one), reducing the initial $10,800$ dimension to just 3.

The resulting embedding over the whole dataset is shown in the left image of Figure 7, where its coordinates have been colored by the average over the 98 stations of their daily-aggregated incoming solar radiation. Observe that this measured radiation is not one of the NWP variables and, thus, is not included in the embedding. Even so, we can appreciate that the three-dimensional DM embedding captures quite clearly the average radiation in a band-like structure, with high radiation days (red and brown points) being clearly separated from low ones (blue and dark blue points). This band-like structure is approximately parallel to the first embedding axis, with a higher density of high radiation points (red and brown dots) to its



(a) DM Embedding colored by radiation. (b) k-means clusters over the DM Embedding.

Figure 7: Diffusion Coordinates, colored by the real incoming solar radiation average in the left hand image, and by the resulting k-means clustering over the embedding, with $k = 3$, in the right hand image.

right and a higher density of low radiation points (blue) to its left. Besides the fact that several of the NWP variables have only an indirect effect on radiation, we point out that radiation is perhaps one variable on which NWP models have a largest degree of uncertainty. Thus we cannot hope that the embedding captures radiation with a high precision, given that this is not the case in the starting NWP radiation forecasts.

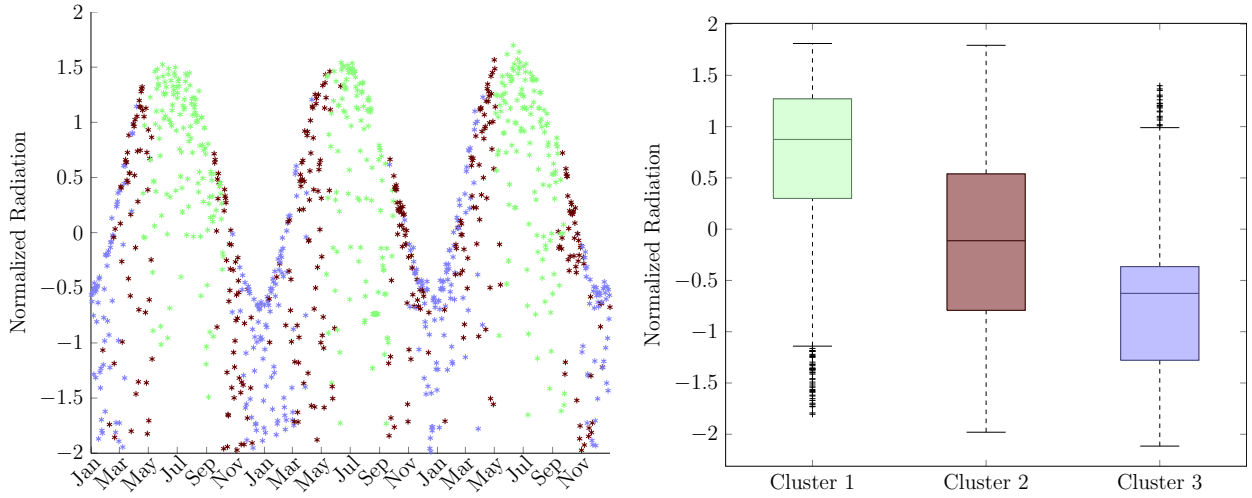
To try to understand the DM embedding, we have built clusters over the embedding coordinates and checked their relationship with the measured radiation patterns. With this objective in mind, we have applied k-means over the diffusion coordinates, with $k = 3$, hoping to detect perhaps in the embedded data days with high radiation, days with intermediate radiation and days with low radiation. The resulting clusters are depicted in the right hand side of Figure 7 and show how clusters are defined mostly along the first DM dimension. Comparing both images in Figure 7 and taking into account the previous discussion, we could say that the DM clusters capture the high density areas of points with low, medium and high average radiation values.

To better visualize the possible meaning of these clusters, we have depicted the radiation time series colored by the cluster assigned to each day. This is shown in the left hand side of Figure 8 for the first three years in the sample, and we can see a structure of relatively wide green and blue vertical bands, corresponding approximately to summer and winter months respectively, and thinner intermediate brown vertical bands that approximately dominate spring and fall.

Moreover, in the right hand side of the figure we have drawn box plots for the distribution of average radiation in each cluster. We can see that the medians of the three clusters are well separated and the extreme radiation boxes overlap only at their respective outliers; the intermediate cluster has a higher overlapping but this is not incompatible with its assignment of time periods between summer and winter. Looking at these images we can conclude that DM achieves a high degree of spatial compression of the NWP data, with the diffusion coordinates capturing the regions with a higher density of low, medium and high radiation days and also representing the seasonality that is intrinsically present in radiation measurements.

4.2.1. Out-of-sample extensions for the spatial compression problem

Now we are going to study the performance of Nyström and LP methods for extending diffusion coordinates to new, unseen patterns in this example. We are going to use the previously described NWP forecasts for the years 1994–2004 as the training set (4,018 patterns), and the years 2005, 2006 and 2007 for testing purposes (1,095 patterns). We take advantage of the natural ordering that is possible in time dependent patterns to work with a single train-test split.



(a) Three years' radiation colored by cluster.

(b) Radiation Boxplots.

Figure 8: Other visualization methods for the embedding clusters effect.

For this purpose, we first compute an embedding over the training set, and we apply the out-of-sample methods to extend the coordinates on the test points. We will also make use of the DM embedding previously computed over the entire sample, for comparing purposes as we have done with the time compression example. For evaluating the results obtained we use the same tools presented in Subsection 4.1.1: confusion matrices over the previously defined three clusters and relative Frobenius distances between the "true" and extended diffusion coordinates. The confusion matrices and the accuracy obtained for the classification problem between "real" and predicted cluster are shown in Table 3. As in the previous example, we can also see now high accuracies for both methods.

	P_1	P_2	P_3	$\sum$
R_1	303	0	0	303
R_2	23	325	0	348
R_3	0	11	433	444
$\sum$	326	336	433	1095

(a) Nyström extension.
Accuracy: 96.89%

	P_1	P_2	P_3	$\sum$
R_1	303	0	0	303
R_2	10	338	0	348
R_3	0	3	441	444
$\sum$	313	341	441	1095

(b) LP extension.
Accuracy: 98.81%

Table 3: Confusion matrices for Nyström and LP methods over the test dataset.

To evaluate how similar is the reconstructed matrix to the one obtained computing directly DM over the whole set we define the relative Frobenius distance between both matrices. The results are presented in Table 4. Recall that, as before, this distance is presented for three different matrices. We observe that the distances for this example are lower than for the time compression one. Here, we can said that the reconstruction is good: both methods perform well and are almost equal.

Note that the results obtained over this example are more robust. One reason is probably that we have more data and the training and test sets constructed are better representatives of the problem to solve (we counted with seven complete years for training and three for test). Another, and perhaps a more important issue is that here we can expect a greater similarity between the training and test subset patterns, as radiation (and to some extent, weather itself) is clearly a seasonally periodic phenomenon where patterns in the testing years cannot be extremely different from patterns in the previous training years. This similarity

		$\|\|\|_F$
	Train	13.45%
Test	Nyström	13.58%
	LP	14.33%

Table 4: Frobenius distances between exact and approximated embedding coordinates.

makes the embeddings of the extension somehow easier. In any case, we can also conclude here that Nyström and LP are both good methods for extending diffusion coordinates when we have a sample big enough to represent the underlying information, and also a good out-of-sample set, correlated enough with the patterns we have previously seen.

5. Conclusions

Diffusion Maps is a recent manifold learning method for dimensionality reduction that assumes the original sample patterns being embedded in a low dimensional manifold whose metric can be characterized by the diffusion distance of a Markov chain model defined from a sample's kernel similarity matrix. The spectral analysis of the transition matrix of the Markov chain suggests naturally that the diffusion metric can be computed as the Euclidean distance of a properly defined embedding, the Diffusion Maps proper. In turn, this Euclidean representation lends itself to a simple procedure to choose a much lower embedding dimension that, nevertheless, still allows a good estimation of the Euclidean (and the manifold's) metric. Also, the Euclidean metric of the embedded space allows to apply on it in a principled way clustering methods such as k-means with an implicit Euclidean distance assumption on their sample.

In this paper we have applied Diffusion Maps for the compression and analysis of weather forecasts from both a spatial and a temporal data compression points of view. In our time compression example, that of the forecast evolution of an entire year over the points of a NWP grid, we have shown that 4-means clustering over three-dimensional DM coordinates gives clusters that relate weather patterns to their concrete spatial location in a more meaningful way than that achieved by other models such as PCA or standard Spectral Clustering. For the spatial compression example, that of daily NWP forecasts on a grid that encompasses the state of Oklahoma, we have shown that the DM embedding can also be helpful to visually analyze the available data and that the spatial compression is indeed achieved, obtaining a meaningful low dimensional representation that captures the intrinsic seasonality present in radiation measures.

Either by itself or by the increasing importance of renewable energy, the analysis of meteorology data, particularly, that of NWP forecasts, is a field of interest for which the very high data dimensionality makes embedding methods such as DM to be important tools. In this line different issues of interest arise for further applications of DM. An important one is the detection and analysis of singular days where actual meteorological behavior markedly differs from what it was to be expected from NWP forecasts. Related to this but from the opposite point of view is the goal of identifying past days that are similar to the current one and to exploit the actual meteorological behavior on them to refine the NWP (or renewable energy) forecasts for the current one. We are presently pursuing these and other related applications of DM.

Acknowledgements

The authors acknowledge partial support from Spain's grant TIN2010-21575-C02-01 and the UAM–ADIC Chair for Machine Learning. The first author is also supported by an FPI–UAM grant and kindly thanks the Applied Mathematics Department of Yale University for receiving her during her visits.

References

[1] C. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[2] T. Cox, M. Cox, Multidimensional Scaling, Chapman and Hall/CRC, 2000.
[3] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (2000) 2323–2326.
[4] J. Tenenbaum, V. de Silva, J. Langford, A global geometric framework for nonlinear dimensional ity reduction, Science 290 (5500) (2000) 2319–2323.
[5] M. Belkin, P. Nyogi, Laplacian Eigenmaps for dimensionality reduction and data representation, Neural Computation 15 (6) (2003) 1373–1396.
[6] D. Donoho, C. Grimes, Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data, Proceedings of the National Academy of Sciences of the United States of America 100 (10) (2003) 5591–5596.
[7] R. Coifman, S. Lafon, Diffusion Maps, Applied and Computational Harmonic Analysis 21 (1) (2006) 5–30.
[8] R. Coifman, S. Lafon, A. Lee, N. Maggioni, B. Nadler, F. Warner, S. Zucker, Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion Maps, Proceedings of the National Academy of Sciences 102 (21) (2005) 7426–7431.
[9] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (6) (2000) 888–905.
[10] U. Luxburg, A tutorial on spectral clustering, Statistics and Computing 17 (4) (2007) 395–416.
[11] A. Ng, M. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, in: Advances in Neural Information Processing Systems, MIT Press, 2001, pp. 849–856.
[12] A. Abraham, E. Corchado, J. Corchado, Hybrid learning machines, Neurocomputing 72 (2009) 2729–2730.
[13] M. Negnevitsky, Artificial Intelligence: A Guide to Intelligent Systems, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.
[14] M. Belabbas, P. Wolfe, Spectral methods in machine learning and new strategies for very large datasets, Proceedings of the National Academy of Sciences 106 (2) (2009) 369–374.
[15] Y. Bengio, O. Delalleau, N. L. Roux, J. Paiement, P. Vincent, M. Ouimet, Learning eigenfunctions links Spectral Embedding and Kernel PCA, Neural Computation 16 (10) (2004) 2197–2219.
[16] N. Rabin, R. R. Coifman, Heterogeneous datasets representation and learning using Diffusion Maps and Laplacian Pyramids, in: SDM, SIAM / Omnipress, 2012, pp. 189–199.
[17] N. Rabin, Data mining in dynamically evolving systems via diffusion methodologies, Ph.D. thesis, Tel-Aviv University (2010).
[18] L. Rogers, D. Williams, Diffusions, Markov Processes, and Martingales. Vol. 1, Cambridge University Press, Cambridge, 2000.
[19] P. J. Burt, E. H. Adelson, The laplacian pyramid as a compact image code, IEEE Transactions on Communications 31 (1983) 532–540.

Ángela Fernández is a doctoral researcher in the Computer Science Department of the Universidad Autónoma de Madrid, Spain, where she also collaborates with the Instituto de Ingeniería del Conocimiento. She received from this university her degrees in Computer Engineering and in Mathematics in 2009, and her MSc degree in Computer Science in 2010. Her main research is focused on manifold learning and clustering, but also covers additional machine learning and pattern recognition paradigms.

Ana M. González (PhD, Universidad Autónoma de Madrid, Spain) is Associated Professor of Computer Engineering at the Universidad Autónoma de Madrid.

Julia Díaz (PhD, Universidad Autónoma de Madrid; Spain) is Health and Energy Predictive Analytics Innovation Director at the Instituto de Ingeniería del Conocimiento (IIC) and Part Time Professor of Computer Engineering at the Universidad Autónoma de Madrid. She has authored more than 20 scientific papers in machine learning and applications, has managed a large number of research and innovation projects and has 10 years' experience on renewable energy prediction.

José R. Dorronsoro (PhD, Washington University in St Louis; USA) is Professor of Computer Engineering at the Universidad Autónoma de Madrid. He has authored more than 100 scientific papers in mathematical analysis, machine learning and applications, has managed a large number of research and innovation projects and has 10 years experience on wind and solar energy prediction. Dr Dorronsoro is also a senior scientist at the Instituto de Ingeniería del Conocimiento (IIC), where he leads IICs research and innovation on wind and solar energy prediction, that services more than 100 wind and solar farms in Spain and about 4GW.