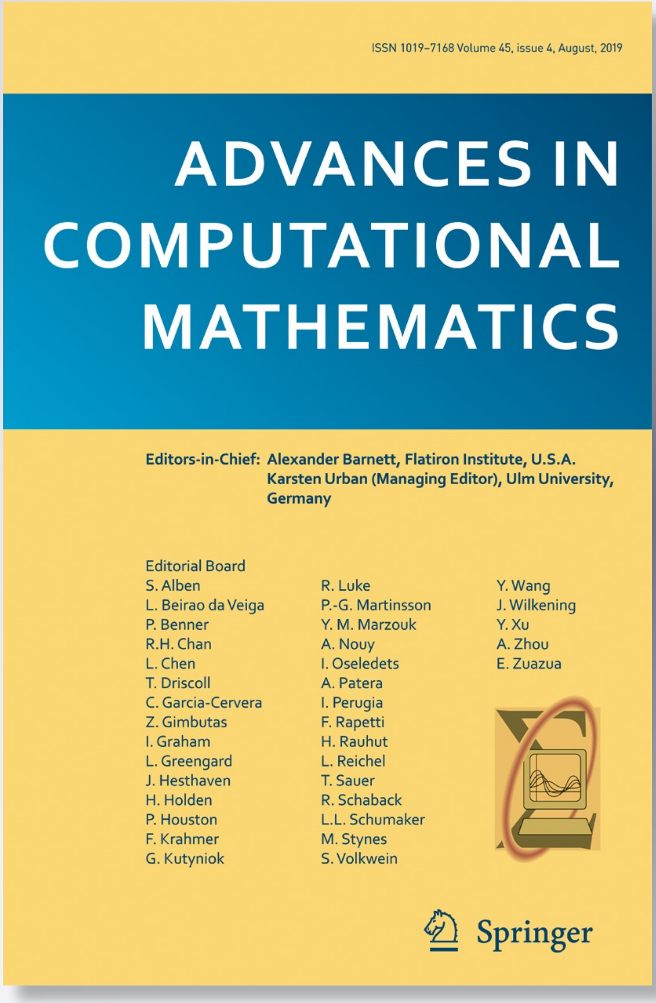
Two directional Laplacian pyramids with application to data imputation

Neta Rabin & Dalia Fishelov

Advances in Computational Mathematics

ISSN 1019-7168
Volume 45
Number 4

Adv Comput Math (2019) 45:2123-2146
DOI 10.1007/s10444-019-09697-7



ISSN 1019-7168 Volume 45, issue 4, August, 2019

ADVANCES IN COMPUTATIONAL MATHEMATICS

Springer

Advances in Computational Mathematics (2019) 45:2123–2146
https://doi.org/10.1007/s10444-019-09697-7

Two directional Laplacian pyramids with application to data imputation

Neta Rabin[1] · Dalia Fishelov[1]

Received: 26 September 2018 / Accepted: 3 April 2019 /
Published online: 22 April 2019

Abstract

Modeling and analyzing high-dimensional data has become a common task in various fields and applications. Often, it is of interest to learn a function that is defined on the data and then to extend its values to newly arrived data points. The Laplacian pyramids approach invokes kernels of decreasing widths to learns a given dataset and a function defined over it in a multi-scale manner. Extension of the function to new values may then be easily performed. In this work, we extend the Laplacian pyramids technique to model the data by considering two-directional connections. In practice, kernels of decreasing widths are constructed on the row-space and on the column space of the given dataset and in each step of the algorithm the data is approximated by considering the connections in both directions. Moreover, the method does not require solving a minimization problem as other common imputation techniques do, thus avoids the risk of a non-converging process. The method presented in this paper is general and may be adapted to imputation tasks. The numerical results demonstrate the ability of the algorithm to deal with a large number of missing data values. In addition, in most cases, the proposed method generates lower errors compared to existing imputation methods applied to benchmark dataset.

Keywords Laplacian pyramids · RNA sequencing data · Two-sided LP scheme · Imputation

Mathematics Subject Classification (2010) 68T30

Communicated by: Pavel Solin

✉ Neta Rabin
netar@afeka.ac.il; neta.rabin@gmail.com

Dalia Fishelov
daliaf@afeka.ac.il; fishelov@gmail.com

[1] Afeka - Tel Aviv Academic College of Engineering, 38 Bnei Efraim St., Tel Aviv, Israel

1 Introduction

Modeling and analyzing high-dimensional data has become a common task in various fields and applications. Kernel-based machine learning methods are capable of generating compact models that capture underling important features of the complex dataset. Typically, given a dataset X of size $M \times N$, a kernel is constructed based on the rows of X. This kernel captures the pairwise distances between the rows of X. In classification algorithms, such as SVM (Support Vector Machines), kernels are used for finding non-linear separations between data classes. In addition, non-linear dimensionality reduction algorithms, such as diffusion maps [6], utilize spectral decomposition of normalized kernels for embedding high-dimensional data. Recent work [7] proposed dual geometry approaches that embed the dataset X in a low-dimensional space using non-linear dimensionality reduction techniques that are consequently applied to rows and columns of X.

Another method that utilizes kernels, which is extended in this paper, is Laplacian pyramids [11, 28]. Laplacian pyramids is a multi-scale algorithm for learning functions over scattered high-dimensional data. The multi-scale representation is generated by convolving the dataset X with row-based kernels of decreasing widths. Then, the construction is used for approximation of a function f defined on X and its extension to new points. By slightly modifying the kernels to a have a zero diagonal, the method becomes more robust to noise and automatically finds the optimal kernel width for stopping the iterations [12, 13]. Other multi-scale kernel-based methods for function extension are described in [2], where the spectral decomposition is completed at each scale.

In this work, we extend the Laplacian pyramids technique to model the data by considering two directional connections. In the spirit of the dual geometry representations, the dataset X is modeled by multi-scale kernels that are constructed on the rows and the columns alternately, until the finest level is reached. The method is simple to implement and has no risk of not converging. It can be used to model a high-dimensional dataset X of size $M \times N$ and learn a function f of size $M \times N$ that is defined over X. One application that benefits from such a construction is completion of missing values in X. Imputation is just a special application of the proposed general setting, where the function f is the identify function: $f = X$.

Completion of missing data, also known as imputation, is a task that is often carried out as a pre-processing step in machine learning, signal processing, and other types of data analysis applications. Straightforward approaches include mean- and median-based imputations that replace the unknown entries with an average value, which is calculated based on known values in the same row or column [17, 22]. Alternately, the most frequent value or a random value, which is drawn from a distribution that describes the known data values, may be used to replace the missing value. The multiple imputation approach [3] estimates the missing data values multiple times and combines the results. Another common approach is the maximum likelihood-based imputation [9]. Regression-based methods are also used to complete missing data. A method based on stepwise regression is described in [33] and optimized linear imputation was recently proposed in [27]. Completion of missing data is related to the matrix completion problem, which is addressed by using nuclear norm minimization (see [4] and [23]), or by generalizations to other regularization other than the nuclear norm (see [31]).

For high-dimensional datasets, the low-dimensional intrinsic representation of the data can be utilized for imputation. In [1], a low-dimensional model is constructed and used for data imputation in road networks. An unsupervised regression-based approach, which constructs a low-dimensional representation of the data and imputes missing values, was proposed in [5]. A local low-rank matrix approximation model was introduces in [20]. Dimensionality reduction and clustering was applied for imputation of medical data [34]. A regression-based imputation method for high-dimensional datasets that imputes the missing data column by column based on dimensionality reduction and diffusion maps was proposed in [29].

Biological datasets, in particular those that describe gene expression, exhibit many missing or zero values. A linear dimensionality reduction technique to fill in zero values of single-cell gene expression was applied in [26]. Kernel-based methods were proposed by [10] for imputation of gene-gene iterations. Recently, an approach, which is based on low rank approximation and manages to preserve true zero entries in RNA sequencing data, was proposed by Linderman et al. [21].

Several papers address the problem of embedding high-dimensional data with missing values to a low-dimensional space. In [15], the authors propose a methods that constructs a distance matrix from the incomplete data and then uses a metric repair to correct the perturbed distances. A non-linear PCA with missing data approach, which is based on an inverse neural network model and applied to metabolite data, is presented in [30]. A PCA-based approach that approximates a low-rank structure for high-dimensional datasets, while enforcing graph smoothness assumptions on the rows and columns of the data is described in [32]. Mishne et al. recently proposed an unsupervised manifold learning method [24] that reveals the underlying geometry of a given matrix based on multi-scale information from the row and column spaces and also works in a missing data setting. In [25], the low algebraic dimension of a matrix is used to complete missing entries.

This paper is organized as follows. Section 2 reviews the one-directional Laplacian pyramids and the error analysis for the one-directional case. In Section 3, the new extension to the two-directional case is described and an error analysis is presented. The application and adaptation of the two-directional Laplacian pyramid for imputation is described in detail in Section 4. Finally, Section 5 provides experimental results that demonstrate the efficiency of our approach on a synthetic example and on a publicly available dataset.

2 One-directional Laplacian pyramid

Let $X = \{x_1, \ldots, x_n\}$ be a set of scattered data points, possibly high dimensional, and f be a function defined on X. The Laplacian pyramid algorithm provides a multi-scale approximation of f as a function of X. First, a coarse representation of f, denoted by f_0, is constructed by convolving f with a kernel K_0. Then, the residual $d_1 = f - f_0$ is computed and convoluted with a finer normalized kernel K_1. This results in a finer approximation of f, given by $f \approx f_0 + K_1 * d_1$. The iterative process is repeated until an optimal scale of f is reached. In what follows, this procedure is described in detail.

We start with a coarse Gaussian kernel $G_0 = \big(g_0(x_i, x_j)\big)$, having a large scale of σ_0, defined by

$$g_0(x_i, x_j) = e^{\frac{-\|x_i - x_j\|^2}{\sigma_0}}, \quad x_i, x_j \in X, \tag{1}$$

where x_i and x_j are the i^{th} and j^{th} rows of X, respectively. The associated normalized kernel to G_0 is $K_0 = \big(k_0(x_i, x_j)\big)$, which is defined by

$$(k_0)(x_i, x_j) = \frac{g_0(x_i, x_j)}{\sum_k g_0(x_i, x_k)}. \tag{2}$$

At a finer scale l, the Gaussian kernel G_l is defined by

$$g_l(x_i, x_j) = e^{-\|(x_i - x_j)\|^2/(\frac{\sigma_0}{2^l})}, \qquad l = 1, 2, 3, \ldots$$

Normalization of G_l yields the smoothing operator $K_l = (k_l(x_i, x_j))$, where

$$k_l(x_i, x_j) = \frac{g_l(x_i, x_j)}{\sum_k g_l(x_i, x_k)}, \quad l = 1, 2, 3, \ldots \tag{3}$$

The Laplacian Pyramid representation of f is iteratively defined as follows. For the first level $l = 0$, a smooth approximation of f is given by

$$f_0(x_i) = \sum_{j=1}^{n} k_0(x_i, x_j) f(x_j), \quad i = 1, \ldots, n, \quad x_i, x_j \in X. \tag{4}$$

Let

$$d_1(x_i) = f(x_i) - f_0(x_i), \quad i = 1, 2, \ldots, n \quad x_i \in X,$$

then a finer representation of f is

$$f_1(x_i) = f_0(x_i) + \sum_{i=1}^{n} k_1(x_i, x_j) d_1(x_j), \qquad i = 1, \ldots, n.$$

In general, for $l = 1, 2, 3 \ldots$,

$$d_l(x_i) = f(x_i) - f_{l-1}(x_i), \qquad i = 1, \ldots, n, \tag{5}$$

$$f_l(x_i) = f_{l-1}(x_i) + \sum_{j=1}^{n} k_l(x_i, x_j) d_l(x_j), \qquad i = 1, \ldots, n, \tag{6}$$

where f_0 is defined in Equation (4). Equation (6) approximates a given function f by the series of functions $\{f_0, f_1, f_2, \ldots\}$ in a multi-scale manner, going from a coarser to a finer representation. The functions $\{f_0, f_1, f_2, \ldots\}$ can be easily extended to a new point $\bar{x}$ in the following way.

$$f_0(\bar{x}) = \sum_{j=1}^{n} k_0(\bar{x}, x_j) f(x_j) \qquad \text{for} \quad l = 0, \tag{7}$$

$$f_l(\bar{x}) = f_{l-1}(\bar{x}) + \sum_{j=1}^{n} k_l(\bar{x}, x_j) d_l(x_j) \qquad \text{for} l = 1, 2, 3, \ldots, \tag{8}$$

where $d_l(x_j)$ is defined in Equation (5).

2.1 Auto-adaptive Laplacian pyramids

The Laplacian Pyramids iterations may be stopped by setting an admissible error to a small threshold, defined by *err*, and requiring $\|f - f_l\| < err$. If *err* is chosen to be too large, then the iterations stop at a coarse scale, thus the approximation does not capture finer structures of the function f. If *err* is chosen to be too small, then in finer scales, it results in a situation where a point may have few or no neighbors; thus, over-fitting may occur. The auto-adaptive Laplacian Pyramids, which were introduced in [12, 13], slightly modify the kernels constructed in Equations (1) and (3). This prevents over-fitting of data and provides a criteria for selecting a proper stopping scale l. The main modification is to replace the kernels $G_l = \big(g_l(x_i, x_j)\big)$ by $\tilde{G}_l$, which are defined by

$$\tilde{G}_l(x_i, x_j) = \begin{cases} G_l(x_i, x_j) & i \neq j \\ 0 & i = j. \end{cases} \tag{9}$$

These yield the normalized operators

$$\tilde{k}_l(x_i, x_j) = \frac{\tilde{g}_l(x_i, x_j)}{\sum_k \tilde{g}(x_i, x_k)}, \quad l = 1, 2, 3, \ldots \tag{10}$$

and the iterative construction

$$f_0(x_i) = \sum_{j=1}^{n} \tilde{k}_0(x_i, x_j) f(x_j) \qquad \text{for level} \quad l = 0 \tag{11}$$

$$f_l(x_i) = f_{l-1}(x_i) + \sum_{j=1}^{n} \tilde{k}_l(x_i, x_j) d_l(x_j) \qquad \text{for level} \quad l = 1, 2, \ldots. \tag{12}$$

Extension to new points is done in a similar manner, $\bar{x}$ replaces x_i in Equations (11) and (12).

Using the above modification results in a leave-one-out cross-validation that is inherent in the algorithm, where each training point $x \in X$ is treated as a test point. The approximation of f at x_i is built without using the value of the point itself, so that the contribution is only from $x_i's$ neighboring points. This modification makes the procedure more robust in the presence of noise. The stopping scale l is determined by computing the mean square error $err_l = \|f - f_l\|$ at each level and choosing the stopping scale l for which the minimum value of the vector err_l occurs. To conclude, this procedure is equivalent to applying the Laplacian Pyramids algorithm in a leave-one-out cross-validation manner and choosing the stopping scale l for which the error is minimal.

The cost of running L steps of the ALP is $O(LN^2)$, where $N \times N$ is the dimension of the sampling kernel. This is compared to the cost of LOOCV (leave-one-out cross-validation, which is $O(LN^3)$ operations.

2.2 Error analysis for the one-sided LP scheme

For the sake of completeness of the presentation, in this section, we review the error analysis for the Laplacian pyramids procedure, described in [12]. Assume that f is in L_2, i.e., $\int_x f^2(x)dx \leq K$, for some constant K. The LP scheme is a relaxation process for which in the first step the function f is approximated by

$$f_0(x) = (k_0 * f)(x). \tag{13}$$

Here, we use kernels of the form $k_l(x)$, which approximate a delta function, satisfying

$$\begin{array}{l} \int k_l(x)\,dx = 1, \\ \int x k_l(x)\,dx = 0, \\ \int |x|^2 |k_l(x)|\,dx \leq C. \end{array} \tag{14}$$

Note that k_l are normalized kernels. As an example, one may choose

$$k_l = c_l e^{-x^2/\sigma_l}, \qquad \sigma_l = \sigma_0/\mu^l, \tag{15}$$

where c_l is a normalizing factor for k_l. Define

$$d_1 = f - f_0,$$

then, in the second step, f is approximated by

$$f_1 = f_0 + k_1 * d_1. \tag{16}$$

Taking the Fourier transform of $k_l(x)$ and using the assumptions (14), we have (see [14])

$$\begin{array}{l} \left|\hat{k}_l(\omega) - 1\right| \leq C_1 \sigma_l^2 \|\omega\|_2^2, \quad l = 0, 1, 2, .., \text{ where} \\ \\ C_1 = \frac{1}{2}\int_{-\infty}^{\infty} |x|^2 |k_l(x)|\,dx. \end{array} \tag{17}$$

Let us analyse the error in the first step. The error $d_1(x)$ is defined by

$$d_1(x) = f(x) - (k_0 * f)(x). \tag{18}$$

Taking the Fourier transform of $d_1(x)$ and using Equation (17), we have

$$\left|\hat{d}_1(\omega)\right| = \left|\hat{f}(\omega)\right| \left|\hat{k}_0(\omega) - 1\right| \leq C\sigma_0^2 \|\omega\|_2^2 \left|\hat{f}(\omega)\right|, \tag{19}$$

where C is a universal constant. The error in the second step is

$$d_2 = d_1 - k_1 * d_1 = (k_0 * f - f) - k_1 * d_1. \tag{20}$$

Taking the Fourier transform of Equation (20) yields

$$\left|\hat{d}_2(\omega)\right| = \left|\hat{d}_1(\omega) - \hat{d}_1(\omega)\hat{k}_1(\omega)\right| = \left|\hat{d}_1(\omega)\right| \left|\hat{k}_1(\omega) - 1\right|. \tag{21}$$

Using Equations (20) and (21), we obtain

$$\begin{array}{l} \left|\hat{d}_2(\omega)\right| \leq C \left|\hat{d}_1(\omega)\right| \sigma_1^2 \|\omega\|_2^2 \leq C\sigma_0^2\sigma_1^2 \|\omega\|_2^4 \left|\hat{f}(\omega)\right| \\ \qquad = C\sigma_0^2(\sigma_0/\mu)^2 \|\omega\|_2^4 \left|\hat{f}(\omega)\right|. \end{array} \tag{22}$$

For the l^{th} level, the error is bounded by

$$\left|\hat{d}_l(\omega)\right| \leq C\sigma_0^2 \left(\frac{\sigma_0^2}{\mu^l}\right)^{l-1} \|\omega\|_2^{2l} \left|\hat{f}(\omega)\right|. \tag{23}$$

By Parseval's equality, we obtain

$$\|d_l(x)\|_{L^2} \leq C\sigma_0^2 \left(\frac{\sigma_0^2}{\mu^l}\right)^{l-1} \|f(x)\|_{2l,2}\,. \tag{24}$$

Thus, the error decays faster than any algebraic rate.

3 Two-directional Laplacian pyramids

Given a function $f = f(x, y)$ of size $M \times N$, the Laplacian pyramids method is modified to be a two-direction approximation, which takes into account the relationship between the rows and columns of f. At each scale l, two kernels are constructed. The kernels are based on the pairwise distances between the rows and columns of f, and are denoted by $G_l^{(L)}$ and $G_l^{(R)}$, for left (L) and right (R) respectively. Denote the associated normalized kernels by $K_l^{(L)}$ and $K_l^{(R)}$. First, f is coarsely approximated by

$$f_0 = K_0^{(L)} * f * K_0^{(R)}.$$

Next, the difference

$$d_1 = f - f_0,$$

is calculated and it becomes the input for the next finer approximation. In the second step, f is approximated by

$$f_1 = f_0 + K_1^{(L)} * d_1 * K_1^{(R)}.$$

After $l - 1$ iterations, the difference between f and its multi-scale representation is given by

$$d_l = f - f_{l-1},$$

and at the l-th iteration, the a finer version of f is

$$f_l = f_{l-1} + K_l^{(L)} * d_l * K_l^{(R)}. \tag{25}$$

3.1 Error analysis for the two-sided LP scheme

Assume that f is in L_2, i.e., $\int_x f^2(x)dx \leq K$, for some constant K.

Define the kernels $k_l^{(L)}(x)$ and $k_l^{(R)}(x)$ which approximate a delta function. The kernel $k_l^{(L)}$ operates on f from the left and $k_l^{(R)}$ operates on f from the right. The two kernels satisfy equation (14).

Note that $k_l^{(L)}$ and $k_l^{(R)}$ are normalized kernels. As an example, one may choose

$$\begin{aligned} k_l^{(L)} &= c_l^{(L)} e^{-x^2/\sigma_l^{(L)}}, \qquad \sigma_l^{(L)} = \sigma_0^{(L)}/\mu^l \\ k_l^{(R)} &= c_l^{(R)} e^{-x^2/\sigma_l^{(R)}}, \qquad \sigma_l^{(R)} = \sigma_0^{(R)}/\mu^l, \end{aligned} \tag{26}$$

where $c_l^{(L)}$ and $c_l^{(R)}$ are normalizing factor for $k_l^{(L)}$ and $k_l^{(R)}$ respectively.

The two-sided LP scheme is a relaxation process for which in the first step, the function f is approximated by

$$f_0 = k_0^{(L)} * f * k_0^{(R)}.$$

Define

$$d_1 = f - f_0,$$

then, in the second step, f is approximated by

$$f_1 = f_0 + k_1^{(L)} * d_1 * k_1^{(R)}$$

Taking the Fourier transform of $k_l^{(L)}(x)$ and using the assumptions (14) for $k_l^{(L)}(x)$, we have (see [14])

$$\begin{aligned} &\left|\hat{k}_1^{(L)}(\omega) - 1\right| \le C(\sigma_l^{(L)})^2 \|\omega\|_2^2, \text{ where} \\ &C = \tfrac{1}{2}\int_{-\infty}^{\infty} x^2 \left|k_1(x)\right| dx. \end{aligned} \tag{27}$$

Similarly for $k_1^{(R)}$. We first analyse the error in the first step. The error $d_1(x)$ is defined by

$$d_1(x) = f(x) - (k_0^{(L)} * f * k_0^{(R)})(x).$$

Taking the Fourier transform of $d_1(x)$ and using Equation (27), we have

$$\begin{aligned} \left|\hat{d}_1(\omega)\right| &= \left|\hat{k}_0^{(L)}\hat{f}(\omega)\hat{k}_0^{(R)} - \hat{f}(\omega)\right| \\ &\le \left|\hat{k}_0^{(L)}\hat{f}(\omega)\hat{k}_0^{(R)} - \hat{f}(\omega)k_0^{\hat{(R)}}\right| + \left|\hat{f}(\omega)\hat{k}_0^{(R)} - \hat{f}(\omega)\right| \\ &= \left|(\hat{k}_0^{(L)} - 1)\hat{f}(\omega)\hat{k}_0^{(R)}\right| + \left|\hat{f}(\omega)(\hat{k}_0^{(R)} - 1)\right|. \end{aligned} \tag{28}$$

First, note that by Taylor expansion of $\left|\hat{k}_0^{(R)}(\omega)\right|$ around $\omega = 0$, and by using Equation (14) for $k_l^{(R)}(x)$, it follows that is bounded $\left|\hat{k}_0^{(R)}\right|$ is bounded by a constant. Bounding the two terms on the right-hand-side of (28), we have

$$\begin{aligned} \left|(\hat{k}_0^{(L)} - 1)\hat{f}(\omega)\hat{k}_0^{(R)}\right| &\le C(\sigma_0^{(L)})^2 \|\omega\|_2^2 \left|\hat{k}_0^{(R)}\right| \left|\hat{f}(\omega)\right| \\ &\le C(\sigma_0^{(L)})^2 \|\omega\|_2^2 \left|\hat{f}(\omega)\right| \end{aligned} \tag{29}$$

and

$$\left|\hat{f}(\omega)(\hat{k}_0^{(R)} - 1)\right| \le C(\sigma_0^{(R)})^2 \|\omega\|_2^2 \left|\hat{f}(\omega)\right|. \tag{30}$$

Here, C denotes a universal constant.

Combining Equations (28), (30), and (29), we have

$$\left|\hat{d}_1(\omega)\right| \le C((\sigma_0^{(R)})^2 + (\sigma_0^{(L)})^2) \|\omega\|_2^2 \left|\hat{f}(\omega)\right|. \tag{31}$$

For simplicity, we assume that $\sigma_0^{(L)}$ and $\sigma_0^{(R)}$ are bounded by σ_0. Thus,

$$\left|\hat{d}_1(\omega)\right| \le C\sigma_0^2 \|\omega\|_2^2 \left|\hat{f}(\omega)\right|. \tag{32}$$

The error in the second step is

$$d_2(x) = d_1 - k_1^{(L)} * d_1 * k_1^{(R)}. \tag{33}$$

Taking the Fourier transform of Equation (33) yields

$$\left|\hat{d}_2(\omega)\right| = \left|\hat{k}_1^{(L)}\hat{d}_1(\omega)(\omega)\hat{k}_1^{(R)} - \hat{d}_1(\omega)\right|. \tag{34}$$

It may be bounded as in Equation (32) by

$$\begin{aligned}\left|\hat{d}_2(\omega)\right| &\le C((\sigma_1^{(R)})^2 + (\sigma_1^{(L)})^2)\|\omega\|_2^2 \left|\hat{d}_1(\omega)\right| \\ &\le C(\sigma_0)^2(\sigma_0/\mu)^2\|\omega\|_2^4|\hat{f}(\omega)|.\end{aligned} \tag{35}$$

For the l^{th} level, the error is bounded by

$$\left|\hat{d}_l(\omega)\right| \le C\sigma_0^2 \left(\frac{\sigma_0^2}{\mu^l}\right)^{l-1} \|\omega\|_2^{2l} \left|\hat{f}(\omega)\right|. \tag{36}$$

By Parseval's equality, we obtain

$$\|d_l(x)\|_{L^2} \le C\sigma_0^2 \left(\frac{\sigma_0^2}{\mu^l}\right)^{l-1} \|f(x)\|_{2l,2}. \tag{37}$$

Thus, the error for the two-sided LP procedure as well decays faster than any algebraic rate.

4 Two-directional Laplacian pyramids for data imputation

This section describes how to utilize the two-directional multi-scale construction for imputation of missing data for a rectangular dataset. Denote the dataset with missing values by $X = (x_{ij})$, a matrix of size $M \times N$. In order to utilize the connections between the rows and columns in X, the dataset needs to be normalized. Thus, each column (or each row) should be adjusted such that its mean equals zeros and its variance equals one. Let $B = (b_{ik})$ be a binary indicator matrix of size $M \times N$ that specifies the missing data locations in X. Thus, if $b_{ik} = 1$ then x_{ik} contains a known value and if $b_{ik} = 0$ then x_{ik} is a missing data entry.

First, in Section 4.1, we describe how the coarse normalized kernels $K_0^{(L)}$ and $K_0^{(R)}$ are constructed from X and B. Next, in Section 4.2, the imputation algorithm is reviewed in detail and applied to a synthetic example. Last, Section 4.3 summarizes the imputation method in an algorithm format.

4.1 Construction of the initial coarse kernels

Given the dataset X and its corresponding indicator matrix $B = (b_{ik})$, the two initial coarse kernels are constructed based on the known entries of X (similar to the distance defined in [15]). Here, Gaussian kernels denoted by $G_0^{(L)}$ and $G_0^{(R)}$ are used to define the normalized kernels $K_0^{(L)}$ and $K_0^{(R)}$. Recall that

$$G_0^{(L)} = g_0^{(L)}(x_i, x_j) = e^{\frac{-\|x_i - x_j\|^2}{\sigma_0^{(L)}}}, \quad x_i, x_j \in X,$$

where x_i and x_j are the i^{th} and j^{th} rows of X. The distance $\|x_i - x_j\|$ in the exponent is computed as follows.

$$\|x_i - x_j\|^2 = \sum_{\substack{k=1 \\ b_{ik}=b_{jk}=1}}^{M} |x_{ik} - x_{jk}|^2.$$

Therefore, the entries which are included in this distance contains only indices k for which both x_{ik} and x_{jk} are known. This process results in a full matrix $G_0^{(L)}$.

Next, the auto-adaptive modification is performed on the kernel $G_0^{(L)}$. As described in Section 2.1, the diagonal of $G_0^{(L)}$ is set to zero and the kernel $\tilde{G}_0^{(L)}$ is normalized to be the smoothing operator $\tilde{K}_0^{(L)}$.

The same process is applied for construction of $G_0^{(R)}$. Let x^i and x^j be two columns of the matrix X, then the elements of $G_0^{(R)}$ are given by

$$G_0^{(R)} = g_0^{(R)}(x^i, x^j) = e^{\frac{-\|x^i - x^j\|^2}{\sigma_0^{(R)}}}, \quad x^i, x^j \in X.$$

The distance $\|x^i - x^j\|$ is then computed by $\|x^{k,i} - x^{k,j}\|$ where k satisfies $b_{ki} = b_{kj} = 1$. Thus,

$$\|x^i - x^j\|^2 = \sum_{\substack{k=1 \\ b_{ki}=b_{kj}=1}}^{N} |x_{ki} - x_{kj}|^2.$$

Therefore, the entries which are included in this distance contains only indices k for which both x_{ki} and x_{kj} are known. Last, a normalization of $G_0^{(R)}$ yields the coarse kernel $K_0^{(R)}$.

In order to utilize the automatic stopping criteria procedure that was described in Section 2.1, it is recommended to modify just the left kernel $G_0^{(L)}$ to be $\tilde{G}_0^{(L)}$. Modifying $G_0^{(R)}$ to $\tilde{G}_0^{(R)}$ may result in over-smoothing of the data, especially in cases where the number of columns M is small.

The initial kernel widths $\sigma_0^{(L)}$ and $\sigma_0^{(R)}$ can be set by estimating the pairwise distances between the rows for the left kernel and between the columns of X for the right kernel. Here, the following *MaxMin* heuristic (see [19]) is used

$$\sigma_0^{(L)} = C^{(L)} \cdot \max_j [\min_{i, i \neq j} (\|x_i - x_j\|)^2]$$

and

$$\sigma_0^{(R)} = C^{(R)} \cdot \max_j [\min_{i, i \neq j} (\|x^i - x^j\|)^2].$$

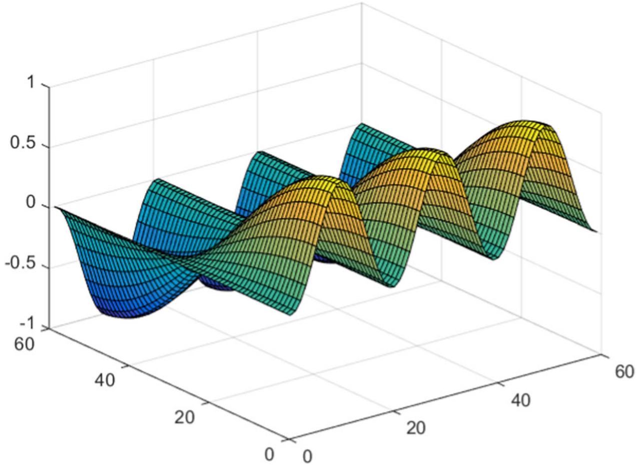


Fig. 1 Synthetic dataset $X = (x_{i,j}$, where $x_{ij} = f(s_i, t_j) = sin^3(s_i)cos(t_j)$

It is possible to choose different values for $C^{(L)}$ and $C^{(R)}$. In this paper, we experiment with two different choices; the first is $C^{(L)} = 2$,$C^{(R)} = 2$ and the second is $C^{(L)} = 2$,$C^{(R)} = 0.5$.

4.2 Multi-scale construction

In this section, the two directional multi-scale construction is described and applied to a synthetic example. Consider the dataset $X = (x_{ij})$ of size 60 × 60, where $x_{ij} = f(s_i, t_j) = \sin^3(s_i)\cos(t_j)$. Here, $0 \leq s_i, t_j \leq \pi$, with an equal spacing of $\frac{\pi}{60}$ between the points in both directions. The dataset X with no missing values is plotted in Fig. 1.

In this running example 20% of the values in X, which were chosen randomly, are missing. Figure 2 displays on the left X with the missing values and on the right its associated binary index matrix B. To enable a reasonable visualization of X with

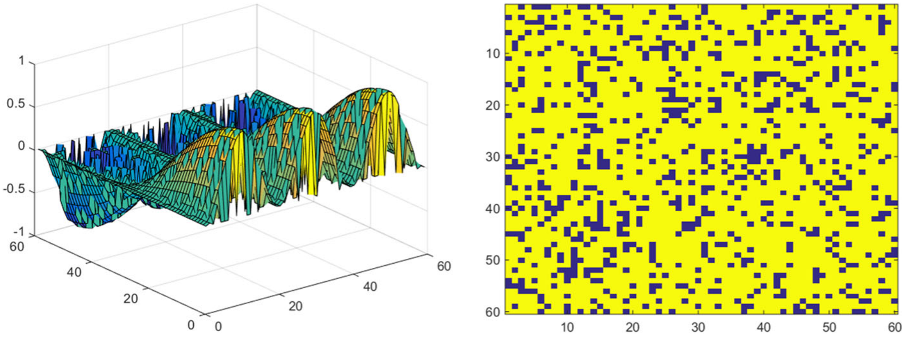


Fig. 2 Left: The dataset X with 20% missing values. Right: The associated index matrix B

missing values, entries with missing data were replaced by the value 0, this value is not used in the algorithm. On the right, the entries of B that satisfy $b_{ij} = 1$ are colored in yellow and the locations of missing entries of X that satisfy $b_{ij} = 0$ are colored in blue.

Two initial coarse kernels $\tilde{K}_0^{(L)}$ and $K_0^{(R)}$ are constructed from X as described in the previous Subsection (4.1). The initial values for the kernels' scales were set to be $\sigma_0^{(L)} = \sigma_0^{(R)} = 10$. In order to convolve them with X, X needs to be a full matrix. We construct the matrix $X^\star$ by simply imputing the missing values in X with the mean value of the known data entries. Denote this mean value by $m^\star = \text{mean}\left(X_{ij}\right)$, where ij are indices that satisfy $b_{ij} = 1$. Thus, $X^\star$ is constructed as follows

$$X^\star_{ij} = \begin{cases} X_{ij}, & \text{if} b_{ij} = 1 \\ m^\star = \text{mean}\left(X_{ij}\right), & \text{if} b_{ij} = 0. \end{cases} \tag{38}$$

Then, a coarse approximation of the dataset is computed by

$$X_{\mathbf{0}} = \tilde{K}_0^{(L)} * X^\star * K_0^{(R)}. \tag{39}$$

The two-sided convolution in Eq. 39 is computed in two consecutive steps. Convolving $X^\star$ with $\tilde{K}_0^{(L)}$ yields

$$X_{\mathbf{0}}^{(L)} = \tilde{K}_0^{(L)} * X^\star,$$

which approximates the data by averaging its rows, as described in Eq. 4. In summation notation, $X_{\mathbf{0}}^{(L)}$ is evaluated row-by-row as follows

$$X_{\mathbf{0}\,i:}^{(L)} = \sum_{j=1}^{M} \tilde{K}_{0\,ij}^{(L)} \cdot X^\star_{j:}, \quad i = 1, \ldots, M, \tag{40}$$

where $X_{\mathbf{0}\,i:}^{(L)}$ is the $i - th$ row in $X_{\mathbf{0}}^{(L)}$, and $X^\star_{j:}$ is the $j - th$ row in $X^\star$. The second convolution is performed by

$$X_{\mathbf{0}} = X_{\mathbf{0}}^{(L)} * K_0^{(R)}.$$

In summation notation,

$$X_{\mathbf{0}\,:j} = \sum_{i=1}^{N} K_{0\,ij}^{(R)} \cdot X_{\mathbf{0}\,:i}^{(L)}, \quad j = 1, \ldots, N, \tag{41}$$

where $X_{\mathbf{0}\,:j}$ is the $j - th$ column in $X_{\mathbf{0}}$, and $X_{\mathbf{0}\,:i}^{(L)}$ is the $i - th$ column in $X_{\mathbf{0}}^{(L)}$.

The error between the known data values and their coarse approximation is computed and stored in $err_{\mathbf{0}}$. It is calculated based on the difference

$$err_{\mathbf{0}} = \|X_{ij} - X_{\mathbf{0}\,ij}\|,$$

where the only indices ij for which $b_{ij} = 1$ are included in the computation of the error. Here, $err_{\mathbf{0}}$ is computed as the root mean square error.

The first residual is given by $D_{\mathbf{1}} = X - X_{\mathbf{0}}$. The values of $D_{\mathbf{1}}$ are known for all locations ij that satisfy $b_{ij} = 1$. The matrix $D_{\mathbf{1}}^\star$ is then constructed as described in Eq. 38, the missing values in $D_{\mathbf{1}}$ are replaced by the mean of its known entries. The

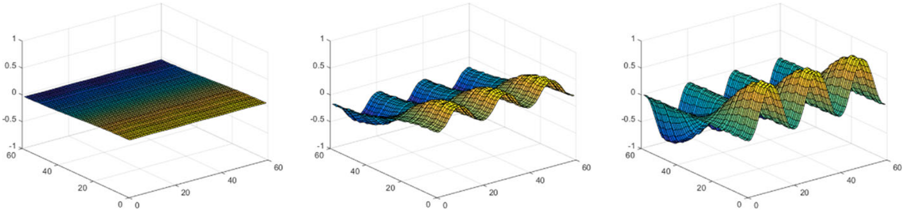


Fig. 3 Multi-scale approximation and imputation of X. Left: $X_{\mathbf{0}}$, middle: $X_{\mathbf{2}}$, right: $X_{\mathbf{5}}$

kernels $\tilde{K}_1^{(L)}$ and $K_1^{(R)}$, which operate on $D_{\mathbf{1}}^{\star}$ from the left and from the right, yield a more accurate representation of X, denoted by $X_{\mathbf{1}}$. It is given by

$$X_{\mathbf{1}} = X_{\mathbf{0}} + \tilde{K}_1^{(L)} * D_{\mathbf{1}}^{\star} * K_1^{(R)}.$$

The construction is carried out in an iterative manner and the iterations continue for a pre-defined maximal number of steps (here set to 20). In the $l - th$ iteration the residual $D_l = X - X_{l-1}$ is computed, $D_l^{\star}$ is constructed. Then, operating on $D_l^{\star}$ from the left and the right by $\tilde{K}_l^{(L)}$ and $K_l^{(R)}$ respectively, yields a finer representation of X, which is given by

$$X_l = X_{l-1} + \tilde{K}_l^{(L)} * D_l^{\star} * K_l^{(R)}.$$

The error, denoted by err_l, is calculated based on the difference between X and X_l on the known data entries. Figure 3 displays the multi-scale approximation and imputation construction for the synthetic example $X_{\mathbf{0}}$, $X_{\mathbf{2}}$, $X_{\mathbf{5}}$ corresponding to the scales $l = 0, 2, 5$, respectively. Figure 4 shows the corresponding multi-scale kernels.

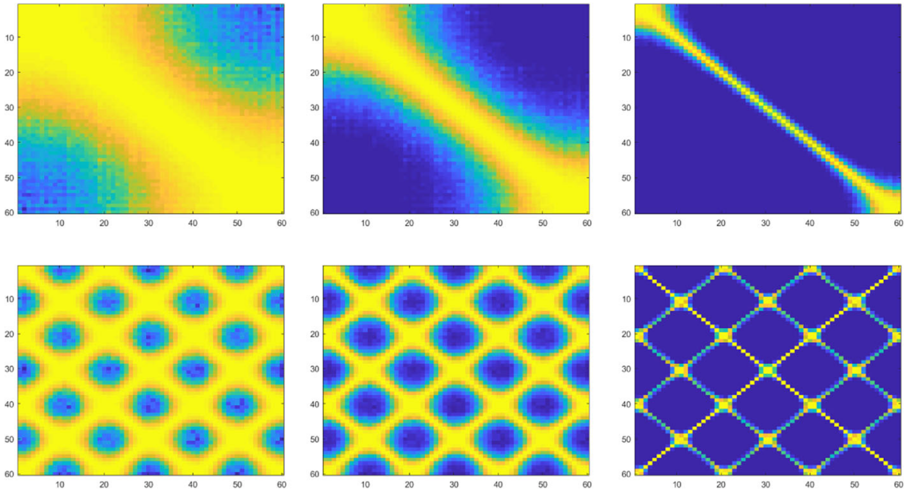


Fig. 4 Multi-scale kernels. In the first row, $K_0^{(L)}$, $K_2^{(L)}$, and $K_5^{(L)}$ are plotted from left to right, respectively. The second row contains $K_0^{(R)}$, $K_2^{(R)}$, and $K_5^{(R)}$

In the first row, $K_0^{(L)}$, $K_2^{(L)}$, and $K_5^{(L)}$ are plotted from left to right, respectively. The second row contains $K_0^{(R)}$, $K_2^{(R)}$ and $K_5^{(R)}$. High kernel values (close to 1) are plotted in yellow, while values close to 0 are plotted in blue. It may be seen that the different geometries along the row and the columns are captured accordingly by the left and right kernels. In addition, as the scale l increases, sharper kernels, which are consecrated in narrower regions, are formed.

The calculated root mean square errors, err_l on the known data entries in each of the 20 iterations are plotted in Fig. 5. It can be seen that the optimal stopping scale, which holds the smallest error, is $l = 5$. This means that the imputed values for X should be taken from $X_\mathbf{5}$. This automatic stopping scale detection is achieved due to the replacement of the left kernels $K_l^{(L)}$ by $\tilde{K}_l^{(L)}$.

Last, it is demonstrated in Fig. 6 how the error for the missing data too decays as it decays for the known data entries. The root mean square errors for scales $l = 0, \ldots, 5$ are computed for the known and for the missing data entries and are plotted in blue and yellow bars respectively.

The proposed method is suitable for imputing data with a large number of missing entries. To demonstrate this advantage, the previous synthetic example is used, this time 60% of the data is missing. Figure 7 top-left shows the complete, full, dataset, on the top-right the indicator matrix of missing data is presented (blue stands for missing values), the data with the missing values set to zero is plotted on the bottom-left and the imputation result are plotted on the bottom right. Recall that only the known values of X are used, setting the missing values to zero is done solely for visualization.

Figure 8 shows how the auto-adaptive construction (see Section 2.1) results in an optimal scale, here $l = 8$, in which err_l reaches a minimum.

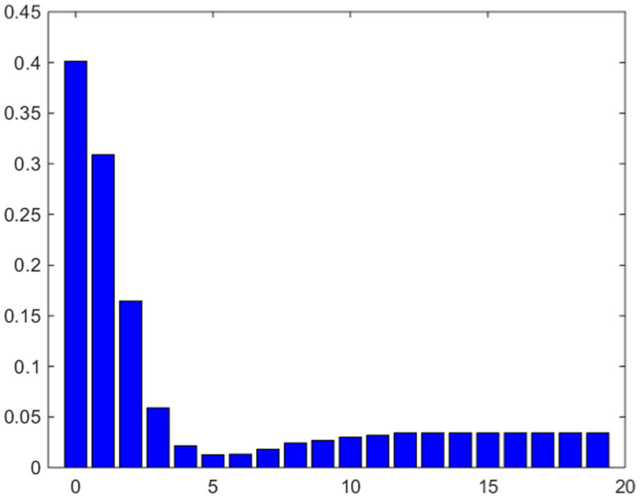


Fig. 5 The root mean square errors err_l that were calculated on the known entries in each of the 20 executed iterations. The minimum error is reached at $l = 5$

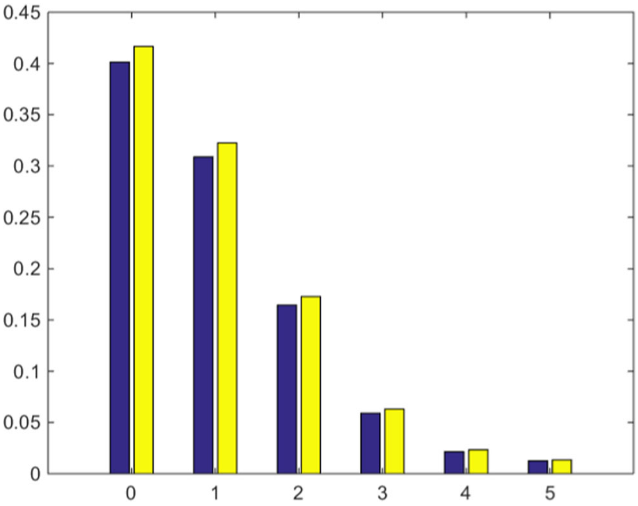


Fig. 6 The root mean square errors $err_l, l = 0, \ldots, 5$ on the known data entries in blue and on the missing data entries in yellow

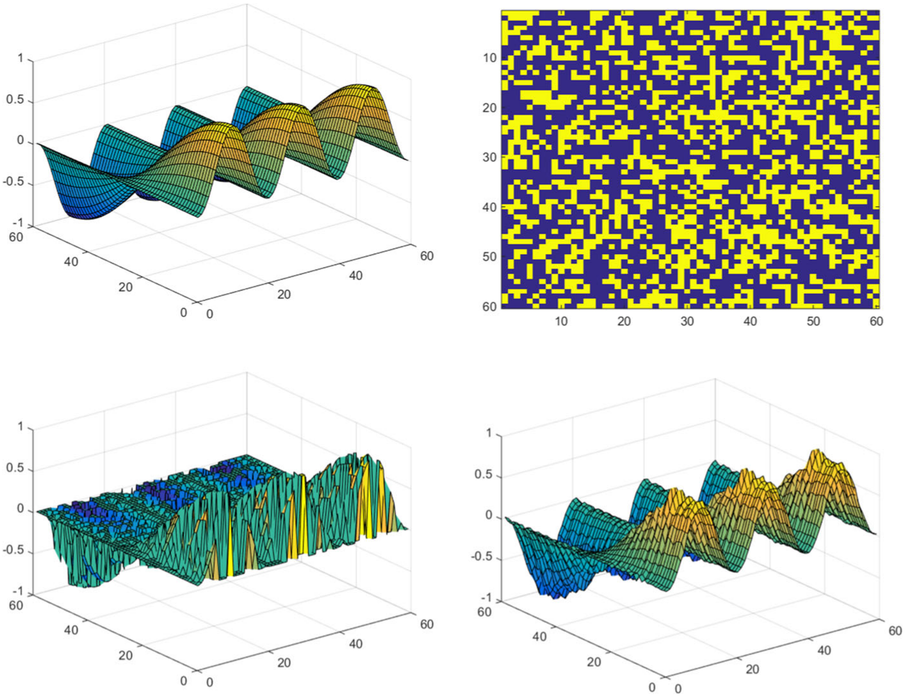


Fig. 7 Top-left: The original dataset X. Top-right: The indicator matrix B. Bottom-left: An illustration of X with missing data. Bottom-right: The imputed dataset X_8

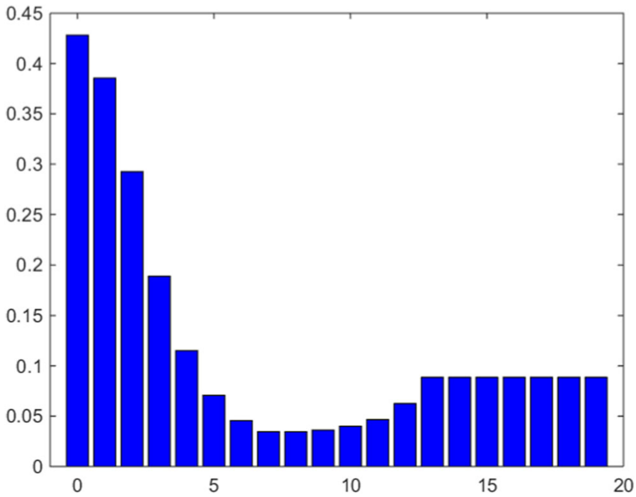


Fig. 8 The root mean square errors err_l that were calculated on the known entries in each of the 20 executed iterations. The minimum error is reached at $l = 8$

The multi-scale approximation and imputation is demonstrated in Fig. 9 for three levels, $l =$. The data matrix X_7 that is presented on the right holds the final imputed version for the original dataset X that had in it missing values.

Figure 10 plots the root mean square errors on the known points and on the missing points in each iteration in blue and yellow bars respectively.

4.3 The imputation algorithm

The proposed approach is summarized in the following algorithm.

5 Experimental results

This section demonstrates the application of the imputation algorithm to several public datasets that were downloaded from the UCI repository (http://archive.ics.uci.edu/ml/datasets). In all the examples, each feature (column) of each dataset was normalized to have mean 0 and standard deviation 1 in order to make the error values

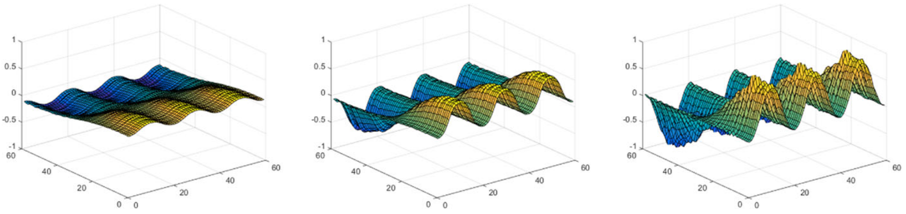


Fig. 9 Multi-scale approximation and imputation of X. Left: X_2, middle: X_4, right: X_8

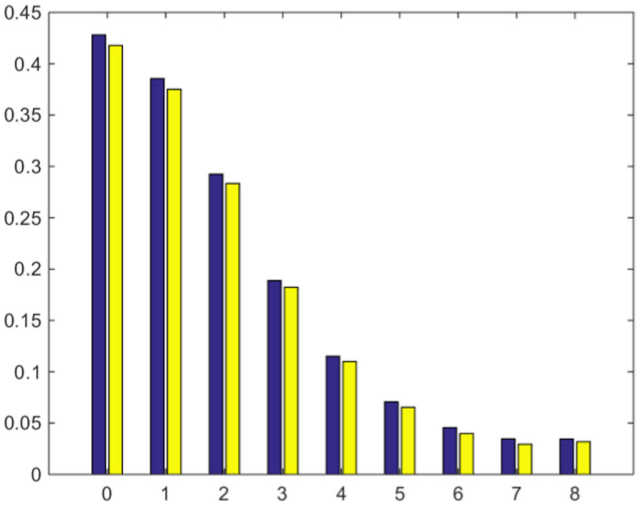


Fig. 10 The root mean square errors err_l, $l = 0, \ldots, 8$ on the known data entries in blue and on the missing data entries in yellow

Algorithm 1 Imputation with two-directional Laplacian pyramids.

Input:

- Dataset X of size $M \times N$, normalized, with missing values.
- A location indicator matrix B of size $M \times N$.
- $\sigma_0^{(L)}, \sigma_0^{(R)}$ - initial kernel widths.
- l_{max} - maximum number of iterations.

Output:

- Multi-scale imputed representation of X: $\{X_0, X_1, \cdots, X_l\}$.
- The values in $X_{l\ ih}$ for which $b_{ij} = 0$ are the imputed data for the missing values in X.

```
Construct $\tilde{K}_0^{(L)}$ and $K_0^{(R)}$ (see Section 4.1).
Construct $X^\star$ (see Eq. 38).
$X_{\mathbf{0}} = \tilde{K}_0^{(L)} * X^\star * K_0^{(R)}$.
Compute the root mean square error $err_0$ and store it in $err[0] = err_0$
for l=1 to $l_{max}$ do
    $D_l = X - X_{l-1}$.
    $D_l^\star$.
    $X_l = X_{l-1} + \tilde{K}_l^{(L)} * D_l^\star * K_l^{(R)}$.
    $err[l] = err_l$
end for
Determine the scale $l$ for which $err[l]$ reaches its minimum value.
return $\{X_0, X_1, \cdots, X_l\}$, where $X_l$ is the final result.
```

Table 1 Errors for the Mice dataset with $C^{(L)} = 2$, $C^{(R)} = 2$

% missing	1D Pyds	2D Pyds	Mean	Freq.
20% (MSE)	0.1545	0.1468	1.0047	9.5592
20% (RMSE)	0.3931	0.3831	1.0024	3.0918
50% (MSE)	0.2722	0.2702	0.9999	9.5419
50% (RMSE)	0.5217	0.5198	0.9999	3.0890
80% (MSE)	0.5756	0.5923	1.0056	7.8405
80% (RMSE)	0.7587	0.7696	1.0028	2.8001

comparable between the datasets (see [27]). The first dataset is the Mice Protein Expression Data Set, which contains expression levels of 77 proteins and has many missing values. In order to be able to evaluate the results, a smaller, complete dataset X, of size $M \times N = 1000 \times 66$ is extracted from the original data. The dataset X is normalized such that each column has zero mean and variance one. We examine three different settings: (i) : 20% of the data is missing, (ii) : 50% of the data is missing, and (ii) : 80% of the data is missing. For each mode, Algorithm 1 is applied ten times, where at each time the missing data locations are chosen at random. The results are compared with standard techniques; The first replaces the missing values in each column by its mean value and the second replaces the missing values by the most frequent value.

Tables 1 and 2 contain the average mean square errors (MSE) and the average root mean square errors (AMSE) for the ten executions of the proposed algorithm (denoted by Pyds) and the two alternative imputation techniques (Mean co. value and Frequent col. value) and compared to the one-dimensional (1D) Laplacian pyramids algorithm in each of the three settings (20%, 50%, and 80%). The experiments were executed with two different initial values for $C^{(L)}$ and $C^{(R)}$ (see Section 4.1); the first setting (see Table 1) is $C^{(L)} = 2$, $C^{(R)} = 2$ and the second setting is $C^{(L)} = 2$, $C^{(R)} = 0.5$ (see Table 2). In most of the cases, the two-directional algorithm performed better than the one-directional method. Only in one case, when the missing percentage of data was 80% and the initial column sigma value was picked to be relatively wide, the one-directional scheme achieved better results. Figure 11 shows

Table 2 Errors for the Mice dataset with $C^{(L)} = 2$, $C^{(R)} = 0.5$

% missing	1D Pyds	2D Pyds	Mean	Freq.
20% (MSE)	0.1460	0.1444	1.0065	9.9686
20% (RMSE)	0.3821	0.3800	1.0033	3.1573
50% (MSE)	0.2603	0.2565	0.9959	10.0160
50% (RMSE)	0.5102	0.5065	0.9979	3.1648
80% (MSE)	0.5691	0.5655	1.0024	8.4540
80% (RMSE)	0.7544	0.7520	1.0012	2.9076

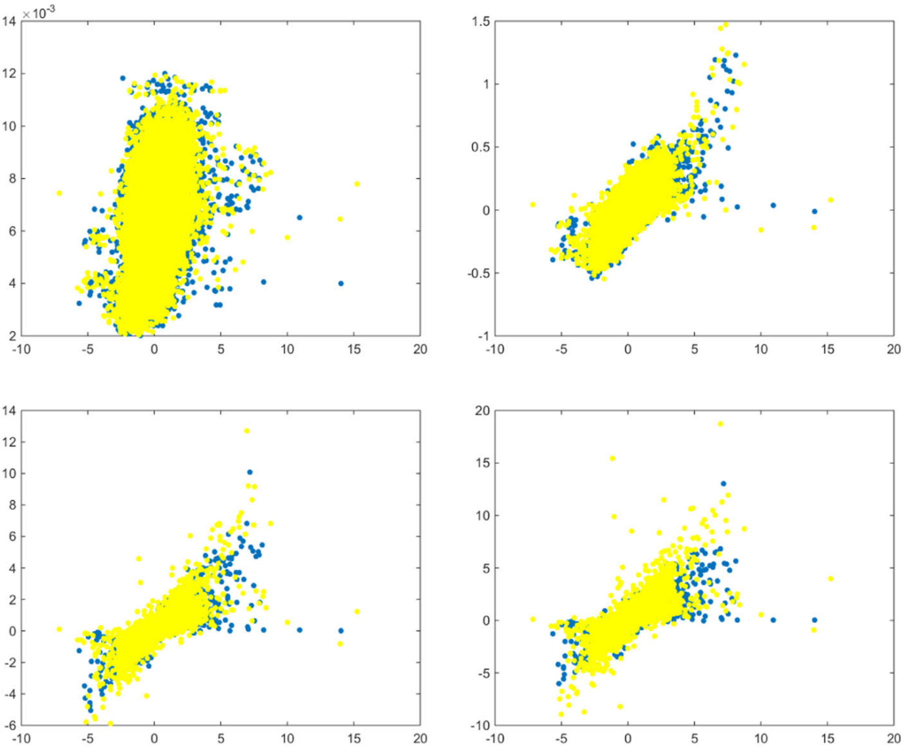


Fig. 11 Multi-scale approximation in blue and imputation in yellow for scale $l = 2$ (top left), $l = 5$ (top right), $l = 7$ (bottom left) and $l = 8$ (bottom right)

several steps of the algorithm's multi-scale construction of the data for the 50% case. In blue, the known values in X (x-axis) versus their approximated values (y-axis) for scales $l = 2, 5, 7, 8$ are plotted. In yellow, the multi-scale imputation of the missing values is plotted. Note the the y-axis has a different scale in each sub-figure, as the iterations proceed it spans more value. In addition, as l increases the points are aligned close to the $y = x$, which means that the imputed results get closer to the exact values.

Note that in our numerical tests, the rate of decay for the l^{th} left and the right kernels was set to μ^l where $\mu = 2$. It is possible to choose different decay rates for the left and the right kernels, depending on the geometry of the data. For example, in Table 1, in case of 80% and 50% missing data, setting the decay rates to $\mu^{(L)} = 1.2$ and $\mu^{(R)} = 4$ resulted in a MSE errors of 0.4043 and 0.2136, which improves the results. However, the choice of these parameters depends both on the data structure and on the amount of missing values.

Another example is the voice rehabilitation dataset, also from the UCI repository (http://archive.ics.uci.edu/ml/datasets). The dataset is of size 126 × 309, where 126 is the number of patients and 309 is the number of features. The features were computed by applying various speech signal processing algorithms such as wavelets,

Table 3 Errors for the voice dataset with $C^{(L)} = 2$, $C^{(R)} = 2$

% missing	1D Pyds	2D Pyds	Mean	Freq.
20% (MSE)	0.5843	0.5793	0.9954	10.9006
20% (RMSE)	0.7644	0.7611	0.9977	3.3016
50% (MSE)	0.7047	0.6788	1.0206	8.2944
50% (RMSE)	0.8394	0.8239	1.0103	2.8800
80% (MSE)	0.8408	0.8103	1.0433	5.0751
80% (RMSE)	0.9170	0.9002	1.0214	2.2528

frequency-based and non-linear time-series algorithms. It is plausible that there is some correlation between rows of this dataset as well as between its columns. Three experiments were performed, with 20%, 50%, and 80% of randomly selected missing data entries. The data was first normalized to have mean equal to zero and standard deviation equal to one in each column.

Tables 3 and 4 contain the mean square errors for the three setting (20%, 50%, and 80% missing data) after running Alg. 1 for 10 times in each case and averaging the results. It can be seen the proposed approach performs much better than the mean imputation approach even for the case of 80% missing data. Moreover, it improves the results of the one-dimensional case for both of the examined scale settings. Note that in this dataset, it has a relatively large number of columns (309); hence, the two-directional construction contributes to process by taking into account the relationships between the columns.

Figure 12 illustrates the multi-scale imputation process on a subset of points that were stretched for this figure to be a one-dimensional vector. The three pictures on the top of this figure show the approximation process on 1000 points x_{ij} for which $b_{ij} = 1$, their value is known. The known values are colored in black and the approximated values are in blue. The known points are used to determine the stopping scale l, which is $l = 6$ for this example $l = 6$; the approximations for levels $l = 4, 5, 6$ are plotted. The three figures on bottom of this figure shows the imputation construction for 1000 points with missing values. The correct values are colored in black and the imputed values are colored yellow.

Table 4 Errors for the voice dataset with $C^{(L)} = 2$, $C^{(R)} = 0.5$

% missing	1D Pyds	2D Pyds	Mean	Freq.
20% (MSE)	0.5761	0.5755	0.9904	10.6954
20% (RMSE)	0.7590	0.7586	0.9952	3.2704
50% (MSE)	0.6858	0.6736	1.0172	8.7580
50% (RMSE)	0.8281	0.8207	1.0086	2.9594
80% (MSE)	0.8432	0.8269	1.0415	4.8659
80% (RMSE)	0.9183	0.9002	1.0205	2.2059

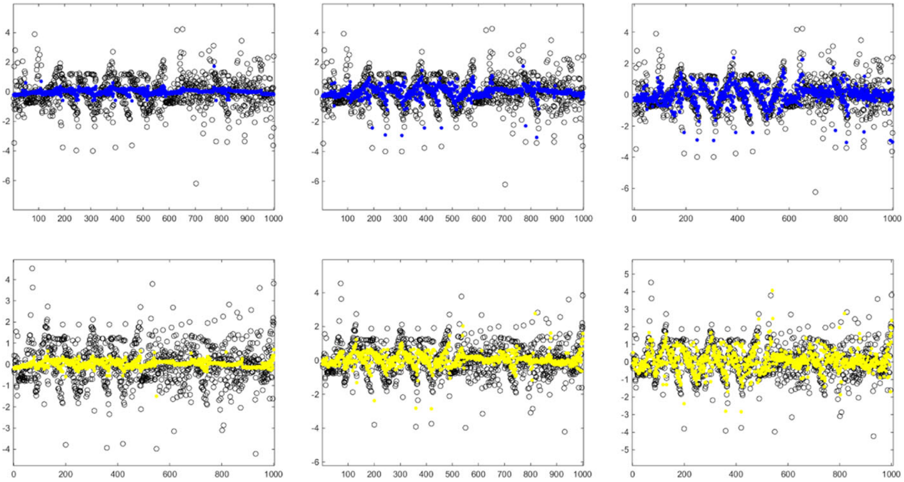


Fig. 12 Multi-scale approximation in blue and imputation in yellow for scales $l = 4, 5, 6$ of the voice dataset

The proposed method is also applied to the following three datasets: Ecoli [18], White wine [8] and Boston housing [16] from the UCI repository and the results are compared with those that were presented in [27]. Categorical variables were removed from the datasets and they were all standardized to have columns with mean equal to 0 and standard deviation equal to 1. The Ecoli data is of size 336×7, 5 of the columns include continuous numeric values and 2 columns include binary values. The wine dataset is of size 4898×11. The columns hold attributes such as acidity, sugar, and pH, which were computed based on physicochemical tests. The size of the housing dataset (without two categorical attributes named CHAS and RAD) is 506×11. Figure 13 plots the multi-scale kernels for the housing dataset example. In order to enhance the column kernel visualization, the attributes were arranged in the

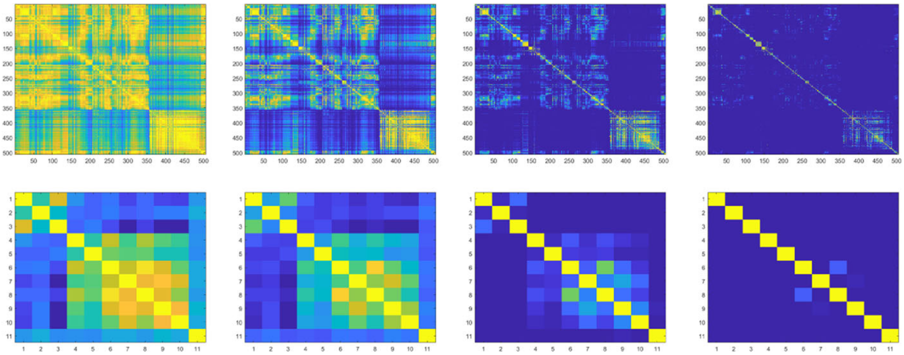


Fig. 13 Multi-scale kernels for the house dataset example. In the first row, $K_0^{(L)}$, $K_1^{(L)}$, $K_2^{(L)}$, and $K_3^{(L)}$ are plotted from left to right, respectively. The second row contains $K_0^{(R)}$, $K_1^{(R)}$, $K_2^{(R)}$, and $K_3^{(R)}$

Table 5 MSE Imputation errors for the three public dataset

Dataset	2D Pyds	1D Pyds	Mean	IRMI	OLI	MICE
oli	0.62	0.66	0.96	8.26	5.75	1.2
Wine	0.43	0.46	0.94	–	0.87	1.1
Housing	0.32	0.34	0.98	0.28	0.30	0.56

following order: ZN, AGE, TAX, CRIM, LSTAT, PTRATIO, INDUS, B, RM, DIS, and NOX, so that correlated attributes are located next to each other.

The last three columns in Table 5 show the errors for the iterative stepwise regression imputation method [33], denoted by IRMI, Optimized Linear Imputation [27], denoted by OLI and Multiple imputation technique [3], which is denoted by MICE. The percentage of the missing data in these examples is 5%. Ten experiments were ran and the error results were averaged. The IRMI method did not converge for the white wine dataset, the error results are left empty for this example. It can be seen that the proposed two-directions Laplacian pyramids approach significantly improve the mean value technique and also, in two cases outperforms the results of other regression-based methods. Additionally, in the proposed method, there is no convergence risk.

Combining the two-directional processing, which acts as local regressions in the row and in the column space, together with the multi-scale construction that allows these regressions to be performed at several *frequency passes* of the data results in stable low error rates even when the percentage of missing data is large.

6 Conclusion

In this paper, we presented a multi-scale approach for modeling a dataset with respect to its dual geometry structures in different scales. The application to data imputation is immediate and the missing data is completed in one pass, together with the model construction. The general representation extended and applied to other learning tasks such learning functions over datasets while considering the two-directional geometric structures.

Acknowledgments The authors would like to thank the reviewers for their professional and thorough review and comments, which significantly improved the paper.

References

1. Asif, M.T., Mitrovic, N., Garg, L., Dauwels, J., Jaillet, P.: Low-dimensional models for missing data imputation in road networks, In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3527–3531 (2013)
2. Bermanis, A., Averbuch, A., Coifman, R.R.: Multiscale data sampling and function extension. Appl. Comput. Harmon. Anal. **34**(1), 15–29 (2013)

3. Buuren, S., Groothuis-Oudshoorn, K.: Mice: Multivariate imputation by chained equations in r. J. Stat. Softw. **45**(3), 1–67 (2011)
4. Candes, E.J., Tao, T.: The power of convex relaxation: near-optimal matrix completion. IEEE Trans. Inf. Theory **56**(5), 2053–2080 (2010)
5. Carreira-Perpin, M.A., Zhengdong, L.: Manifold learning and missing data recovery through unsupervised regression. In: (ICDM) 2011 IEEE 11th International Conference on Data Mining, pp. 1014–1019 (2011)
6. Coifman, R.R., Lafon, S.: Diffusion maps. Appl. Comput. Harmon. Anal. **21**, 5–30 (2006)
7. Coifman, R.R., Gavish, M.: Harmonic analysis of digital data bases. In: Wavelets and Multiscale analysis, pp. 161–197. Birkhäuser, Boston (2011)
8. Cortez, P., Cerdeira, A., Almeida, F., Matos, T., Reis, J.: Modeling wine preferences by data mining from physicochemical properties. Decis. Support. Syst. **47**(4), 547–553 (2009)
9. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B **39**, 1–38 (1977)
10. van Dijk, D., Nainys, J., Sharma, R., Kathail, P., Carr, A.J., Moon, K.R., Mazutis, L., Wolf, G., Krishnaswamy, S., Pe'er, D.: MAGIC, a diffusion-based imputation method reveals gene-gene interactions in single-cell RNA-sequencing data, Preprint (bioRxiv.org). https://doi.org/10.1101/111591 (2017)
11. Dsilva, C.J., Talmon, R., Rabin, N., Coifman, R.R., Kevrekidis, I.G.: Nonlinear intrinsic variables and state reconstruction in multiscale simulations. J. Chem. Phys. **139**(18), 11B608-1 (2014)
12. Fernández, Á., Rabin, N., Fishelov, D., Dorronsoro, J.R.: Auto-adaptative laplacian pyramids for high-dimensional data analysis, arXiv:1311.6594 (2014)
13. Fernández, Á., Rabin, N., Fishelov, D., Dorronsoro, J.R.: Auto-adaptive Laplacian Pyramids, In: 24th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN, Bruges, Belgium, pp. 59–64 (2016)
14. Fishelov, D.: A new vortex scheme for viscous flows. J. Comput. Physics **86**(1), 211–224 (1986)
15. Gilbert, A.C., Sonthalia, R.: Unrolling Swiss Cheese: metric repair on manifolds with holes, arXiv:1807.07610 (2018)
16. Harrison, D., Rubinfeld, D.L.: Hedonic housing prices and the demand for clean air. J. Environ. Econ. Manag. **5**(1), 81–102 (1978)
17. Huisman, M.: Missing data in behavioral science research: investigation of a collection of data sets. Kwantitatieve Methoden **57**, 69–93 (1998)
18. Horton, P., Nakai, K.: A probabilistic classification system for predicting the cellular localization sites of proteins. Ismb **4**, 109–115 (1996)
19. Lindenbaum, O., Bregman, Y., Rabin, N., Averbuch, A.: Multiview kernels for low-dimensional modeling of seismic events. IEEE Trans. Geosci. Remote Sens. **56**(6), 3300–3310 (2018)
20. Lee, J., Kim, S., Lebanon, G., Singer, Y., Bengio, S.: Local low-rank matrix approximation. J. Mach. Learn. Res. **17**(15), 1–24 (2016)
21. Linderman, G.C., Zhao, J., Kluger, Y.: Zero-preserving imputation of scRNA-seq data using low-rank approximation, bioRxiv:397588 (2018)
22. Little, J.A.R., Rubin, B.D. ”Statistical Analysis with Missing Data, 2nd edn. Wiley, New York (2002)
23. Mazumder, R., Hastie, T., Tibshirani, R.: Spectral regularization algorithms for learning large incomplete matrices. J. Mach. Learn. Res. **11**, 2287–2322 (2010)
24. Mishne, G., Chi, E.C., Coifman, R.R.: Co-manifold learning with missing data, arXiv:1810.06803 (2018)
25. Ongie, G., Balzano, L., Pimentel-Alarcón, D., Willett, R., Nowak, R.D.: Tensor Methods for Nonlinear Matrix Completion, arXiv:1804.10266 (2018)
26. Pierson, E., Yau, C.: ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis. Genome Biol. **16**, 241 (2015)
27. Resheff, Y.S., Weinshall, D.: Optimized Linear Imputation, Proceedings: 6Th International Conference on Pattern Recognition Application and Methods (ICPRAM), Porto Portugal (2017)
28. Rabin, N., Coifman, R.R.: Heterogeneous datasets representation and learning using diffusion maps and Laplacian pyramids, In: Proceedings of the 2012 SIAM International Conference on Data Mining, pp. 189–199 (2012)
29. Rabin, N., Fishelov, D.: Missing data completion using diffusion maps and laplacian pyramids. International Conference on Computational Science and Its Applications, pp. 284–297 (2017)
30. Scholz, M., Kaplan, M.F., Guy, C.L.., Kopka, J., Selbig, J.: Non-linear PCA: a missing data approach. Bioinformatics **21**(20), 3887–3895 (2005)

31. Shabat, G., Shmueli, Y., Averbuch, A.: Missing entries matrix approximation and completion. Proceedings of the 10th International Conference on Sampling Theory and Applications (SampTA), pp. 440–443 (2013)
32. Shahid, N., Perraudin, N., Kalofolias, V., Vandergheynst, P.: Fast robust PCA on graphs. IEEE J. Sel. Top. Signal Process. **10**(4), 740–756 (2016)
33. Templ, M., Kowarik, A., Filzmoser, P.: Iterative stepwise regression imputation using standard and robust methods. Comput. Stat. Data Anal. **55**(10), 2793–2806 (2011)
34. UshaRani, Y., Sammulal, P.: An Efficient Disease Prediction and Classification Using Feature Reduction Based Imputation Technique. In: International Conference on Engineering & MIS (ICEMIS) (2016)