Using Lowly Correlated Time Series to Recover Missing Values in Time Series: a Comparison between SVD and CD

Mourad Khayati[1,2], Michael H. Böhlen[1] and Philippe Cudré Mauroux[2]

[1] Department of Computer Science, University of Zurich, Switzerland
{mkhayati,boehlen}@ifi.uzh.ch
[2] eXascale Infolab, University of Fribourg, Switzerland
{mkhayati,phil}@exascale.info

Abstract. The Singular Value Decomposition (SVD) is a matrix decomposition technique that has been successfully applied for the recovery of blocks of missing values in time series. In order to perform an accurate block recovery, SVD requires the use of highly correlated time series. However, using lowly correlated time series that exhibit shape and/or trend similarities could increase the recovery accuracy. Thus, the latter time series could also be exploited by including them in the recovery process.

In this paper, we compare the accuracy of the Centroid Decomposition (CD) against SVD for the recovery of blocks of missing values using highly and lowly correlated time series. We show that the CD technique better exploits the trend and shape similarity to lowly correlated time series and yields a better recovery accuracy. We run experiments on real world hydrological and synthetic time series to validate our results.

1 Introduction

In real world applications sensors are used to measure time series data of different types, which are then collected, processed and stored in central stations. In the hydrological field, for instance, weather stations collect measurements that describe meteorological phenomena, e.g., temperature, humidity, air pressure, precipitation, etc. These time series contain blocks of missing values due to many reasons, e.g., sensor failure, power outage, sensor to central server transmission problem, etc. In order to recover these missing values, existing recovery techniques use the (base) time series that contains the missing values in addition to highly correlated (reference) time series. However, these recovery techniques can not learn from the trend and shape similarity of lowly correlated reference time series. Thus, the latter are not included in the recovery process.

The Foehn, for instance, is a warm wind that reaches weather stations at different time points. This environmental phenomenon yields time series with shape and trend similarities, but shifted in time. For example, the Foehn yields shifted temperature time series with similar shapes, e.g., peaks that contain similar spikes. These shifted time series are lowly correlated. It is of interest to benefit from Foehn based time series and include them, in addition to the highly correlated time series, in the recovery process. In this paper, we consider the category of lowly correlated reference time series, e.g., Fohen based time series, that exhibit shape and/or trend similarities to the base time series.

Matrix decomposition techniques decompose an input matrix into the product of k matrices where $k \in [2, 3]$. The truncated Singular Value Decomposition (SVD) has been successfully applied to recover missing values in time series [1]. The truncated SVD performs a decorrelation of vectors and subsequently an unweighted relative reduction of the Mean Squared Error (MSE) to the reference time series. The unweighted MSE reduction yields a recovery that ignores the correlation difference between the input time series. Thus, this recovery technique is not suitable to apply in case of using highly and lowly correlated reference time series (cf. Section 5). To the best of our knowledge, there does not exist any technique that introduces different weights in the decomposition process of SVD. In [3–5] fast approximations of the truncated SVD have been proposed. Similarly to SVD, the latter approximations perform a decorrelation of vectors and thus, produce an unweighted MSE relative reduction.

In this work, we are interested in the case of using highly and lowly correlated time series for the recovery of blocks of missing values. Intuitively, in such cases, an accurate recovery technique should give different weights to the used time series. In contrast of the truncated SVD, the truncated Centroid Decomposition (CD) technique gives a weight proportional to the correlation between the base and the reference time series (cf. Section 5). Consequently, the obtained recovery produces a relative reduction of the MSE to the highly correlated reference time series more than to the lowly correlated one yielding a block recovery better than the one produced by the truncated SVD. We assume that the lowly correlated time series that exhibit trend and/or shape similarity are given as input. Searching for these time series is beyond the scope of this paper.

The main contributions of this paper are:

- We prove that CD technique produces correlated output vectors while SVD technique produces uncorrelated output vectors.
- We empirically show that CD performs a weighted MSE relative reduction that is proportional to the correlation of the input time series. The resulting recovery of missing values uses the correlation difference between the input time series.
- We empirically show that SVD performs an unweighted MSE relative reduction. The resulting recovery of missing values ignores the correlation difference between the input time series.
- We present the results of an experimental evaluation of the recovery accuracy of the CD and SVD techniques. The iterated truncated CD produces a better recovery accuracy in case of using a similar number of highly and lowly correlated time series.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 describes the recovery process using SVD and CD techniques. Section 4 defines the unweighted recovery and the correlation based recovery respectively performed by SVD and CD. Section 5 reports the evaluation results. Section 6 concludes the paper and points to future work.

2 Related Work

The Singular Value Decomposition (SVD) is a commonly used matrix decomposition technique. It computes the singular values with their corresponding right and left singular vectors. The truncated SVD, which is computed out of SVD by nullifying the smallest singular values, has been extensively used in many fields, e.g., compression, noise reduction, etc. Khayati et al. [1] applied the truncated SVD for the recovery of missing values in time series. The basic idea is as follows: the truncated SVD is iteratively applied to a matrix that has as columns the time series for which the missing values have been initialized through linear interpolation. The iterative process refines only the initialized missing values and terminates when the difference between the updated values before and after the refinement is smaller than a small threshold value, e.g., 10^{-5}. The Mean Squared Error (MSE), between the real values and the recovered ones, is used to evaluate the recovery accuracy [2].

The Centroid Decomposition (CD) is a matrix decomposition technique that decomposes an input matrix into the product of two matrices. Chu et al. [6] introduce an algorithm that computes the CD of an input matrix in quadratic run time, but requires the construction of a correlation square matrix that has a quadratic space complexity. Khayati et al. [7] propose an algorithm to compute the CD out of the input matrix using a weight vector instead of the construction of the correlation matrix. They prove the correctness of the proposed solution. The space complexity is thus reduced from quadratic to linear while keeping the same run time complexity.

The Semi Discrete Decomposition (SDD) [8] is a matrix decomposition technique that decomposes an input matrix into three matrices such that their product approximates the input matrix, i.e., $\mathbf{X} \approx \mathbf{X'} \cdot \mathbf{D} \cdot \mathbf{Y}^T$. The resulting $\mathbf{D}$ is a diagonal matrix and the values of $\mathbf{X'}$ and $\mathbf{Y}$ are restricted to belong to the set $\{-1, 0, 1\}$. The truncated SDD has been used as clustering method [9]. The non-zero elements of the matrix obtained from the product $d_{ii} \times X'_{*i} \cdot Y^T_{*i}$ are the elements of the input matrix $\mathbf{X}$ which have the closest values and thus can be clustered together. Due to the set restriction of the elements of $\mathbf{X'}$ and $\mathbf{Y}$, the application of SDD for the recovery of blocks of missing values does not produce accurate results.

In addition to matrix decomposition techniques, matrix factorization techniques have been also applied for the recovery of missing values. The latter techniques start from k random matrices in order to approximate the input matrix. Stochastic Gradient Descent (SGD) [10] is a matrix factorization technique that approximates an input matrix $\mathbf{X}$ by the product of two matrices $\mathbf{P}$ and $\mathbf{Q}$, i.e., $\mathbf{X} \approx \mathbf{P} \cdot \mathbf{Q}$. SGD iteratively minimizes an error function by computing the gradient. At each iteration, the gradient is computed using random sample square blocks of the input matrix. The accuracy of the gradient increases with the size and the number of the used blocks [11]. Thus, using an input matrix with high number of rows and columns yields an accurate gradient's computation and subsequently a good approximation of the input matrix. In [12], SGD has been successfully applied to predict ratings in recommender systems for a matrix of items as rows and users as columns. Balzano et al [13] propose an SGD-based solution, called GROUSE, for the recovery of blocks of missing values in an input matrix. GROUSE performs an accurate recovery for matrices of a high number of rows and columns. The recovery accuracy of the proposed solution deteriorates if the number of columns is much smaller than the number of rows such as in the time series field where the number of time series is much smaller than the number of observations.

3 Preliminaries

3.1 Notation

Bold upper-case letters refer to matrices, regular font upper-case letters to vectors (rows and columns of matrices) and lower-case letters to elements of vectors/matrices. For example, $\mathbf{X}$ is a matrix, $\mathbf{X}^T$ is the transpose of $\mathbf{X}$, X_{i*} is the i-th row of $\mathbf{X}$, X_{*i} is the i-th column of $\mathbf{X}$ and x_{ij} is the j-th element of X_{i*}.

In multiplication operations we use the sign $\times$ for scalar multiplication and the sign $\cdot$ otherwise. The symbol $\|\|$ refers to the l-2 norm of a vector. Assume $X = [x_1, \dots, x_n]$, then $\|X\| = \sqrt{\sum_i^n (x_i)^2}$.

3.2 Background

Time Series A time series $X_{i*} = \{(t_1, v_1), (t_2, v_2), \dots, (t_n, v_n)\}$ is a set of n temporal values v_i ordered with respect to their timestamps t_i. We consider time series that have the same granularity of values. Thus, we omit the timestamps and we write time series using only their ordered values, e.g., time series $X_{1*} = \{(1,4),(2,5),(3,1)\}$ is written as $X_{1*} = \{4,5,1\}$. Time series are inserted as columns of the input matrix $\mathbf{X}$.

Pearson Correlation Coefficient Given two vectors X and Y of equal length n, with respective averages $\bar{x}$ and $\bar{y}$, the Pearson correlation coefficient is defined as,

$$r(X,Y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2}\sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (1)$$

The absolute value of r ranges between 0 and 1 where $r \in [0.7, 1]$ stands for highly correlated vectors. The value of r is undefined if all x_i (and/or y_i) are equal.

Initialization Strategy The missing values of each time series are initialized as a preprocessing step before the application of the recovery process. A missing value is initialized with a linear interpolation between the predecessor and the successor values. If the missing value occurs as the first or the last elements of the time series, we use the nearest neighbor initialization. Thus, the missing values of a time series X_{*1} are initialized as follows:

$$(t_i, v_i) = \begin{cases} (t_i, v) \textbf{ if } (s(t_i), \_) \notin X_{*1}, \\ \qquad\qquad (p(t_i), v) \in X_{*1} \\ (t_i, v) \textbf{ if } (p(t_i), \_) \notin X_{*1}, \\ \qquad\qquad (s(t_i), v) \in X_{*1} \\ (t_i, \frac{(t_i - p(t_i))(s(v_i) - p(v_i))}{s(t_i) - p(t_i)} + s(v_i)) \\ \qquad\qquad \textbf{otherwise} \end{cases}$$

where $p(t_i) = max\{t_j \mid (t_j, \_) \in X_{*1} \wedge t_j < t_i\}$ is the predecessor of timestamp t_i in X_{*1} and $s(t_i) = min\{t_j \mid (t_j, \_) \in X_{*1} \wedge t_j > t_i\}$ is the successor timestamp of t_i in X_{*1}. Similarly, $p(v_i) = \{v_j \mid (t_j, \_) \in X_{*1} \wedge t_j = p(t_i)\}$ is the predecessor of value v_i in X_{*1} and $s(v_i) = \{t_j \mid (t_j, \_) \in X_{*1} \wedge t_j = s(t_i)\}$ is the successor value of v_i in X_{*1}.

3.3 Matrix Decomposition

Singular Value Decomposition The *Singular Value Decomposition (SVD)* is a matrix decomposition technique that decomposes an $n \times m$ matrix, $\mathbf{X} = [X_{*1}| \ldots |X_{*m}]$, into an $n \times p$ matrix, $\mathbf{U}$, a $p \times m$ matrix, $\mathbf{\Sigma}$, and an $m \times m$ matrix $\mathbf{V}$, i.e.,

$$\begin{aligned} \mathbf{X} &= \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^T \\ &= \sum_{i=1}^{p} \sigma_i \times U_{*i} \cdot (V_{*i})^T, \end{aligned} \tag{2}$$

where $p = \min(n, m)$, the columns of $\mathbf{U}$ and $\mathbf{V}$ are respectively called left and right singular vectors, and $\mathbf{\Sigma}$ is a matrix whose diagonal elements, σ_i, are called singular values and are arranged in decreasing order, i.e., $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$. The obtained columns of $\mathbf{U}$ are orthogonal to each other, i.e., $U_{*1} \perp U_{*2} \perp \ldots \perp U_{*p}$. Similarly, the columns of $\mathbf{V}^T$ are orthogonal to each other. In order to guarantee the orthogonality of columns of respectively $\mathbf{U}$ and $\mathbf{V}$, SVD requires the use of the same input matrix [14]. Since SVD decomposition is performed based on the same input matrix $\mathbf{X}$, we refer to SVD as a *flat* decomposition method.

Fig 1 illustrates the SVD decomposition of an input matrix $\mathbf{X}$.

$$\mathbf{X} = \begin{bmatrix} \text{-4} & 0 \\ 2 & 1 \\ 3 & \text{-2} \end{bmatrix};\ \text{SVD}(\mathbf{X}) = \underbrace{\begin{bmatrix} \text{-0.725} & \text{-0.307} \\ 0.333 & 0.627 \\ 0.603 & \text{-0.716} \end{bmatrix}}_{\mathbf{U}}, \underbrace{\begin{bmatrix} 5.445 & 0 \\ 0 & 2.086 \end{bmatrix}}_{\mathbf{\Sigma}}, \underbrace{\begin{bmatrix} 0.987 & 0.16 \\ \text{-0.16} & 0.987 \end{bmatrix}}_{\mathbf{V}}$$

such that

$$\mathbf{X} = \underbrace{\begin{bmatrix} \text{-0.725} & \text{-0.307} \\ 0.333 & 0.627 \\ 0.603 & \text{-0.716} \end{bmatrix}}_{\mathbf{U}} \times \underbrace{\begin{bmatrix} 5.445 & 0 \\ 0 & 2.086 \end{bmatrix}}_{\mathbf{\Sigma}} \times \underbrace{\begin{bmatrix} 0.987 & \text{-0.16} \\ 0.16 & 0.987 \end{bmatrix}}_{\mathbf{V}^T}$$

Fig. 1. Example of Singular Value Decomposition.

Centroid Decomposition The *Centroid Decomposition (CD)* is a matrix decomposition technique that decomposes an $n \times m$ matrix, $\mathbf{X} = [X_{*1}| \dots |X_{*m}]$, into an $n \times m$ loading matrix, $\mathbf{L}$, and an $m \times m$ relevance matrix, $\mathbf{R}$, i.e.,

$$\mathbf{X} = \mathbf{L} \cdot \mathbf{R}^T = \sum_{i=1}^{m} L_{*i} \cdot (R_{*i})^T, \tag{3}$$

where $\|L_{*1}\| > \|L_{*2}\| > \dots > \|L_{*m}\| \geq 0$. Fig. 2 illustrates the CD of matrix $\mathbf{X}$.

$$\mathbf{X} = \begin{bmatrix} \text{-4} & 0 \\ 2 & 1 \\ 3 & \text{-2} \end{bmatrix};\ \text{CD}(\mathbf{X}) = \underbrace{\begin{bmatrix} \text{-3.977} & -0.43 \\ 1.878 & 1.214 \\ 3.202 & \text{-1.658} \end{bmatrix}}_{\mathbf{L}}, \underbrace{\begin{bmatrix} 0.994 & 0.11 \\ \text{-0.11} & 0.994 \end{bmatrix}}_{\mathbf{R}}$$

such that

$$\mathbf{X} = \begin{bmatrix} \text{-4} & 0 \\ 2 & 1 \\ 3 & \text{-2} \end{bmatrix} = \underbrace{\begin{bmatrix} \text{-3.977} & -0.43 \\ 1.878 & 1.214 \\ 3.202 & \text{-1.658} \end{bmatrix}}_{\mathbf{L}} \times \underbrace{\begin{bmatrix} 0.994 & \text{-0.11} \\ 0.11 & 0.994 \end{bmatrix}}_{\mathbf{R}^T}$$

Fig. 2. Example of Centroid Decomposition.

The CD technique applies an iterative process to compute matrices $\mathbf{L}$ and $\mathbf{R}$. At each iteration i, the input matrix $\mathbf{X}$ is updated by subtracting the product $L_{*i} \cdot R^T_{*i}$ from it. The columns of $\mathbf{L}$ (and $\mathbf{R}$) are not orthogonal to each other. Since CD decomposition is performed by hierarchically updating $\mathbf{X}$, we refer to CD as a *hierarchical* decomposition method.

Chu et al. [6] prove that the decomposition performed by CD best approximates the one produced by SVD, i.e., $\mathbf{L}$ approximates the product $\mathbf{U} \cdot \mathbf{\Sigma}$ and $\mathbf{R}$ approximates $\mathbf{V}$.

Truncation The *truncated SVD* computes a matrix $\mathbf{X}_k$ out of the SVD of $\mathbf{X}$. It takes only the k first columns of $\mathbf{U}$ and $\mathbf{V}$ and the k largest elements of $\mathbf{\Sigma}$ such that $k < p$, i.e.,

$$\mathbf{X}_k = \sum_{i=1}^{k} \sigma_i \times U_{*i} \cdot (V_{*i})^T. \tag{4}$$

Eq. (4) is equivalent to $\mathbf{X}_k = \mathbf{U} \cdot \mathbf{\Sigma}_k \cdot \mathbf{V}^T$ where $\mathbf{\Sigma}_k$ is obtained by setting the $r - k$ smallest (non zero) singular values of $\mathbf{\Sigma}$ to 0. Let rank p be the maximal number of linearly independent rows or columns of $\mathbf{X}$. Then, among all matrices with rank $k < p$, $\mathbf{X}_k$ is proven to be the optimal approximation to the input matrix $\mathbf{X}$ in the Frobenius norm [15].

The *truncated CD* computes a matrix $\mathbf{X}_k$ out of the CD of $\mathbf{X}$ by setting to 0 the $m - k$ (non zero) last columns of $\mathbf{L}$, with $k < m$, in order to respectively get $\mathbf{L}_k$ and $\mathbf{X}_k = \mathbf{L}_k \cdot \mathbf{R}^T$.

4 Decomposition Comparison

In this section, we compare the decomposition produced by the truncated SVD against the one produced by the truncated CD using the Mean Squared Error ($MSE = \frac{1}{k}\sum_{i=1}^{k}(\tilde{x}_i - x_i)^2$; initialized value x_i; recovered value $\tilde{x}_i$; number of observations k) between the initialized values and the recovered ones.

4.1 Recovery Process

Algorithm 1 describes the pseudo code of function *RecM()* that applies truncated SVD and truncated CD to recover missing values. The algorithm takes an input matrix $\mathbf{X}$ where the missing values have been initialized, and returns a matrix with recovered values $\widetilde{\mathbf{X}}$. Different initialization techniques would lead to the same result but with a higher number of iteration [7]. *RecM()* iteratively replaces the initialized missing values by the result of the truncation of a given matrix decomposition technique. The algorithm terminates if the difference in Frobenius norm ($\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m}(x_{ij} - \tilde{x}_{ij})^2}$; x_{ij}: element of $\mathbf{X}$; $\tilde{x}_{ij}$: element of $\widetilde{\mathbf{X}}$) between the matrix before the update of missing values, $\mathbf{X}$, and the one after, $\widetilde{\mathbf{X}}$, is less than a small threshold value, e.g., $\epsilon = 10^{-5}$.

In what follows we describe the recovery properties using respectively SVD and CD. We assume the case where the correlation ranking between time series does not change over the entire history, i.e., the most correlated reference time series has the highest correlation value to the base time series all over the entire history. In case where the correlation ranking changes over the history, then a segmentation of the time series has to be applied.

Algorithm 1: RecM($\mathbf{X}$, n, m, T_j^m)

Input: $n \times m$ matrix $\mathbf{X}$; set of missing time stamps T_j^m in X_{*j}
Output: $n \times m$ matrix $\widetilde{\mathbf{X}}$ of recovered values

repeat
  $\widetilde{\mathbf{X}} = \mathbf{X}$;
  `// Apply truncated SVD or truncated CD`
  $\mathbf{X}_k = Truncate(\widetilde{\mathbf{X}})$;
  `// Update missing values`
  foreach $t \in T_j^m$ **do**
    $x_{tj} = w_{tj}$;
    `//` w_{tj} `element of` $\mathbf{X}_k$
until $\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F < \epsilon$;
return $\widetilde{\mathbf{X}}$

4.2 SVD recovery

Lemma 1 *Given an input matrix* $\mathbf{X}$ *of* m *correlated columns. SVD(*$\mathbf{X}$*) produces non correlated vectors.*

Proof 1 *By definition of SVD, we have that* $U_{*1} \perp U_{*2} \perp \ldots \perp U_{*p}$. *This implies that the pairwise dot product of columns of* $\mathbf{U}$ *is equal to 0 and thus,* $\forall a, b \in [1, p] \wedge a \neq b$, *we get* $(U_{*a})^T \cdot U_{*b} = 0$. *Using the fact that all input time series have been normalized to have mean equal to 0 (cf. Section 5), we assume* u *and* u' *to be the* i*-th elements of respectively* U_{*a} *and* U_{*b}, *and get from Equation (1) the following*

$$r(U_{*a}, U_{*b}) = \frac{\sum_{i=1}^{n}(u_i \times u'_i)}{\sqrt{\sum_{i=1}^{n}(u_i - \bar{u})^2}\sqrt{\sum_{i=1}^{n}(u'_i - \bar{u'})^2}}$$

$$= \frac{(U_{*a})^T \cdot U_{*b}}{\sqrt{\sum_{i=1}^{n}(u_i - \bar{u})^2}\sqrt{\sum_{i=1}^{n}(u'_i - \bar{u'})^2}} = 0$$

As a result, the pairwise correlation between all columns of $\mathbf{U}$ *is equal to 0. The previous property holds also for the columns of* $\mathbf{V}^T$.

Definition 1 (Unweighted Recovery) *Let* $\mathbf{X}$ *be an input matrix that contains a base time series* B *and* $k > 2$ *reference time series each with a correlation* r_i *to* B. *An* unweighted recovery *of* B *produces a similar relative reduction of the MSE between* B *and the reference time series.*

Proposition 1 *Assume an* $n \times m$ *matrix* $\mathbf{X} = [B, R_1, \ldots, R_{m-1}]$. *A truncated matrix decomposition of* $\mathbf{X}$ *that produces uncorrelated vectors performs an unweighted recovery of* B.

Based on Lemma 1 and Proposition 1, we get that the truncated SVD performs an unweighted recovery.

Example 1 *Let's take the example of a matrix* $\mathbf{X} = [B, R_1, R_2]$ *where initialized missing values are marked in bold.*

$$\mathbf{X} = \begin{bmatrix} \text{-4} & 1 & 3 \\ \textbf{-1} & 3 & \text{-1} \\ \mathbf{2} & 6 & 6 \\ 5 & 5 & 3 \end{bmatrix}$$

R_1 is a highly correlated reference time series to B with $r(B, R_1)$= 0.88 and R_2 is a lowly correlated reference time series to B with $r(B, R_2) = 0.32$. The computation of the MSE before the recovery gives $MSE(B, R_1) = 16$ and $MSE(B, R_2) = 8$.

The following matrix $\widetilde{\mathbf{X}} = [\widetilde{B}, R_1, R_2]$ is an example of an SVD based recovery of B.

$$\widetilde{\mathbf{X}} = \begin{bmatrix} \text{-4} & 1 & 3 \\ 0 & 3 & \text{-1} \\ 4 & 6 & 6 \\ 5 & 5 & 3 \end{bmatrix}$$

The computation of the MSE after the recovery gives $MSE(\widetilde{B}, R_1) = 6.5$ and $MSE(\widetilde{B}, R_2) = 2.5$. The percentage of the MSE relative reduction between B and R_1 is $red(R_1) = \frac{16-6.5}{16} \times 100 = 60\%$. Similarly, the percentage of the MSE relative reduction between B and R_2 is $red(R_2) = 69\%$. As a result, we have $red(R_1) \approx red(R_2)$.

4.3 CD recovery

Lemma 2 *Given an input matrix $\mathbf{X}$ of m correlated columns. CD($\mathbf{X}$) produces correlated vectors.*

Proof 2 *This proof follows directly from the proof of Lemma 1. On the contrary of SVD, the columns of $\mathbf{L}$ and $\mathbf{R}^T$ computed by the truncated CD are not orthogonal and thus, the pairwise dot product and consequently the pairwise correlation values are different from 0.*

Definition 2 (Correlation Weighted Recovery) *Let $\mathbf{X}$ be an input matrix that contains a base time series B and $k > 2$ reference time series each with a correlation r_i to B. A* correlation weighted recovery *of B performs a relative reduction of the MSE between B and the reference time series proportionally to $|r_i|$.*

Proposition 2 *Assume an $n \times m$ matrix $\mathbf{X} = [B, R_1, \ldots, R_{m-1}]$. A truncated matrix decomposition of $\mathbf{X}$ that produces correlated vectors performs a correlation weighted recovery of B.*

Based on Lemma 2 and Proposition 2 we get that the truncated CD performs a correlation weighted recovery.

Example 2 *Let's take the example of a matrix* $\mathbf{X} = [B, R_1, R_2]$ *used in Example 1. The following matrix* $\widetilde{\mathbf{X}} = [\widetilde{B}, R_1, R_2]$ *is an example of a CD based recovery of* B*.*

$$\widetilde{\mathbf{X}} = \begin{bmatrix} -4 & 1 & 3 \\ 2 & 3 & -1 \\ 5 & 6 & 6 \\ 5 & 5 & 3 \end{bmatrix}$$

The computation of the MSE after the recovery gives $MSE(\widetilde{B}, R_1) = 1$ *and* $MSE(\widetilde{B}, R_2) = 5$*. The percentage of the MSE relative reduction between* B *and* R_1 *is* $red(R_1) = 94\%$*. The percentage of the MSE relative reduction between* B *and* R_2 *is* $red(R_2) = 37.5\%$*. As a result, we have* $red(R_1) \gg red(R_2)$*.*

4.4 Complexity

We compare the runtime and space complexity of CD based recovery against SVD based recovery. We use the algorithm that computes the exact decomposition for each technique.

Run time Consider an input matrix $\mathbf{X}$ with n rows and m columns. The number of arithmetic operations to compute SVD of $\mathbf{X}$, using Golub and Reinsch algorithm [14], is $4n^2m + 8nm^2 + 9m^3$. The number of arithmetic operations to compute CD of $\mathbf{X}$ is $2pnm$ where p is the number of iterations [7]. At each iteration of CD, the input matrix is subtracted yielding an updated matrix that contains negative elements. Thus, the value of p depends on the distribution of the minus sign across the updated matrix. In practice, the value of p ranges between $\frac{n}{2}$ and $\frac{n}{3}$ (cf. Section 5.5).

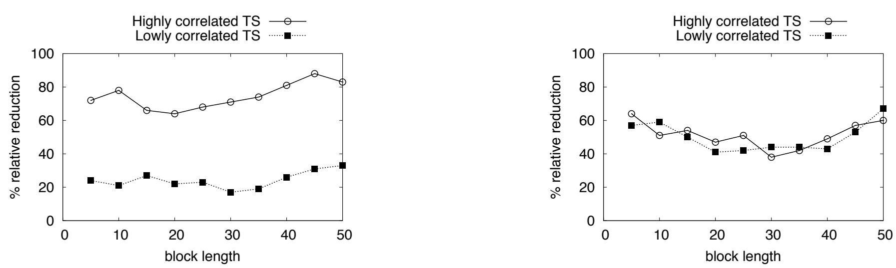
Space SVD technique requires the storage of nm values of $\mathbf{X}$, nm values of $\mathbf{U}$, m values of Σ and m^2 values of $\mathbf{V}$. Additionally, SVD has to transform $\mathbf{X}$ to a bidiagonal matrix using Householder reduction [16] which requires the storage of three additional matrices, i.e., the first matrix contains nm values and the two others contain m^2 values each. The total number values stored by SVD is thus equal to $m(3n + 3m + 1)$ values. CD technique requires the storage of nm values of $\mathbf{X}$, nm values of $\mathbf{L}$ and m^2 values of $\mathbf{R}$. No data structure other than the input and the two output matrices is stored. Thus, the total number values stored by CD is equal to $m(2n + m)$ values.

5 Experiments

The experiments are performed using real world datasets that describe hydrological time series where each tuple records a timestamp and a value of a specific observation. Hydrological time series with shifted peaks and/or valleys are lowly correlated. Our first set of time series, HYD^3, contains 200 time series of six years length each, where measurements are recorded every five minutes. The second set of time series we refer to, SBR^4, contains 120 time series of twelve years length each, where measurements are recorded every 30 minutes. The hydrological time series have been normalized with the z-score normalization technique [17]. We consider hydrological time series where the correlation ranking does not change all over the history. We use also synthetic time series, where the correlation is constant all over the entire history. To measure the recovery accuracy, we compute the Mean Squared Error (MSE) between the original and the recovered blocks (cf. Section 4).

5.1 Recovery using real world TS

MSE relative reduction In this experiment we compute the MSE relative reduction between a base time series B and two reference time series. In Fig. 3 we choose one highly and one lowly correlated reference time series with the respective correlation values $r(B, R_1) = 0.83$ and $r(B, R_2) = 0.18$. The result of this experiment shows that the iterated truncated CD produces a correlation weighted recovery that reduces the relative MSE more to the highly correlated time series than the lowly correlated time series. The iterated truncated SVD performs an unweighted recovery that produces an almost equal reduction of the relative MSE to both reference time series.



(a) Iterated truncated CD. (b) Iterated truncated SVD.

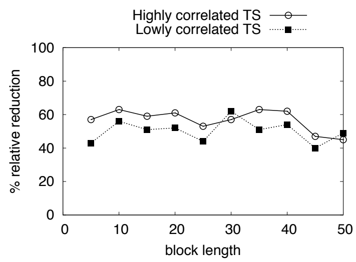
Fig. 3. MSE relative reduction of CD and SVD using highly and lowly correlated time series: case 1.

In Fig. 4 we consider one highly correlated reference time series with a correlation value $r(B, R_1) = 0.76$. We add also a lowly correlated time series with

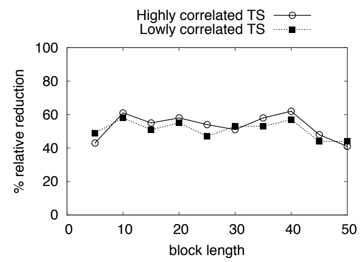
[3] The data was kindly provided by the environmental engineering company HydroloGIS (http://www.hydrologis.edu).

[4] The data was kindly provided by the consultancy organization Südtiroler Beratungsring (http://www.beratungsring.org).

a correlation value $r(B, R_2) = 0.62$ that is higher than the one used in the experiment of Fig. 3. As expected, the MSE relative reduction of the iterated truncated CD is slightly higher to R_1 than to R_2. The MSE relative reduction of the iterated truncated SVD remains similar to both reference time series.



(a) Iterated truncated CD.



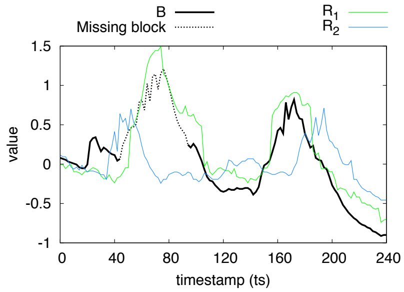
(b) Iterated truncated SVD.

Fig. 4. MSE relative reduction of CD and SVD using highly and lowly correlated time series: case 2.

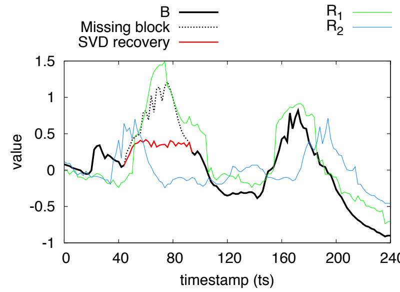
Recovery accuracy In this section we compare the recovery accuracy of the iterated truncated SVD against the iterated truncated CD using highly and lowly correlated time series.

In the experiment of Fig. 5, we use three temperature time series from *HYD* measured respectively in Aria Borgo (B), Ponte Adige (R_1) and Aria La Villa (R_2) in the region of South Tyrol, Italy. B is highly correlated to R_1 with $r(B, R_1) = 0.75$. B is lowly correlated to R_2 with $r(B, R_2) = 0.32$. However, the peaks of B and R_2 exhibit shape similarity, i.e., the peaks contain similar spikes. The time shift is caused by the Foehn phenomenon (cf. Section 1). We drop from the base time series, B, a block for ts $\in [45, 95]$ and recover it using two reference time series, R_1 and R_2. The result of this experiment shows that the iterated truncated CD gives a weight to the reference time series proportional to their correlation with B, yielding a good block recovery accuracy, i.e., the amplitude and the shape of the missing block are accurately recovered. On the contrary, the iterated truncated SVD performs a block recovery that gives the same weight to both time series R_1 and R_2 at a time yielding a bad block recovery accuracy.

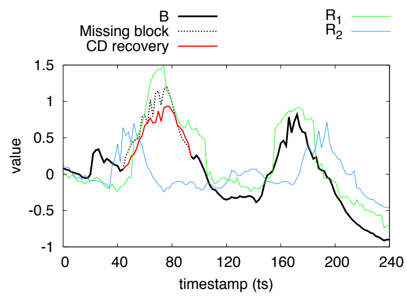
Fig. 6 shows the MSE for removed blocks of values of increasing length from a base time series: starting from the middle of a block we increase the length of the removed block in both directions and we compute the MSE for each block. We run the experiment on five different base time series from *HYD* and we take the average of the MSE. For each run we use, in addition to the base time series, one highly correlated and one lowly correlated time series. As expected, the iterative truncated CD learns from the highly and lowly correlated time series at a time and thus, produces a small recovery error that slightly increases with the length



(a) B with 1 highly and 1 lowly correlated TS.



(b) Recovery of iterated truncated SVD.



(c) Recovery of iterated truncated CD.

Fig. 5. Recovery using highly and lowly correlated hydrological TS.

of the missing block to recover. However, the recovery accuracy of the iterated truncated SVD considerably deteriorates with the length of the missing block to recover.

Impact of the time shift In Fig. 7 we evaluate the impact of a varying time shift, denoted as s, on the recovery accuracy of the iterated truncated CD and the iterated truncated SVD. We show that for a high value of time shift, the two techniques produce similar block recoveries. In Fig. 7(a) we take three time series from SBR measured respectively in Kaltern (B), Kollman (R_1) and Ritten (R_2) in the region of South Tyrol, Italy. The peaks of B and R_2 have a similar shape, but with a time shift. We drop one peak from B, we shift backwards R_2 with a value s and we compute the MSE recovery accuracy. The result of the experiment shows that starting from $s = 30$, the iterated truncated CD is not able anymore to exploit the lowly correlated time series and produces a block recovery similar to the one produced by the iterated truncated SVD.

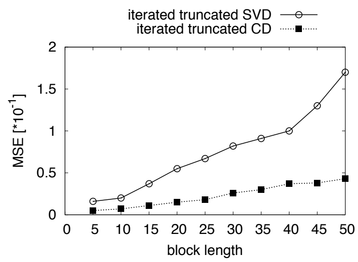
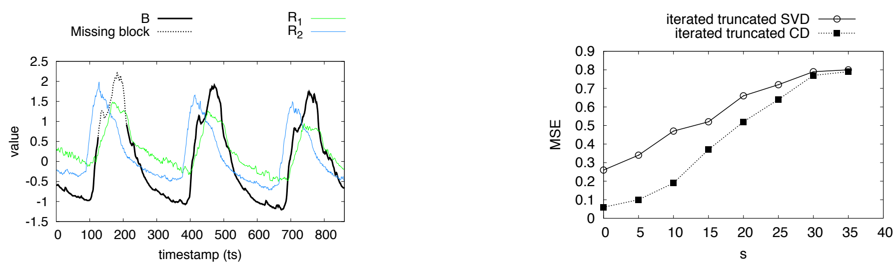


Fig. 6. MSE for successive removed blocks.



(a) Original time series.

(b) MSE with varying time shift.

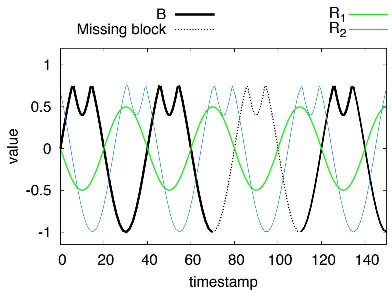
Fig. 7. Impact of varying time shift

5.2 Recovery using synthetic TS

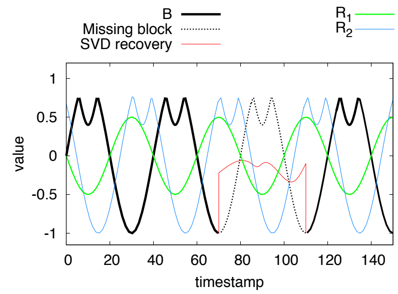
For the following experiments, we consider a time series $sin(t)$ that has a small valley at each of the peaks, denoted as B, from which we drop a block of values for $t \in [70, 110]$ and we recover using both techniques.

Recovery accuracy In Fig. 8 we add to B one highly correlated time series $-0.5 * sin(t)$ denoted as R_1 such that $r(B, R_1) = 0.84$. We add also a lowly correlated time series by shifting B and we denote it as R_2 such that $r(B, R_2) = 0.16$. As expected, by giving a higher weight to R_2, the iterated truncated CD is able to perform a good recovery of the shape and the amplitude of the missing block. The iterated truncated SVD fails to recover the shape and the amplitude of the missing block.

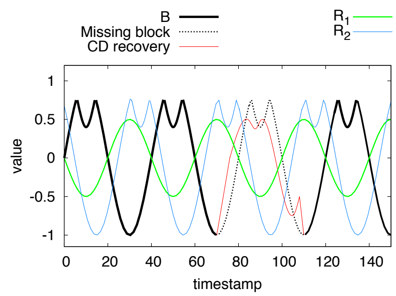
Impact of number of input time series In Fig. 9 we evaluate the robustness of the recovery produced by both techniques using a varying number of highly and lowly correlated time series. In Fig. 9(a) we take B from the experiment of Fig. 8 and one highly correlated time series with $r = 0.9$ to which we add a varying number of lowly correlated time series, by shifting B, such that $r \in [0.2, 0.6]$. The latter time series are added in the decreasing order of their correlation. This experiment shows that for $p_1 < 4$, the iterated truncated CD is able to use the



(a) R with 1 highly and 1 lowly correlated time series



(b) Recovery of iterated truncated SVD.



(c) Recovery of iterated truncated CD.

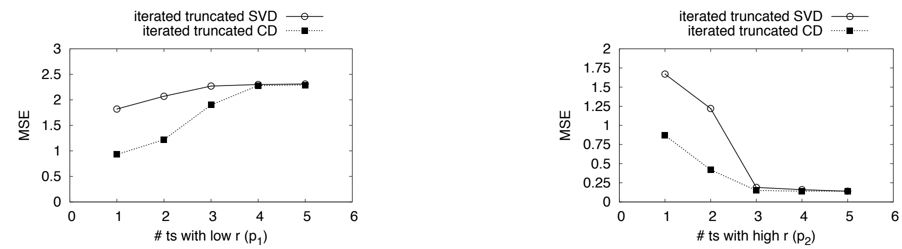
Fig. 8. Recovery using highly and lowly correlated synthetic TS.

most correlated time series yielding a smaller MSE than the iterated truncated SVD. For $p_1 \geq 4$, the MSE of both techniques converges towards similar value. In the experiment of Fig. 9(b) we take B and one lowly correlated time series with $r = 0.2$ to which we add a varying number of highly correlated time series such that $r \in [0.7, 0.9]$. The latter time series are added in the increasing order of their correlation. In the presence of one lowly correlated time series, the iterated truncated SVD requires at least three additional highly correlated time series in order to reach the same MSE as one of the iterated truncated CD.

The experiment of Fig. 9 shows that, for a close number of highly and lowly correlated time series, the correlation weighted recovery helps the iterated truncated CD to produce a better recovery than the one produced by the iterated truncated SVD. Otherwise, the two techniques produce similar recovery of missing values. However, the iterated truncated CD technique is computationally more efficient than the iterated truncated SVD, i.e., CD is linear with the number of input time series while SVD is cubic with the number of input time series.

5.3 Comparison with SGD based recovery

In the experiment of Fig. 10 we compare the accuracy recovery of the iterated truncated CD against GROUSE [12] for the recovery of 20 missing values using



(a) MSE using varying # of lowly correlated time series.

(b) MSE using varying # of highly correlated time series

Fig. 9. Recovery accuracy using varying number of input TS.

an increasing number of segments of time series from the same type where each contains 200 values. We omit the iterated truncated SVD from this experiment because of the high computational time. The result of this experiment shows that the iterated truncated CD produces a more accurate block recovery in the case where the length of the input time series is bigger than their number. However, the recovery accuracy produced by GROUSE outperforms the one produced by the iterated truncated CD as the number of time series approaches the number of observations (cf. Section 2). In real world applications such as hydrology, the length of time series is much bigger than their number and thus, CD based recovery outperforms GROUSE recovery.

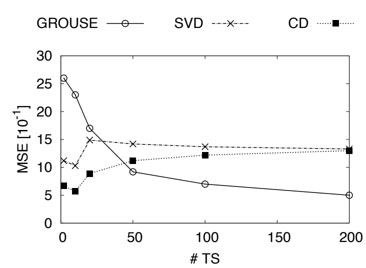


Fig. 10. Recovery accuracy of CD against GROUSE.

5.4 Approximation accuracy

Fig. 11 compares the approximation accuracy of the iterated truncated CD and the iterated truncated SVD to the input matrix. We use the Frobenius norm between the input matrix and the one obtained after the decomposition as an approximation error (cf. Section 4.1). The input matrix contains 10 columns where each one is a time series from *HYD*. This experiment shows that by updating all values of the input matrix at a time (and not only the missing ones), the two techniques perform similar approximation accuracy. The same result holds for different values of the truncation parameter k.

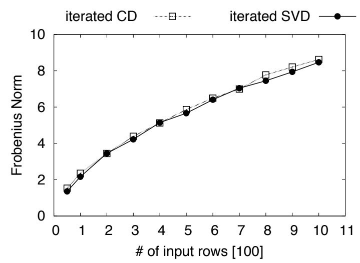


Fig. 11. Approximation error.

5.5 Number of iterations of CD

In the experiment of Fig. 12 we consider three temperature time series from *HYD*: a base time series, one highly correlated reference time series and one lowly correlated time series. We compute the number of iterations p required by the CD technique with an increasing number of rows n. The result of this experiment shows that p ranges between $\frac{n}{2}$ and $\frac{n}{3}$.

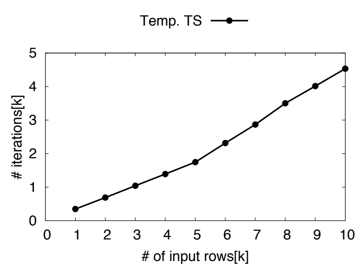


Fig. 12. number of iterations performed by CD.

6 Conclusion

In this paper, we compare the CD and SVD techniques for the recovery of missing values using time series with mixed correlation values. We empirically show that CD produces a weighted relative reduction of MSE that is proportional to the correlation of the input time series, while SVD produces an unweighted relative reduction of MSE. Our experiments on real world hydrological and synthetic time series also show that the iterated truncated CD performs a better recovery in case of similar number of highly and lowly correlated time series.

In future work, it would be of interest to compare the segmentation techniques that are applied in the case where the correlation ranking varies along the time series history. Another promising direction is to refine the definition of highly and lowly correlated time series.

References

1. Khayati, M., Böhlen, M.: Rebom: Recovery of blocks of missing values in time series. In: Proceedings of the 2012 ACM International Conference on Management of Data. COMAD '12, Computer Society of India (2012) 44–55
2. Li, L., McCann, J., Pollard, N.S., Faloutsos, C.: Dynammo: mining and summarization of coevolving sequences with missing values. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009. (2009) 507–516
3. Li, M., Bi, W., Kwok, J.T., Lu, B.: Large-scale nyström kernel matrix approximation using randomized SVD. IEEE Trans. Neural Netw. Learning Syst. **26** (2015) 152–164
4. Halko, N., Martinsson, P.G., Tropp, J.A.: Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev. **53** (2011) 217–288
5. Achlioptas, D., McSherry, F.: Fast computation of low-rank matrix approximations. J. ACM **54** (2007)
6. Chu, M., Funderlic, R.: The centroid decomposition: Relationships between discrete variational decompositions and svds. SIAM J. Matrix Anal. Appl. **23** (2001) 1025–1044
7. Khayati, M., Böhlen, M., Gamper, J.: Memory-efficient centroid decomposition for long time series. In: ICDE. (2014) 100–111
8. Kolda, T.G., O'Leary, D.P.: A semidiscrete matrix decomposition for latent semantic indexing information retrieval. ACM Trans. Inf. Syst. **16** (1998) 322–346
9. Kolda., T.G., O'Leary, D.P.: Algorithm 805: computation and uses of the semidiscrete matrix decomposition. ACM Trans. Math. Softw. **26** (2000) 415–435
10. Yu, H., Hsieh, C., Si, S., Dhillon, I.S.: Scalable coordinate descent approaches to parallel matrix factorization for recommender systems. In: 12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012. (2012) 765–774
11. Gemulla, R., Nijkamp, E., Haas, P.J., Sismanis, Y.: Large-scale matrix factorization with distributed stochastic gradient descent. In: KDD. (2011) 69–77
12. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. IEEE Computer **42** (2009) 30–37
13. Balzano, L., Nowak, R., Recht, B.: Online identification and tracking of subspaces from highly incomplete information. CoRR **abs/1006.4046** (2010)
14. Golub, G.H., van Loan, C.F.: Matrix computations (3. ed.). Johns Hopkins University Press (1996)
15. Björck, A.: Numerical methods for least squares problems. SIAM (1996)
16. Griffiths, D.V., Smith, I.M.: Numerical methods for engineers. CRC press (2006)
17. Jain, A., Nandakumar, K., Ross, A.: Score normalization in multimodal biometric systems. Pattern Recognition **38** (2005) 2270–2285