Int. J. Appl. Math. Comput. Sci., 2014, Vol. 24, No. 1, 123–131
DOI: 10.2478/amcs-2014-0010

AN EFFICIENT EIGENSPACE UPDATING SCHEME FOR HIGH–DIMENSIONAL SYSTEMS

SIMON GANGL, DOMEN MONGUS, BORUT ŽALIK

Faculty of Electrical Engineering and Computer Science
University of Maribor, Smetanova ulica 17, 2000 Maribor, Slovenia
e-mail: `{simon.gangl,domen.mongus,borut.zalik}@um.si`

Systems based on principal component analysis have developed from exploratory data analysis in the past to current data processing applications which encode and decode vectors of data using a changing projection space (eigenspace). Linear systems, which need to be solved to obtain a constantly updated eigenspace, have increased significantly in their dimensions during this evolution. The basic scheme used for updating the eigenspace, however, has remained basically the same: (re)computing the eigenspace whenever the error exceeds a predefined threshold. In this paper we propose a computationally efficient eigenspace updating scheme, which specifically supports high-dimensional systems from any domain. The key principle is a prior selection of the vectors used to update the eigenspace in combination with an optimized eigenspace computation. The presented theoretical analysis proves the superior reconstruction capability of the introduced scheme, and further provides an estimate of the achievable compression ratios.

Keywords: eigenspace updating, projection space, data compression, principal component analysis.

1. Introduction

The use of eigenbases for projecting data was traditionally part of performing Principal Component Analysis (PCA) (Jolliffe, 1986). More precisely, performing a PCA includes the computation of a static eigenbasis, usually with the aim of exploratory data analysis (Lenz and Bui, 2004; Turk and Pentland, 1991). A typical task is to project various vectors of data using an identical eigenbasis, in order to obtain a representation that best explains the variance in the data (Han, 2010; Sumi *et al*., 2012). While PCA is continuously applied in various fields for statistical analysis, such as recently in the work of Skraban *et al*. (2013), it has also been successfully adapted for solving more complex tasks, e.g., for load forecasting in power systems (Siwek *et al*., 2009) and for automated recognition of faces (Liu *et al*., 2003; Turk and Pentland, 1991).

In some of the recently introduced applications, however, schemes based on PCA have been used for projecting as well as reconstructing data (Gang and Žalik, 2011; Perez-Iglesias *et al.*, 2005; Söderström and Li, 2005; 2007). More precisely, by projecting high-dimensional data (e.g., videos) to a suitable eigenbasis, the data can be successfully reconstructed on the basis of a significantly smaller vector of projection coefficients (Turk and Pentland, 1991). Based on this concept, several methods for data compression have been proposed (Perez-Iglesias *et al*., 2005; Söderström and Li, 2005; 2007). While the promise of encoding high-dimensional data using only a small set of coefficients seems interesting, a key problem is in computing, and even more so in maintaining, an eigenbasis which is capable of representing a constantly varying sequence of data. The difficulty in maintaining an accurate eigenbasis is primarily due to the mathematical complexity of its computation, which is in this case strongly amplified by the high data dimensionality. An estimate of that can be obtained by taking into account typical video data (ITU, 2007). In concrete terms, a single typical data vector contains at least several hundred thousand elements (Lo *et al*., 2003; Taubman and Marcellin, 2002). Considering state-of-the-art formats (e.g., HDTV), a single data vector can consist even of millions of elements (ITU, 2007). In the context of this paper, a high-dimensional system is thus any system using (eigen)vectors which contain at least several hundred thousand elements.

Previously proposed applications have sought to overcome the computational difficulty by introducing eigenspace update algorithms (Chandrasekaran *et al.*, 1997; Perez-Iglesias *et al.*, 2005). While all of those concepts are actually based on updating an eigenbasis (matrix), the commonly used term describing the problem is updating the eigenspace (matrix). For consistency with previously published works, the term *eigenspace updating scheme* is used here as well. An overview of the proposed PCA-based compression schemes reveals two basic concepts, which are common to most of them. Firstly, the eigenspace is updated whenever the error in the reconstructed data surpasses a predefined threshold (Liu *et al.*, 2003). The reconstruction error is estimated by computing the vector distance between the original data vector and the reconstructed data. Secondly, the computational cost of the schemes is reduced by computing estimates of the actual updated eigenspaces.

Using alternatives to a straightforward analytical eigenspace computation appears almost obligatory due to the high-dimensional data. The scheme proposed in this paper uses mathematical workarounds for this task as well, but is primarily designed to enable an efficient reconstruction process. On the other hand, updating the eigenspace in a manner that performs no analysis of the original data sequence before processing it is non-optimized, the more so that an initial analysis of the data is a long established concept in data compression and, therefore, at the core of many existing applications (Richardson, 2003; ITU, 2007). Another common property of many existing PCA-based applications is in their usability, which is limited to a single (video) data domain (Perez-Iglesias *et al.*, 2005; Söderström and Li, 2007). This is primarily due to the integration of domain-specific knowledge, introduced with the aim of decreasing reconstruction errors.

The aim of the present work is to introduce a computationally efficient, yet mathematically exact, eigenspace updating scheme which specifically supports high-dimensional systems from any domain. The updating scheme itself is introduced in Section 2. Section 3 presents the inverse (reconstruction) process, a theoretical analysis of the achievable compression ratios as well as a comparison with previous eigenspace updating schemes. The conclusion is given in Section 4.

2. Eigenspace updating for high-dimensional systems

Without loss of generality, the initial data can be interpreted as a time series, which is a model commonly used in statistics as well as signal processing (Spiegel *et al.*, 2011). The data is therefore represented by a finite sequence of T equally-sized vectors, denoted by $V = \{\boldsymbol{v}_t\}$, where each element $\boldsymbol{v}_t \in \mathbb{R}^D$ and the index $t \in [1, T]$. While such a partitioning can be derived from any kind of data (by padding), it is a natural representation of high-dimensional data, where the entirety is conventionally defined as a sequence of structurally identical elements. The sizes, or dimensionalities, of such elements are consequently identical as well.

By interpreting the original data as a finite sequence of vectors, the problem initially translates to selecting a subsequence of vectors from V, which are used to compute the eigenspace, denoted by $V^B = \{\boldsymbol{v}_i^B\}; V^B \subset V, i \in [1, I]$. The total number of selected base vectors is hence denoted by I, where $I < T$. Because the ability of the eigenspace to successfully represent and reconstruct data is primarily defined by the vectors used to compute it, the vectors included in V^B are referred to as base vectors (therefore the superscript B). Meanwhile, the algorithm used to obtain the Projection Space (PS) from the selected base vectors obviously remains the most critical part of the entire scheme. The residual vectors from the initial sequence V are afterwards projected using the obtained eigenspace, and therefore described as non-base vectors.

According to the introduced model, the proposed scheme consists of three phases:

- base vector selection,
- projection space creation, and
- non-base vector projection.

2.1. Base vector selection.

Even though no complex computation is performed during the initial phase, the base vector selection algorithm is of significant importance due to the relationship between the base vectors and the domain of successfully representable non-base vectors. Since the projection space domain is directly defined by the base vectors, selecting them in a process which is independent of the (data) domain is critical to establish an entirely domain independent scheme. Defining a single-criterion selection process is complicated by contradicting aims; while including fewer vectors decreases the computational burden, counting more vectors extends the domain of successfully representable vectors. More precisely, using more properly selected base vectors results in a broader, more detailed description of the domain. Therefore, a double-criterion base vector selection algorithm is introduced, comprising the following steps:

(i) The first vector of V, $\boldsymbol{v}_1$, is by definition included as the first base vector in V^B, thus $\boldsymbol{v}_1^B \overset{\text{def}}{=} \boldsymbol{v}_1$.

(ii) The subsequent $d \in \mathbb{N}^+$ vectors from V are not considered base vectors.

Here $\mathbb{N}^+$ is used to specifically exclude the possibility of $d = 0$, which would result in relying entirely on Step (iii) for the base vector selection.

(iii) Having performed Step (ii), we denote

- the last previously included base vector by $\boldsymbol{v}_i^B$,
- the currently observed vector from V by $\boldsymbol{v}_{t^0}$,
- the number of base vectors used (later-on) to compute the PS by $D^{\rm es} \in \mathbb{N}_{>1}$,
- the threshold, which is the lower limit for the similarity of two vectors by $\varepsilon \in \mathbb{R}$,

in order to define the next base vector $\boldsymbol{v}_{i+1}^B$ as

$$\begin{aligned}\boldsymbol{v}_{i+1}^B \overset{\rm def}{=} \boldsymbol{v}_{\min(t')} \quad & d_{l^1}(\boldsymbol{v}_{t'}, \boldsymbol{v}_{i'}^B) > \varepsilon, \\ & \forall \boldsymbol{v}_{i'}^B \in \{\boldsymbol{v}_{i-D^{\rm es}}^B, \dots, \boldsymbol{v}_i^B\} \wedge \boldsymbol{v}_{t'} \in \{\boldsymbol{v}_{t^0}, \dots, \boldsymbol{v}_T\},\end{aligned} \tag{1}$$

where $d_{l^1}(\boldsymbol{x}, \boldsymbol{y})$ denotes the l^1 vector distance, defined by

$$d_{l^1}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{D} \sum_{i=1}^{D} |x_i - y_i|. \tag{2}$$

The basic idea, which is formalised by Eqns. (1) and (2), is to define the next base vector as the first vector from V, which appears at least $1 + d$ elements after the previous base vector, and is at the same time sufficiently varying from the last $D^{\rm es}$ base vectors. Initially, when $D^{\rm es}$ base vectors have not yet been selected, the comparison is performed only for the already defined base vectors. While d and ε represent straightforward algorithm parameters, the definition of $D^{\rm es}$ is additionally discussed in Section 2.2.

(iv) Steps (ii) and (iii) are repeated until the entire sequence V has been analysed for possible base vectors.

(v) If not included during previous steps, the last element of V, $\boldsymbol{v}_{\boldsymbol{T}}$, is included as the final base vector $\boldsymbol{v}_I^B$, thus $\boldsymbol{v}_I^B \overset{\rm def}{=} \boldsymbol{v}_T$.

As mentioned, d and ε function as parameters of the base vector selection. Because their effect becomes critical only in practical applications, where reconstruction quality is perceived subjectively, the optimal values were found empirically (on image and video data, for example, the values used were $d \in [4, 6]$ and $\varepsilon \in [5, 8]$). Besides representing a generally subjective measure of similarity, the influence of d on the ratio between the encoded and the original data size (hereinafter referred to as Compression Ratio (CR)) is especially significant. Since the encoded non-base vectors are significantly smaller in size than the original and base vectors, the CR's upper bound converges to $1 : (1 + d)$ as the data dimensionality D goes to infinity. In more practical terms: as the data dimensionality increases, the CR is ensured to be at least $1 : (1 + d)$. A rigorous proof is presented in combination with the formal definition (see Theorem 1 in Section 3).

Since the base vectors are further on represented in a new sequence of vectors V^B, their indices in the original vector sequence V have to be known, thus stored, for a successful reconstruction as well. The sequence of non-base vectors, denoted by $V^{NB} = \{\boldsymbol{v}_j^{NB}\}; j \in [1, J]$, can now be finally defined as $V^{NB} = V \setminus V^B$.

2.2. Projection space creation.

As important as an optimized selection of the base vector subsequence may be, the most critical step remains the process of efficiently transforming the base vectors into projection spaces. Even if computing a single PS based on the entire subsequence V^B appears possible in theory, the high-dimensional as well as generally varying nature of data implies superior solutions. This is true especially from a computational point of view, where both factors have significant impact.

Based on the assumptions regarding the dimension and variability of the original vector sequence, we introduce the creation of a PS sequence. Each of the PS is computed from $D^{\rm es}$ adjacent base vectors, where in general $D^{\rm es}$ is significantly smaller than the number of selected base vectors I, especially when assuming that the original sequence V itself contains thousands of elements (i.e., $D^{\rm es} \ll I < T$). Adjacent PSs are not computed simply from adjacent sets of base vectors; rather, they are obtained from overlapping sets of base vectors. This approach is taken to mitigate the shortcomings of previous eigenspace updating schemes, where the reconstruction error increased evidently before each update (Liu *et al.*, 2003; Perez-Iglesias *et al.*, 2005; Söderström and Li, 2007). The reconstruction error of the proposed scheme is in contrast throughout of a continuous nature. This comparison is formally proven by Theorem 2.

Two important aspects of defining $D^{\rm es} \ll I$ need to be mentioned before the formal definition of the computation scheme. Firstly, using a rather small set of base vectors to compute the PS makes sense especially when the vector sequence is constantly varying. This holds because the eigenvectors forming the PS can successfully represent only vectors which show a certain degree of similarity. Therefore, using a PS obtained from a large number of base vectors would result in effectively 'using' only a small subset of the PS's eigenvectors, while the majority of the produced projection coefficients would be redundant. An additional aspect follows from the assumed high data dimensionality, which implies that the single possibility of defining a manageable system is in constructing it from a significantly smaller number of vectors ($D^{\rm es} \ll D$).

PS computation can now be formally defined. The base vectors from V^B are firstly aligned as columns of a single matrix, denoted by $M \in \mathbb{R}^{D\times I}$. Each PS is actually computed from a submatrix of M containing D^{es} adjacent columns. Any such submatrix is in Step (i) of PS creation denoted by M^n, whereby it contains D^{es} adjacent columns, counting from the n-th one onwards. The computation is then performed in the following four steps:

(i) The submatrix M^n is initially standardised. In other words, the separate dimension averages are translated to zero by subtracting the average dimension value from the elements of that dimension:

$$S = M^n - \bar{\boldsymbol{a}}^n \boldsymbol{e}^T, \tag{3}$$

where S denotes the resulting standardised matrix, $\boldsymbol{e}$ represents the ones vector of dimension D, and $\bar{\boldsymbol{a}}^n$ corresponds to the vector of dimension averages, which is defined by

$$\bar{\boldsymbol{a}}_i^n = \frac{1}{D^{\text{es}}} \sum_{j=1}^{D^{\text{es}}} M_{ij}^n. \tag{4}$$

Further, the vector of dimension averages $\bar{\boldsymbol{a}}^n$ is stored, since it is subtracted later on from the non-base vectors as well.

(ii) In order to compute the eigenvectors which define the PS, a Covariance Matrix (CM) has to be obtained first. In the usual PCA scheme, the general definition of a CM is used, thus it is defined as $S \cdot S^T$ (Jolliffe, 1986). Due to the above-mentioned circumstances, however, a slightly different approach is used, which follows from the works of Jin and Wei (2007), Nie *et al.* (2008) as well as Turk and Pentland (1991). While the final results are identical, the computational cost is reduced significantly, since $S \cdot S^T$, yet alone its Singular Value Decomposition (SVD), is never computed. Instead, the CM is defined as

$$C = S^T S, \tag{5}$$

where C is a significantly smaller CM (compared to $S \cdot S^T$), $C \in \mathbb{R}^{D^{\text{es}}\times D^{\text{es}}}$.

(iii) The following step is to compute the SVD of C. An important fact lies in the definition of C, which is a CM, and therefore at least positive semi-definite. A further consequence is that the left- and right-hand singular vectors of C are equal. Therefore, the matrix of left (respectively, right), singular vectors are denoted equally by W. The SVD of C is thus given by

$$C = W\Sigma W^T. \tag{6}$$

The initially required eigenvectors can now be obtained, beginning with the eigenvalue problem of $C = S^T S$,

$$S^T S\boldsymbol{y} = \lambda \boldsymbol{y}, \tag{7}$$

where λ and $\mathbf{y}$ denote the eigenvalues, respectively eigenvectors, of C. However, the eigenvector $\mathbf{y}$ can be substituted with W from the SVD. This is valid, because the singular- and eigenvectors, like the singular- and eigenvalues, of C are identical (C is positive semi-definite) (Meyer, 2000). Additionally, Eqn. (7) is left-multiplied by the standardised matrix S, thereby obtaining

$$SS^T SW = \lambda SW. \tag{8}$$

Finally, by introducing $U = SW$ (respectively, $\boldsymbol{x} = SW$), we obtain

$$SS^T U = \lambda U, \tag{9a}$$

$$SS^T \boldsymbol{x} = \lambda \boldsymbol{x}, \tag{9b}$$

where $\boldsymbol{x}$ denotes a particular eigenvector and U is the matrix of all eigenvectors. This conclusion (equality) is valid because the eigenvalues of SS^T, and $S^T S$, are identical as well (Meyer, 2000).

While Eqns. (7) through (9) are parts of the constructive proof, the solution is actually hiding in

$$U = SW, \tag{10}$$

where W is the matrix of singular vectors of C, S is the standardised matrix, and U is the above-mentioned eigenvector matrix of SS^T.

(iv) To be exact, the obtained matrix U actually represents only an orthogonal basis for SS^T. In other words, the columns of U require normalisation in order to obtain the required orthonormal set of eigenvectors.

By finally considering the computational load to compute W from M^n, respectively U from W, and additionally taking into account the dimensions of W in comparison with those of U, knowing W emerges as critical for an efficient PS construction. W is, therefore, in the following described as an intermediary eigenspace. Remembering (storing) the intermediary eigenspaces, which are computed during the coding process, is consequently the basic concept of the proposed (reconstruction) scheme that ensures efficient data reconstruction.

2.3. Non-base vector projection. Once the base vectors are selected, and the PSs are created, the final step of projecting the non-base vectors is almost a trivial task. Actually, the single unanswered important aspect remains: which PS exactly to use for projecting any given non-base vector?

Denoting the possibly computable PS by $U_1, U_2, \ldots, U_{I-D^{\text{es}}+1}$ (in respect to M^1, M^2, …, $M^{I-D^{\text{es}}+1}$), it is evident that not all of the PSs are required. At least every D^{es}-th PS has to be computed in order to achieve a continuous reconstruction error. Although D^{es} is assumed to be 'small', omitting the computation of some PS can evidently reduce the computational load. One possibility is to compute every second PS, and thereby reducing the possible computational costs by half.

Independently of the actually computed PS, the aim is always to project each non-base vector to the PS, which is defined by the eigenvectors most similar to it. Without additional analysis it can be assumed that for each non-base vector this is the PS which is computed from the base vectors that are closest to it in the original vector sequence. By using the PS in this way, each non-base vector is actually represented by eigenvectors which were obtained from base vectors positioned before as well as after it in V.

As for the projection itself, it is accomplished by straightforward matrix multiplication:

$$\boldsymbol{p_j} = U_n^T(\boldsymbol{v}_j^{NB} - \bar{\boldsymbol{a}}^n), \tag{11}$$

where $\boldsymbol{v}_j^{NB}$ represents any non-base vector from the non-base vector sequence ($\boldsymbol{v}_j^{NB} \in V^{NB}$), U_n^T is the PS, which is defined by the base vectors closest to $\boldsymbol{v}_j^{NB}$ in the original sequence, $\bar{\boldsymbol{a}}^n$ is the vector of dimension averages (as defined by Eqn. (4)) and, finally, $\boldsymbol{p}_j$ is the vector of projection coefficients ($\boldsymbol{p}_j \in \mathbb{R}^{D^{\text{es}}}$).

Of special significance is the ratio between the dimensions of the projected $\boldsymbol{p}_j$ and the initial non-base $\boldsymbol{v}_j^{NB}$ vectors, namely, $D^{\text{es}} : D$. That being the CR for any particular non-base vector, it was defined from the onset that $D^{\text{es}} \ll D$. One of the implications is the contribution of the projected non-base vectors to the overall CR, which becomes negligible as the vector dimensionality D increases.

3. Theoretical analysis

Since the introduced scheme is designed with the aim of efficiently reconstructing (data) vectors, the inverse (reconstruction) process is obviously computationally less expensive. Before a thorough analysis is performed, however, the details of the inverse process have to be given.

In general, the reconstruction process consists of performing the inverse of the last two steps, thus recreating the PS, and reconstructing the non-base vectors. The idea, as well as the equations, for recreating any PS are already known. As mentioned before, the previously computed intermediary eigenspaces W are used to directly transform a standardised matrix S into a PS using Eqn. (10). The steps of computing a CM and finding its SVD (Eqns. (5) and (6)) are hence omitted. Performing the standardisation procedure (Eqns. (3) and (4)) from the beginning is less critical, since the (here unknown) vector of dimension averages $\bar{\boldsymbol{a}}^n$ is required for reconstructing non-base frames as well.

Once the PS U_n and the vector of dimension averages $\bar{\boldsymbol{a}}^n$ are recreated, a non-base vector $\boldsymbol{v}_j^{NB}$ is reconstructed from the vector of projection coefficients $\boldsymbol{p}_j$ by

$$\boldsymbol{v'}_j^{NB} = U_n\boldsymbol{p}_j + \bar{\boldsymbol{a}}^n, \tag{12}$$

where $\boldsymbol{v'}_j^{NB}$ denotes the reconstructed non-base vector $\boldsymbol{v}_j^{NB}$, where in general $\boldsymbol{v'}_j^{NB} \approxeq \boldsymbol{v}_j^{NB}$.

Proceeding with the theoretically ensured properties of the scheme, an assumption regarding the 'worst case' CR was already given in Section 2.1, but left without a proof.

Theorem 1. *As the data dimensionality within the context of the presented scheme goes to infinity, $D \to \infty$, the compression ratio's upper bound converges to $1 : (1 + d)$.*

Proof. The CR, denoted equally by CR, is formally defined by (Taubman and Marcellin, 2002)

$$CR = \frac{\text{encoded\_data\_size}}{\text{original\_data\_size}}, \tag{13}$$

where the original_data_size equals the number of vectors in V multiplied by the vector dimensionality, thus $T \cdot D$. Meanwhile, encoded_data_size equals the combined size of all data, required to reconstruct the original data. More precisely, encoded_data_size equals the sum of sizes of the non-encoded base vectors, the vectors of projection coefficients, and the intermediary eigenspaces. Assuming that every second eigenspace is computed (stored), the three mentioned terms sum up to

$$\begin{aligned}&\text{encoded\_data\_size}\\&\qquad = ID + (T - I)D^{es} + \frac{I}{2}D^{\text{es}2},\end{aligned} \tag{14}$$

where the number of eigenspaces is rounded from $I - N_{\text{es}} + 1$ to I.

Using the derived values in Eqn. (13) results in

$$CR = \frac{ID + (T - I)D^{\text{es}} + \frac{I}{2}D^{\text{es}2}}{TD}, \tag{15}$$

where the number of base frames I still needs to be resolved.

Following the assumption of a 'worst case' scenario, the upper bound of CR is defined by selecting the most possible base vectors. In terms of the base vector selection algorithm, this would be equivalent to setting $\varepsilon = 0$, and therefore producing

$$I = \frac{T}{1+d}, \tag{16}$$

hence skipping always exactly d vectors from v_n before including the next base vector.

Rearranging Eqn. (15) and combining it with Eqn. (16) result in

$$\begin{aligned} CR &= \frac{ID}{TD} + \frac{TD^{\mathrm{es}}}{TD} - \frac{ID^{\mathrm{es}}}{TD} + \frac{ID^{\mathrm{es}2}}{2TD} \\ &= \frac{I}{T} + \frac{D^{\mathrm{es}}}{D} + \frac{ID^{\mathrm{es}}(D^{\mathrm{es}}-2)}{2TD} \\ &= \frac{T}{(1+d)T} + \frac{D^{\mathrm{es}}}{D} + \frac{TD^{\mathrm{es}}(D^{\mathrm{es}}-2)}{2(1+d)TD} \\ &= \frac{1}{1+d} + \frac{D^{\mathrm{es}}}{D} + \frac{D^{\mathrm{es}}(D^{\mathrm{es}}-2)}{2(1+d)D}. \end{aligned} \tag{17}$$

Taking finally into account that $D \to \infty$ provides the proof as

$$\lim_{D\to\infty} CR = \lim_{D\to\infty} \left(\frac{1}{1+d} + \frac{D^{\mathrm{es}}}{D} + \frac{D^{\mathrm{es}}(D^{\mathrm{es}}-2)}{2(1+d)D} \right),$$

$$\lim_{D\to\infty} CR = \frac{1}{1+d}. \tag{18}$$

■

While the influence of d on the CR is to some degree less important from a theoretical point of view, it provides a straightforward estimate as well as a control parameter of the expected CR in practical environments.

Besides analysing the scheme, a comparison with previously presented eigenspace updating schemes is of critical importance. The proposed scheme is therefore analytically compared with the generally used updating scheme, where the eigenspace is updated whenever the reconstruction error exceeds a predefined threshold (Perez-Iglesias *et al.*, 2005; Liu *et al.*, 2003). In order to formalise the comparison, though, a measure of the reconstruction error needs to be defined firstly. For this purpose, the reconstruction error sequence is introduced. For the presented scheme it is denoted by $E = \{e_t\}; e_t \in \mathbb{R}, t \in [1,T]$, and (since $V^{NB} = V \setminus V^B$) defined as

$$e_t = \begin{cases} d_{l^1}(\boldsymbol{v}_t, \boldsymbol{v}_t') & \text{if } \boldsymbol{v}_t \in V^{NB}, \\ 0 & \text{otherwise}, \end{cases} \tag{19}$$

where $\boldsymbol{v}_t$ represents a vector from V, which has not been defined as a base vector, $\boldsymbol{v}_t'$ represents its reconstruction using the introduced scheme, and the error is computed by using the metric as defined in Eqn. (2). All of the base vectors are known in their initial form during the reconstruction process, and therefore introduce no error to the sequence.

The reconstruction error sequence of the compared scheme, denoted by $G = \{g_t\}, g_t \in \mathbb{R}, t \in [1,T]$, is meanwhile defined equivalently to E, assuming that the base vectors are selected, respectively the eigenspace is updated, whenever the reconstruction error exceeds a predefined threshold. Additionally, the average reconstruction errors are defined by

$$\bar{E} = \frac{1}{T}\sum_{t=1}^{T} e_t, \tag{20a}$$

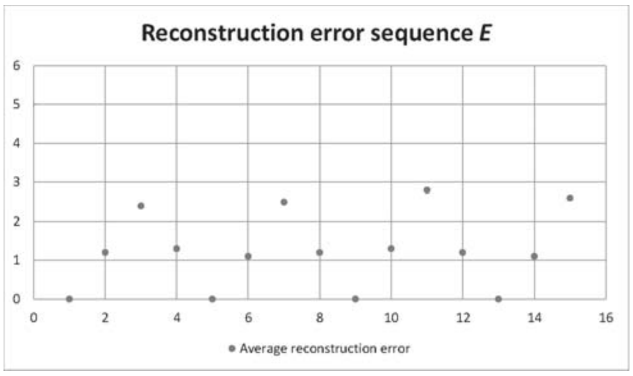
$$\bar{G} = \frac{1}{T}\sum_{t=1}^{T} g_t. \tag{20b}$$

While the key difference between the two schemes has already been mentioned (see Section 2.2), it is formalised by the following result:

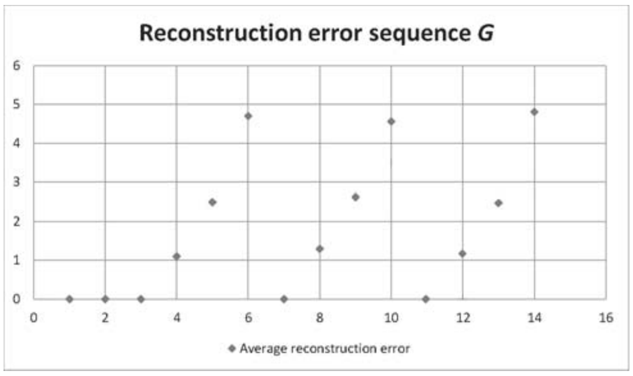
Theorem 2. *The reconstruction error sequence E of the introduced eigenspace updating scheme is always of a periodic and continuous nature. The average reconstruction error $\bar{E}$ is additionally always smaller than the average reconstruction error $\bar{G}$ created by using an updating scheme where the eigenspace is updated always when the reconstruction error exceeds a predefined threshold.*

Proof. As the above theorem compares two different but conceptually similar schemes, it has to be understood within its scope. It is obviously assumed that for the comparison both schemes are applied to the same vector sequence, V. Using the exactly same subsequence of base vectors V^B is impossible due to the varying definitions. Therefore, it is assumed that both schemes use every $(1 + d)$-th vector from V as a base vector (excluding the first D^{es} vectors, which are used for initialising the eigenspace), and compute the PS always from D^{es} vectors at a time. The roles of parameter ε in the presented scheme (respectively, of the predefined threshold in the compared scheme) are eliminated by applying these assumptions. It is important to note that such an approach modifies (respectively, narrows) the schemes to some degree, but primarily it creates the required neutral conditions where the two algorithms used to update the eigenspace can be compared. Hence, the proof is focused on the effect of the modified updating scheme.

Figure 1 shows an illustration of typical reconstruction error sequences E and G as obtained from numerical experiments using video data of varying content. Since the experiments were performed on various samples, the illustration in Fig. 1 represents the averaged results, whereby the exact parameters used are given in



(a)



(b)

Fig. 1. Illustrations of typical reconstruction error sequences as obtained by the proposed scheme (a), and by the scheme, where the eigenspace is updated when the error crosses a predefined threshold (b). The according parameters used in (a) are $D^{\mathrm{es}} = 3$, $d = 3$, and $\varepsilon = 0$, while the compared scheme (b) updates the eigenspace on every $(1 + d)$-th, thus 4-th vector. No actual threshold is used in this case in (b), in order to create the required neutral conditions.

the figure caption. Although some assumptions were necessary in order to be able to compare the two schemes, a computational proof would require additionally the exact evaluation of the reconstruction error sequences. Since this is beyond the scope of this paper, only the reasoning for the correctness of the first part of the stated theorem is given. Even though confirmed by various experiments, which are also illustrated in Fig. 1, this has to be consequently understood rather as an indication than an exact proof.

The graphs of the sequences E and G clearly indicate that actually both of the reconstruction error sequences are of a periodic nature. Intuitively this can be traced to the periodic nature of the distance between the position of a given vector and the base vector used for its projection, which is closest to it in the initial sequence. While we develop this idea in order to prove the second part of the stated theorem, it says little about the continuity of the reconstruction error sequence E.

Comparing the graphs, it is evident that the intervals of discontinuous nature in G, actually the intervals where the eigenspace is updated, are transformed to intervals of a continuous nature. The key observation here is the influence of the introduced scheme, where the base vectors are throughout the sequence chosen in advance. Therefore, the number of elements between the currently projected vector and the base vector closest to it always follow a continuous sequence. Even more, all of the non-base vectors are projected to a PS that is computed from base vectors which are positioned before as well as after them in the original sequence V.

While the given reasoning implies to some degree the correctness of the second part of the stated theorem as well, an analytic proof is complicated again by the requirement to exactly evaluate the elements of E, respectively G. To actually prove the second part, however, it is sufficient to use any estimate which preserves relations between the elements of the sequences. The most critical relation is formalised by

$$e_t < g_t \Leftrightarrow e'_t < g'_t, \quad \forall e_t \in E \wedge \forall g_t \in G, \tag{21}$$

where e_t and g_t represent elements of the reconstruction error sequences E and G, while e'_t and g'_t represent the estimated values of the according elements from E, respectively G. In simpler terms: as long as any element e_t, which is smaller than g_t, is transformed into an element e'_t, which is smaller than g'_t, and vice-versa, the estimate producing this transformation can be used to prove the second part of Theorem 2.

Based on the assumptions made, and confirmed by the described experiments, the distance between a non-base vector and the closest base vector, which is used for creating the PS, represents an appropriate estimate of the reconstruction error. This estimate is used for both schemes, since both are assumed to compute the PS and project the non-base vectors using the same equations. The definition for the estimated values e'_t and g'_t is thus defined for both schemes by

$$\{e'_t, g'_t\} \mapsto |t - i|, \quad \boldsymbol{v}_t \in V \wedge \boldsymbol{v}_i^B \in V^B, \tag{22}$$

where, most importantly, $\boldsymbol{v}_i^B$ denotes the base vector closest to $\boldsymbol{v}_t$, which was used to create the PS that $\boldsymbol{v}_t$ was projected onto. In this particular equation, i refers to the index of $\boldsymbol{v}_{\boldsymbol{i}}^{\boldsymbol{B}}$ in the initial sequence V, not the base vector sequence V^B.

With all elements finally in place, the average reconstruction errors $\bar{E}$ and $\bar{G}$ can now be estimated as

$$\bar{E} \approx \frac{1}{T} \sum_{t=1}^{T} e'_t, \tag{23a}$$

$$\bar{G} \approx \frac{1}{T}\sum_{t=1}^{T} g'_t. \tag{23b}$$

While using the entire sequences E' and G' is required to obtain an exact value, the proof can be obtained actually from shorter subsequences. As an estimate is used here, computing an exact value (of the estimate) provides obviously no additional information. Since all of the base vectors are reconstructed without errors, they can be, without loss of validity, omitted. Even more, because both sequences are periodical (as illustrated in Fig. 1), the result is equal when computing the average reconstruction error of a single period of reconstructed non-base vectors. Using the developed ideas in Eqn. (23) results in

$$\bar{E} \approx \frac{1}{d}\sum_{t=1}^{d} e'_t, \tag{24a}$$

$$\bar{G} \approx \frac{1}{d}\sum_{t=1}^{d} g'_t. \tag{24b}$$

Finally, by using the values produced by Eqn. (22) in (24), the average reconstruction errors can be computed as

$$\bar{E} = \begin{cases} \frac{1}{d} 2 \sum_{i=1}^{d/2} i & \text{if } d \text{ is even,} \\ \frac{1}{d}\left(\sum_{i=1}^{\lfloor d/2 \rfloor} i + \sum_{i=1}^{\lceil d/2 \rceil} i\right) & \text{if } d \text{ is odd,} \end{cases} \tag{25a}$$

$$\bar{G} = \frac{1}{d}\sum_{i=1}^{d} i. \tag{25b}$$

Only the most interesting scenario, the case of $\bar{E}$ when d is odd, is explicitly developed here:

$$\begin{aligned} \bar{E} &= \frac{1}{d}\Big(\sum_{i=1}^{\lfloor d/2 \rfloor} i + \sum_{i=1}^{\lceil d/2 \rceil} i\Big) \\ &= \frac{1}{d}\Big(\frac{(1+\frac{d-1}{2})(d-1)}{4} + \frac{(1+\frac{d+1}{2})(d+1)}{4}\Big) \\ &= \Big(\frac{\frac{d(d-1)}{2} - \frac{d-1}{2} + d - 1 + \frac{d(d+1)}{2} + \frac{d+1}{2} + d + 1}{4d}\Big) \\ &= \frac{\frac{d(d-1+(d+1))}{2} + \frac{d+1+1-d}{2} + 2d}{4d} \\ &= \frac{d^2+2d+1}{4d} = \frac{d}{4} + \frac{1}{2} + \frac{1}{4d}. \end{aligned} \tag{26}$$

Meanwhile, evaluating all three scenarios (by the principally identical process) finally provides the proof as

$$\bar{E} = \begin{cases} \frac{d}{4} + \frac{1}{2} & \text{if } d \text{ is even,} \\ \frac{d}{4} + \frac{1}{2} + \frac{1}{4d} & \text{if } d \text{ is odd,} \end{cases} \tag{27a}$$

$$\bar{G} = \frac{d}{2} + \frac{1}{2}. \tag{27b}$$

■

It is important to note that different metrics, for example, as defined by Eqn. (2), are generally used in practical applications. The proof, as well as the theorem itself, does not therefore exactly compute the ratio between the average reconstruction errors. This, however, does not harm the validity of either.

4. Conclusion

While several new applications for PCA-based systems have been proposed in the recent years, no dedicated mechanism for using projection spaces (eigenspaces) in a truly dynamic environment has been developed. The approach of using originally static algorithms, and schemes, has proven to be sufficient only for basic tasks, respectively low-dimensional problems.

In order to use eigenspaces for the projection and reconstruction of high-dimensional (data) vectors, an eigenspace updating scheme has been introduced, which performs a beforehand analysis of the data to optimally select the base vectors. An efficient reconstruction process is achieved by employing 'intermediary' eigenspaces, which are basically intermediate products of the eigenspace computation stored for later use.

The reconstruction quality, as well as the achievable compression ratios of the introduced scheme, have been examined analytically. Not only does the introduced scheme provide superior reconstruction in comparison with the previously used approach, but also the compression ratio can be practically defined by a single parameter.

Acknowledgment

This work was supported by the Slovenian Research Agency under the grant 1000-10-310121 and the project P2-0041, and produced within the framework of the operation entitled *Centre of Open Innovation and Research UM*. The operation is co-funded by the European Regional Development Fund and conducted within the framework of the Operational Programme for Strengthening Regional Development Potentials for the period 2007–2013, Development priority 1: Competitiveness of companies and research excellence, Priority axis 1.1: Encouraging competitive potential of enterprises and research excellence.

References

Chandrasekaran, S., Manjunath, B., Wang, Y., Winkeler, J. and Zhang, H. (1997). An eigenspace update algorithm for image analysis, *Graphical Models and Image Processing* **59**(5): 321–332.

Gangl, S. and Žalik, B. (2011). Partially lossless compression of dicom image sets, *Anales del Congreso Argentino de Informatica y Salud, Córdoba, Argentina*, pp. 131–136.

Han, X. (2010). Nonnegative principal component analysis for cancer molecular pattern discovery, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **7**(3): 537–549.

ITU (2007). ITU-T Recommendation H.264: Advanced video coding for generic audiovisual services.

Jin, X.-Q. and Wei, Y.-M. (2007). A short note on singular values of optimal and superoptimal preconditioned matrices, *International Journal of Computer Mathematics* **84**(8): 1261–1263.

Jolliffe, I.T. (1986). *Principal Component Analysis*, 1st Edn., Springer, New York, NY.

Lenz, R. and Bui, T.H. (2004). Recognition of non-negative patterns, *Proceedings of the 17th International Conference on Pattern Recognition ICPR 2004, Cambridge, UK*, Vol. 3, pp. 498–501.

Liu, X., Chen, T. and Thornton, S.M. (2003). Eigenspace updating for non-stationary process and its application to face recognition, *Pattern Recognition* **36**(9): 1945–1959.

Lo, S.-C. B., Li, H. and Freedman, M.T. (2003). Optimization of wavelet decomposition for image compression and feature preservation, *IEEE Transactions on Medical Imaging* **22**(9): 1141–1151.

Meyer, C.D. (2000). *Matrix Analysis and Applied Linear Algebra*, Society for Industrial and Applied Mathematics, Philadelphia, PA.

Nie, Y.Y., Li, Z. and Han, J.D. (2008). Origin-shifted algorithm for matrix eigenvalues, *International Journal of Computer Mathematics* **85**(9): 1397–1411.

Perez-Iglesias, H., Dapena, A. and Castedo, L. (2005). A novel video coding scheme based on principal component analysis, *2005 IEEE Workshop on Machine Learning for Signal Processing, Mystic, CT, USA,* pp. 361–366.

Richardson, I. (2003). *H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia*, Wiley, Chichester.

Siwek, K., Osowski, S. and Szupiluk, R. (2009). Ensemble neural network approach for accurate load forecasting in a power system, *International Journal of Applied Mathematics and Computer Science* **19**(2): 303–315, DOI: 10.2478/v10006-009-0026-2.

Skraban, J., Dzeroski, S., Zenko, B., Mongus, D., Gangl, S. and Rupnik, M. (2013). Gut microbiota patterns associated with colonization of different *Clostridium difficile* ribotypes, *PLoS ONE* **8**(2): e58008.

Söderström, U. and Li, H. (2005). Very low bitrate full-frame facial video coding based on principal component analysis, *Proceedings of the Signal and Image Processing Conference, Honolulu, HI, USA*, pp. 127–132.

Söderström, U. and Li, H. (2007). Principal component video coding for simple decoding on mobile devices, *Proceedings of the Swedish Symposium on Image Analysis, Linköping, Sweden*, no. 47, pp. 149–152.

Spiegel, S., Gaebler, J., Lommatzsch, A., De Luca, E. and Albayrak, S. (2011). Pattern recognition and classification for multivariate time series, *Proceedings of the 5th International Workshop on Knowledge Discovery from Sensor Data, Sensor KDDM'11, San Diego, CA, USA*, pp. 34–42.

Sumi, S.M., Zaman, M.F. and Hirose, H. (2012). A rainfall forecasting method using machine learning models and its application to the Fukuoka city case, *International Journal of Applied Mathematics and Computer Science* **22**(4): 841–854, DOI: 10.2478/v10006-012-0062-1.

Taubman, D.S. and Marcellin, M.W. (2002). *JPEG2000: Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, Boston, MA.

Turk, M. and Pentland, A. (1991). Eigenfaces for recognition, *Journal of Cognitive Neuroscience* **3**(1): 71–86.

Simon Gangl is a researcher at the University of Maribor, Slovenia. He completed his B.Sc. in computer science in 2010 at the University of Maribor. He is a Ph.D. candidate at the Laboratory for Geometric Modeling and Multimedia Algorithms. His research interests include image and video compression, medical imaging techniques, image processing, and biometric applications.

Domen Mongus is a researcher at the University of Maribor, Slovenia. He obtained his Ph.D. in computer science in 2013 at the University of Maribor. His research interests include volumetric data compression, LiDAR data classification, geographic information systems, biometric applications, and scientific visualization.

Borut Žalik is a professor of computer science at the University of Maribor, Slovenia. He obtained his Ph.D. in computer science in 1993 at the University of Maribor. He is the head of the Laboratory for Geometric Modeling and Multimedia Algorithms. His research interests include computational geometry, geometric data compression, scientific visualization, and geographic information systems.

Received: 27 June 2013
Revised: 6 November 2013
Re-revised: 22 November 2013