On the Wiberg Algorithm for Matrix Factorization in the Presence of Missing Components

Takayuki Okatani and Koichiro Deguchi
Graduate School of Information Sciences, Tohoku University
Aramaki Aza Aoba 6-6-1
980-8579 Sendai, Japan
Email:okatani@fractal.is.tohoku.ac.jp

Abstract

This paper considers the problem of factorizing a matrix with missing components into a product of two smaller matrices, also known as principal component analysis with missing data (PCAMD). The Wiberg algorithm is a numerical algorithm developed for the problem in the community of applied mathematics. We argue that the algorithm has not been correctly understood in the computer vision community. Although there are many studies in our community, almost every one of which refers to the Wiberg study, as far as we know, there is no literature in which the performance of the Wiberg algorithm is investigated or the detail of the algorithm is presented. In this paper, we present derivation of the algorithm along with a problem in its implementation that needs to be carefully considered, and then examine its performance. The experimental results demonstrate that the Wiberg algorithm shows a considerably good performance, which should contradict the conventional view in our community, namely that minimization-based algorithms tend to fail to converge to a global minimum relatively frequently. The performance of the Wiberg algorithm is such that even starting with random initial values, it converges in most cases to a correct solution, even when the matrix has many missing components and the data are contaminated with very strong noise. Our conclusion is that the Wiberg algorithm can also be used as a standard algorithm for the problems of computer vision.

1 Introduction

This paper deals with the problem of factorizing a matrix into a product of two smaller matrices, assuming the rank of the matrix to be a known number, as follows:

$$\mathbf{Y} \to \mathbf{U}\mathbf{V}^\top. \tag{1}$$

This problem commonly appears in several problems of computer vision, such as structure from motion (SFM) [10, 7, 4, 2] and computation of shape and illumination from an image set taken under different illumination conditions [6, 5, 1]. Considering the presence of noise in the data, the problem is most often formulated as finding $\mathbf{U}$ and $\mathbf{V}$ that minimize the error $\| \mathbf{Y} - \mathbf{U}\mathbf{V}^\top \|_F^2$. In this paper, we consider the case where there are missing components in $\mathbf{Y}$. In this case, the function to be minimized is defined to be the error sum $\| \mathbf{Y} - \mathbf{U}\mathbf{V}^\top \|_F^2$ only over the existing components. Unlike the case without missing components, where the problem is reduced to an eigenvalue problem, in this case, we have to directly solve the minimization problem that is truly nonlinear; thus, convergence and computational complexity of its algorithms are of significant interest.

It is the study [9] by Shum et al. in the computer vision community that first introduced the problem and numerical algorithms for it. They rediscovered the study of Wiberg [11] in the 70's, based on which they developed an algorithm and applied it to 3D object modeling. Since then, many algorithms have been proposed in our community for the problem. There is a good survey of them in the recent study [2] by Buchanan et al (See also [3]).

The study of Wiberg [11] (and that of Ruhe and Wedin [8], on which it is based) should be a basis for the mentioned initiative work of Shum et al. as well as many subsequent studies. In fact, the Wiberg name is cited in almost every one of those studies. However, we would like to point out the possibility that the Wiberg algorithm has not been correctly understood in our community. As far as we know, there is no literature (within our community) in which its numerical performance is investigated or the algorithm itself is presented. In the paper [9] of Shum et al., a brief outline of the Wiberg algorithm is presented, but no experimental result by the algorithm is shown. In the recent paper [2] of Buchanan et al., where many algorithms are classified into several categories, the Wiberg algorithm is mentioned but not explicitly classified. In some literature, the algorithm seems even to be confused with that of the alternated least squares (ALS). As will be shown later, the Wiberg algorithm is based on the Gauss-Newton algorithm and is different from the ALS algorithm.

The goal of this paper is firstly to present the Wiberg algorithm correctly and then to accurately investigate its numerical performance. There are several possible reasons for the above-mentioned misunderstanding of the Wiberg algorithm. One is that the Wiberg paper lacks detailed explanations of the underlying logic of the algorithm. Our derivation in what follows will supplement these. Another possible reason for the misunderstanding might be that the equation for computing the Gauss-Newton update in the Wiberg algorithm is always degenerate. This needs to be borne in mind when implementing the algorithm. We prove the degeneracy and show how to handle it.

As will be shown in what follows, our experimental results show that the Wiberg algorithm (with our implementation) shows a considerably good numerical performance, which may contradict the conventional view in our community concerning the performance of algorithms for the problem. For example, there are several studies [7, 4] of a different type of algorithm that is based on imputation, in which missing components are locally estimated from those existing. These studies are motivated by a recognition that the minimization-based algorithms do not have good convergence performance, and good initial values are necessary to have them converge to a global minimum. However, as far as the Wiberg algorithm is concerned, choosing good initial values does not seem to be necessary. Even starting with completely random initial values, the algorithm will converge to a global minimum for most of the cases, as will be shown in what follows. We also show brief comparisons of some of the minimization-based algorithms in terms of computational complexity, in which the Wiberg algorithm is superior to others.

2 Notation

In this paper, we consider factorization of the form:

$$\mathbf{Y} \to \mathbf{U}\mathbf{V}^\top, \tag{2}$$

as well as factorization with a mean vector:

$$\mathbf{Y} \to \mathbf{U}\mathbf{V}^\top + \mathbf{1}_m\mu^\top, \tag{3}$$

where $\mathbf{Y}$, $\mathbf{U}$ and $\mathbf{V}$ are matrices of $m \times n$, $m \times r$, and $n \times r$, respectively, $\mathbf{1}_m$ is a m-vector of all 1's and μ is a n-vector. Although in what follows, the form (2) is mainly considered for the sake of simplicity, the results for (2) are applicable to (3) with only a few modifications.

Some components of $\mathbf{Y}$ are missing. Let $\mathbf{H}$ be an $m \times n$ matrix indicating which components are missing; $\mathbf{H}$ is such that the component h_{ij} indicates the existence of y_{ij}; $h_{ij} = 1$ if y_{ij} exists and $h_{ij} = 0$ if y_{ij} is missing. This matrix $\mathbf{H}$ will be called the indicator matrix and is assumed to be known throughout this paper. Then, for the factorization form (2), the problem is formulated as a minimization problem with respect to $\mathbf{U}$ and $\mathbf{V}$:

$$\phi(\mathbf{U}, \mathbf{V}) \equiv \| \mathbf{H} \odot (\mathbf{Y} - \mathbf{U}\mathbf{V}^\top) \|_F^2 \to \text{min.}, \tag{4}$$

where $\odot$ represents the component-wise product of matrices. For the factorization form (3) with a mean vector, the problem is formulated as minimization with respect to $\mathbf{U}$, $\mathbf{V}$, and μ:

$$\phi'(\mathbf{U}, \mathbf{V}, \mu) \equiv \| \mathbf{H} \odot (\mathbf{Y} - \mathbf{U}\mathbf{V}^\top - \mathbf{1}_m \mu^\top) \|_F^2 \to \text{min.} \tag{5}$$

The functions ϕ and ϕ' are inconvenient to manipulate analytically, because of the $\odot$ product with $\mathbf{H}$. Thus, we rewrite them into a simpler form without $\mathbf{H}$, by introducing several notations. Let $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]^\top$ and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]^\top$; $\mathbf{u}_i$ and $\mathbf{v}_j$ are both r-vectors. Also, let $\mathbf{u} \equiv [\mathbf{u}_1^\top, \ldots, \mathbf{u}_m^\top]^\top$ and $\mathbf{v} \equiv [\mathbf{v}_1^\top, \ldots, \mathbf{v}_n^\top]^\top$. In what follows, $\mathbf{U}$ and the mr-vector $\mathbf{u}$ will be used in an interchangeable manner, and so are $\mathbf{V}$ and the nr-vector $\mathbf{v}$; for example, $\phi(\mathbf{U}, \mathbf{V}) = \phi(\mathbf{u}, \mathbf{v})$. Let p be the number of observed components, and $\mathbf{y}$ be a p-vector containing only observed components y_{ij} in lexical order of i and j. Using a $p \times mr$ matrix $\mathbf{F}$ containing $\mathbf{u}_1, \ldots, \mathbf{u}_m$ and a $p \times nr$ matrix $\mathbf{G}$ containing $\mathbf{v}_1, \ldots, \mathbf{v}_n$, $\phi(\mathbf{u}, \mathbf{v})$ can be rewritten as follows:

$$\phi(\mathbf{u}, \mathbf{v}) = \| \mathbf{F}\mathbf{u} - \mathbf{y} \|^2 = \| \mathbf{G}\mathbf{v} - \mathbf{y} \|^2 \tag{6}$$

The matrices **F** and **G** have the following structures:

$$\mathbf{F} \equiv \begin{bmatrix} \mathbf{v}_1^\top & & & \\ \mathbf{v}_2^\top & & & \\ \vdots & & & \\ \mathbf{v}_n^\top & & & \\ & \mathbf{v}_1^\top & & \\ & \mathbf{v}_2^\top & & \\ & \vdots & & \\ & \mathbf{v}_n^\top & & \\ & & \ddots & \\ & & & \mathbf{v}_1^\top \\ & & & \mathbf{v}_2^\top \\ & & & \vdots \\ & & & \mathbf{v}_n^\top \end{bmatrix}, \qquad \mathbf{G} \equiv \begin{bmatrix} \mathbf{u}_1^\top & & & \\ & \mathbf{u}_1^\top & & \\ & & \ddots & \\ & & & \mathbf{u}_1^\top \\ \mathbf{u}_2^\top & & & \\ & \mathbf{u}_2^\top & & \\ & & \ddots & \\ & & & \mathbf{u}_2^\top \\ & & \vdots & \\ \mathbf{u}_m^\top & & & \\ & \mathbf{u}_m^\top & & \\ & & \ddots & \\ & & & \mathbf{u}_m^\top \end{bmatrix}. \tag{7}$$

Note that the above shows only the basic structures of the matrices for notational simplicity, and in the presence of missing components, $\mathbf{v}_j$'s and $\mathbf{u}_i$'s do not regularly appear in them, as implied above. The matrices will have only the rows corresponding to the observed components. For example, if y_{12} is missing, the second rows of the above **F** and **G** are removed, and if y_{mn} is missing, the last rows are removed. Thus, the number of rows of **F** and **G** becomes the number p of observed components.

The distribution of the missing/observed components in **Y** (i.e., the indicator matrix **H**) determines the size and structure of **F**, and therefore, for a fixed **H**, **F** can be viewed as a function of **v**; thus, we will write $\mathbf{F}(\mathbf{v})$. Similarly, we will write $\mathbf{G} = \mathbf{G}(\mathbf{u})$.

For the factorization form (3) with a mean vector (and the corresponding cost ϕ'), we define $\mathbf{m} = [\mu_1, \ldots, \mu_n, \ldots, \mu_1, \ldots, \mu_n]^\top$ (m-repetition of a vector $[\mu_1, \ldots, \mu_n]$ while excluding the entries corresponding to the missing components) and then $\tilde{\mathbf{V}} \equiv [\mathbf{V}, \mathbf{m}]$. We denote each row vector of $\tilde{\mathbf{V}}$ by $\tilde{\mathbf{v}}_j$, i.e., $\tilde{\mathbf{V}} = [\tilde{\mathbf{v}}_1, \tilde{\mathbf{v}}_2, \ldots \tilde{\mathbf{v}}_n]^\top$. Then stacking these vectors, define $\tilde{\mathbf{v}}^\top = [\tilde{\mathbf{v}}_1^\top, \tilde{\mathbf{v}}_2^\top, \ldots \tilde{\mathbf{v}}_n^\top]$. Using these notations, ϕ' is represented as

$$\phi'(\mathbf{u}, \tilde{\mathbf{v}}) = |\,\mathbf{F}\mathbf{u} + \mathbf{m} - \mathbf{y}\,|^2 = |\,\tilde{\mathbf{G}}\tilde{\mathbf{v}} - \mathbf{y}\,|^2,$$

where $\tilde{\mathbf{G}}$ is a $p \times n(r+1)$ matrix such that $(\tilde{\mathbf{G}}\tilde{\mathbf{v}} - \mathbf{y})_k$ yields $\mathbf{u}_i^\top \mathbf{v}_j + \mu_j - y_{ij}$ assuming that y_{ij} is the k-th observed element; $\tilde{\mathbf{G}}$ is formally obtained by replacing $\mathbf{u}_i^\top$'s on $\mathbf{G}$ by $[\mathbf{u}_i, 1]^\top$'s on Eq.(7).

3 The Wiberg algorithm and its implementation

The Wiberg algorithm was proposed for the minimization problem (5) [11]. (It is straightforward to make it applicable to the problem (4).) This section shows the derivation of the algorithm in detail and then its implementation.

3.1 Alternated least squares algorithm

To begin with, we summarize the alternated least squares (ALS) algorithm that are often confused with the Wiberg algorithm. From now on, we deal with only the factorization form (2) without a mean vector.

We want to find a minimum of ϕ. Then, we search for a solution to the equations $\partial\phi/\partial\mathbf{u} = \partial\phi/\partial\mathbf{v} = \mathbf{0}$. Using the notations introduced above, these are expressed as

$$\begin{bmatrix} \partial\phi/\partial\mathbf{u} \\ \partial\phi/\partial\mathbf{v} \end{bmatrix} = \begin{bmatrix} \mathbf{F}^\top(\mathbf{F}\mathbf{u} - \mathbf{y}) \\ \mathbf{G}^\top(\mathbf{G}\mathbf{v} - \mathbf{y}) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}. \tag{8}$$

When considering the two equations independently, solutions are given by

$$\hat{\mathbf{u}} = (\mathbf{F}^\top\mathbf{F})^{-1}\mathbf{F}^\top\mathbf{y}, \tag{9a}$$

$$\hat{\mathbf{v}} = (\mathbf{G}^\top\mathbf{G})^{-1}\mathbf{G}^\top\mathbf{y}. \tag{9b}$$

Based on these bilinear relations, the ALS algorithm updates $\mathbf{u}$ from $\mathbf{v}$ by Eq.(9a) and $\mathbf{v}$ from $\mathbf{u}$ by Eq.(9b) in an alternative manner, starting from some initial values of $\mathbf{u}^{(0)}$ or $\mathbf{v}^{(0)}$. An interpretation of the algorithm is that it searches for a fixed point of the mapping $(\mathbf{u}, \mathbf{v})$ to $(\hat{\mathbf{u}}, \hat{\mathbf{v}})$ by the iteration. Although ϕ decreases monotonically with the iterations, there is no guarantee as to the speed of its convergence. In fact, as is pointed out in [2], it tends to be quite slow, especially for "badly conditioned" data such as those with there are many missing components and strong noise.

3.2 Naive Gauss-Newton algorithm

Before going into the Wiberg algorithm, for the sake of comparison, we summarize the standard Gauss-Newton algorithm applied to the problem. Defining $\mathbf{x} \equiv [\mathbf{u}^\top, \mathbf{v}^\top]^\top$, we write ϕ as

$$\phi(\mathbf{x}) = \frac{1}{2}\mathbf{f}^\top\mathbf{f}, \tag{10}$$

where $\mathbf{f} \equiv \mathbf{F}\mathbf{u} - \mathbf{y} = \mathbf{G}\mathbf{v} - \mathbf{y}$. In order to find a solution $\mathbf{x}$ to $d\phi/d\mathbf{x} = \mathbf{0}$, the Newton's algorithm seeks a solution by iteratively updating $\mathbf{x}$ as $\mathbf{x} + \Delta\mathbf{x} \rightarrow \mathbf{x}$ where the update is computed as a solution to $(d\phi/\mathbf{x}) + (d^2\phi/d\mathbf{x}^2)\Delta\mathbf{x} = 0$. Using $\mathbf{f}$, the first and second derivatives are represented as $d\phi/d\mathbf{x} = (d\mathbf{f}/d\mathbf{x})^\top\mathbf{f}$ and $d^2\phi/d\mathbf{x}^2 = (d\mathbf{f}/d\mathbf{x})^\top(d\mathbf{f}/d\mathbf{x}) + (d^2\mathbf{f}/d\mathbf{x}^2)^\top\mathbf{f}$. In the Gauss-Newton algorithm, the second term $(d^2\mathbf{f}/d\mathbf{x}^2)^\top\mathbf{f}$ is neglected. Then, the equation for the update $\Delta\mathbf{x}$ turns to $\left(\frac{d\mathbf{f}}{d\mathbf{x}}\right)^\top \mathbf{f} + \left(\frac{d\mathbf{f}}{d\mathbf{x}}\right)^\top \left(\frac{d\mathbf{f}}{d\mathbf{x}}\right)\Delta\mathbf{x} = 0$, or equivalently

$$\left|\mathbf{f} + \frac{d\mathbf{f}}{d\mathbf{x}}\Delta\mathbf{x}\right|^2 \rightarrow \min. \tag{11}$$

Note that the popular Levenberg-Marquardt (LM) algorithm, which is widely used in the literature of SFM bundle adjustment and will be later compared with the Wiberg algorithm, is a composite method of the (Gauss-)Newton and steepest descent algorithms.

3.3 Derivation of the Wiberg algorithm

In some of the nonlinear least squares problems with multiple parameters, when assuming part of the parameters to be fixed, minimization of the least squares with respect to the rest of the parameters becomes a simple problem, such as a linear problem, and gives a closed form solution. For such problems, by eliminating the latter parameters, the original minimization problem can be rewritten into a minimization problem of a function only of the former parameters (i.e., those assumed to be fixed). There are some cases where deriving a Newton-based algorithm for the rewritten problem achieves better algorithms in terms of computational complexity etc., than deriving one for the original problem. A general framework of this methodology is shown by Ruhe and Wedin [8]. Wiberg applied this framework to the specific problem, factorization of a matrix with missing components [11].

Thus, the basic idea is to rewrite the minimization problem of $\phi(\mathbf{u}, \mathbf{v})$ into that of a function $\psi(\mathbf{v})$ of only $\mathbf{v}$ using (half of) the bilinearity of Eq.(8). For a fixed $\mathbf{v}$, an optimal $\mathbf{u}$ can be linearly computed according to Eq.(9a). We may represent this by a function $\hat{\mathbf{u}}(\mathbf{v})$. By substituting this into $\phi(\mathbf{u}, \mathbf{v})$, we have

$$\psi(\mathbf{v}) \equiv \phi(\hat{\mathbf{u}}(\mathbf{v}), \mathbf{v}). \tag{12}$$

It is clear that this new minimization problem yields the same solution as the original, since $\mathbf{v}$ minimizing $\psi(\mathbf{v})$ together with $\mathbf{u} = \hat{\mathbf{u}}(\mathbf{v})$ minimizes $\phi(\mathbf{u}, \mathbf{v})$. Then, the application of the standard Gauss-Newton algorithm to solve this minimization of ψ, yields what we call the Wiberg algorithm.

Because of the structure of its square-sums, $\psi(\mathbf{v})$ can be written as

$$\psi(\mathbf{v}) = \frac{1}{2}\mathbf{g}^\top \mathbf{g}, \tag{13}$$

where $\mathbf{g} = \mathbf{g}(\mathbf{v}) = \mathbf{f}(\hat{\mathbf{u}}(\mathbf{v}), \mathbf{v}) = \mathbf{F}\hat{\mathbf{u}}(\mathbf{v}) - \mathbf{y}$. Then we want to solve an equation $d\psi(\mathbf{v})/d\mathbf{v} = 0$. An updating scheme of the Newton method for this equation is immediately given as

$$\left| \mathbf{g} + \frac{d\mathbf{g}}{d\mathbf{v}}\Delta\mathbf{v} \right|^2 \to \min. \tag{14}$$

The $\Delta\mathbf{v}$ minimizing this gives the optimal update toward a local minimum. Thus, we now want to calculate the function on the left hand side of (14). Since $\mathbf{g} = \mathbf{f}(\hat{\mathbf{u}}(\mathbf{v}), \mathbf{v})$, its first-order derivative is given according to the chain rule as

$$\frac{d\mathbf{g}}{d\mathbf{v}} = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}\frac{d\hat{\mathbf{u}}}{d\mathbf{v}} + \frac{\partial \mathbf{f}}{\partial \mathbf{v}}. \tag{15}$$

The partial derivatives of $\mathbf{f}$ can be written using $\mathbf{f} = \mathbf{F}\mathbf{u} - \mathbf{y}$ etc. as

$$\frac{\partial \mathbf{f}}{\partial \mathbf{u}} = \mathbf{F}, \qquad \frac{\partial \mathbf{f}}{\partial \mathbf{v}} = \mathbf{G}. \tag{16}$$

The remaining derivative $d\hat{\mathbf{u}}/d\mathbf{v}$ can be calculated as follows. Since $\phi_{\mathbf{u}}(\hat{\mathbf{u}}(\mathbf{v}), \mathbf{v}) \equiv 0$ independently of $\mathbf{v}$, its derivative with respect to $\mathbf{v}$ should also be zero. From $\phi_{\mathbf{u}}(\hat{\mathbf{u}}(\mathbf{v}), \mathbf{v}) = \mathbf{F}^\top(\mathbf{F}\hat{\mathbf{u}} - \mathbf{y}) = \mathbf{F}^\top\mathbf{g}$, we have $d(\mathbf{F}^\top\mathbf{g})/d\mathbf{v} = 0$. By neglecting the term $(d\mathbf{F}/d\mathbf{v})^\top\mathbf{g}$ in the spirit of the Gauss-Newton algorithm, it is written as

$$\frac{d}{d\mathbf{v}}(\mathbf{F}^\top\mathbf{g}) \approx \mathbf{F}^\top\frac{d\mathbf{g}}{d\mathbf{v}} = \mathbf{F}^\top\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\frac{d\hat{\mathbf{u}}}{d\mathbf{v}} + \frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right). \tag{17}$$

Since this should be zero, we have

$$\frac{d\hat{\mathbf{u}}}{d\mathbf{v}} = -(\mathbf{F}^\top\mathbf{F})^{-1}\mathbf{F}^\top\mathbf{G}. \tag{18}$$

Using this along with Eq.(16), Eq.(15) is rewritten as

$$\frac{d\mathbf{g}}{d\mathbf{v}} = (\mathbf{I} - \mathbf{F}(\mathbf{F}^\top\mathbf{F})^{-1}\mathbf{F}^\top)\mathbf{G}. \tag{19}$$

Let $\mathbf{Q_F} \equiv \mathbf{I} - \mathbf{F}(\mathbf{F}^\top\mathbf{F})^{-1}\mathbf{F}^\top$. Note that $\mathbf{Q_F}$ is a projector to the space orthogonal to the column space of $\mathbf{F}$. Using $\mathbf{Q_F}$, $\mathbf{g}$ is rewritten as $\mathbf{g} = \mathbf{F}\hat{\mathbf{u}}(\mathbf{v}) - \mathbf{y} = -\mathbf{Q_F}\mathbf{y}$. The substitution of this and Eq.(19) into Eq.(14) yields

$$|\,\mathbf{Q_F}\mathbf{G}\Delta\mathbf{v} - \mathbf{Q_F}\mathbf{y}\,|^2 \to \min. \tag{20}$$

Now, the optimal update $\Delta\mathbf{v}$ is determined from this. However, this minimization does not have a unique solution, since $\mathbf{Q_F}\mathbf{G}$ is always singular (not of full rank), as shown in the next section, although this is not mentioned in the Wiberg paper [11].

3.4 Determination of the Gauss-Newton update

We first show that the following holds for the rank of $\mathbf{Q_F}\mathbf{G}$.

Proposition 1. When $\mathbf{U}$, $\mathbf{V}$, $\mathbf{F}$ and $\mathbf{G}$ are all of full rank, the rank of the matrix $\mathbf{Q_F}\mathbf{G}$ is at most $(n - r)r$.

Proof. For any arbitrary $r \times r$ matrix, it always holds that $\mathbf{U}'\mathbf{V}^\top = (\mathbf{U}\mathbf{A})\mathbf{V}^\top = \mathbf{U}(\mathbf{V}\mathbf{A}^\top)^\top = \mathbf{U}\mathbf{V}'^\top$. This equality can be rewritten as

$$\mathbf{F}\mathrm{diag}_m(\mathbf{A}^\top)\mathbf{u} = \mathbf{G}\mathrm{diag}_n(\mathbf{A})\mathbf{v}, \tag{21}$$

where $\mathrm{diag}_m(\mathbf{A})$ represents an m-block diagonal matrix with the diagonal subblock A. Defining an $mr \times r^2$ matrix $\mathbf{X_u}$ and an $nr \times r^2$ matrix $\mathbf{X_v}$ appropriately, the equation is further rewritten as

$$\mathbf{F}\mathbf{X_u}\mathbf{a} = \mathbf{G}\mathbf{X_v}\mathbf{a}, \tag{22}$$

where $\mathbf{a}$ is an r^2-vector containing the components of $\mathbf{A}$. The matrix $\mathbf{X_u}$ consists only of $\mathbf{u}_1, \ldots, \mathbf{u}_m$ and the matrix $\mathbf{X_v}$ only of $\mathbf{v}_1, \ldots, \mathbf{v}_n$. Since Eq.(22) always holds for any arbitrary $\mathbf{a}$, we have $\mathbf{F}\mathbf{X_u} = \mathbf{G}\mathbf{X_v}$.

Let $\mathbf{C}(\mathbf{u}, \mathbf{v}) \equiv \mathbf{F}\mathbf{X}_{\mathbf{u}} = \mathbf{G}\mathbf{X}_{\mathbf{v}}$. From the definition, the column space of $\mathbf{C}$ is always a subspace of $\mathbf{F}$ and $\mathbf{G}$. If $\mathbf{V}$ is of full rank, it is easy to see that $\mathbf{X}_{\mathbf{v}}$ is of full rank ($\mathbf{X}_{\mathbf{v}}^{\top}\mathbf{X}_{\mathbf{v}} = \mathrm{diag}_r(\mathbf{V}^{\top}\mathbf{V})$). Thus, if $\mathbf{G}$ is also of full rank, $\mathbf{C} = \mathbf{G}\mathbf{X}_{\mathbf{v}}$ is of full rank, too. Then, the column space of $\mathbf{C}$ has the dimension of r^2. Thus, it has been shown that the column spaces of $\mathbf{F}$ and $\mathbf{G}$ share a subspace of at least r^2 dimension.

Since $\mathbf{Q}_{\mathbf{F}}$ is a projector to the space orthogonal to the column space of $\mathbf{F}$, the rank of $\mathbf{Q}_{\mathbf{F}}\mathbf{G}$ is given as the number nr of its columns minus the dimension of the common subspace of the column spaces of $\mathbf{F}$ and $\mathbf{G}$. Thus, we have shown that $\mathrm{rank}(\mathbf{Q}_{\mathbf{F}}\mathbf{G}) \leq nr - r^2 = (n-r)r$. □

This result is clearly connected to the fact that factorization $\mathbf{Y} \to \mathbf{U}\mathbf{V}^{\top}$ is always not unique; for any arbitrary non-singular $r \times r$ matrix $\mathbf{A}$, it is possible to rewrite $\mathbf{U}\mathbf{V}^{\top} = \mathbf{U}'\mathbf{V}'^{\top}$ by setting $\mathbf{U}' = \mathbf{U}\mathbf{A}^{-1}$ and $\mathbf{V} = \mathbf{V}\mathbf{A}^{\top}$. The degree of freedom of the ambiguity is r^2, the number of components of $\mathbf{A}$.

In the above proof, $\mathbf{U}$, $\mathbf{V}$, $\mathbf{F}$, and $\mathbf{G}$ are all assumed to be of full rank. This does not limit the applicability of the result for the following reasons. Firstly, the matrices $\mathbf{U}$ and $\mathbf{V}$ should be of full rank to make factorization meaningful. Also, $\mathbf{F}$ and $\mathbf{G}$ should be of full rank, too, since if they are not, the factorization is not unique. Thus, the assumption should hold for non-degenerate, valid data, from which meaningful solutions can be derived.

Since $\mathbf{Q}_{\mathbf{F}}\mathbf{G}$ is always not of full rank, the equation (20) for the update has an infinite number of solutions. A promising choice is

$$\Delta\mathbf{v} = (\mathbf{Q}_{\mathbf{F}}\mathbf{G})^{\dagger}\mathbf{Q}_{\mathbf{F}}\mathbf{y}, \tag{23}$$

where $(\mathbf{Q}_{\mathbf{F}}\mathbf{G})^{\dagger}$ is a generalized inverse of $\mathbf{Q}_{\mathbf{F}}\mathbf{G}$. This solution corresponds to choosing $\Delta\mathbf{v}$ that minimizes $|\Delta\mathbf{v}|^2$. Representing the singular value decomposition of $\mathbf{Q}_{\mathbf{F}}\mathbf{G}$ as $\mathbf{Q}_{\mathbf{F}}\mathbf{G} \to \mathbf{S}\mathbf{D}\mathbf{T}^{\top}$, $(\mathbf{Q}_{\mathbf{F}}\mathbf{G})^{\dagger}$ is given as

$$(\mathbf{Q}_{\mathbf{F}}\mathbf{G})^{\dagger} \equiv \mathbf{T}\widetilde{\mathbf{D}}^{-1}\mathbf{S}^{\top}, \tag{24}$$

where $\widetilde{\mathbf{D}}^{-1}$ is defined as $\widetilde{\mathbf{D}}^{-1} = \mathrm{diag}[1/d_1, \ldots, 1/d_q, 0, \ldots, 0]$, where $d_1, \ldots, d_q$ are non-zero singular values. The algorithm is summarized as Fig.1.

As shown above, the rank of $\mathbf{Q}_{\mathbf{F}}\mathbf{G}$ is equal to or smaller than $(n-r)r$. However, it can be shown, from the result of one of our studies, that in order for factorization to be unique, the rank should not be smaller than $(n-r)r$ and should namely be exactly $(n-r)r$. Therefore, when computing

1. Initialize $\mathbf{v}$.
2. Make $\mathbf{F}$ from $\mathbf{v}$ and compute $\mathbf{u}$ that minimizes $|\mathbf{F}\mathbf{u} - \mathbf{y}|^2$.
3. Stop if converged. Otherwise go to 4.
4. Make $\mathbf{G}$ from $\mathbf{u}$ and determine $\Delta\mathbf{v}$ that minimizes $|\mathbf{Q_F}\mathbf{G}\Delta\mathbf{v} - \mathbf{Q_F}\mathbf{y}|^2$. From the possible solutions, select that with minimum length $|\Delta\mathbf{v}|$. Update $\mathbf{v} + \Delta\mathbf{v} \rightarrow \mathbf{v}$. Go to 2.

Figure 1: The Wiberg algorithm for the factorization form without a mean vector.

$(\mathbf{Q_F}\mathbf{G})^\dagger$ on Eq.(24), we need only select the $q = (n - r)r$ largest singular values. This enables a secure implementation of the algorithm in which the numerical difficulty of identifying non-zero singular values from a numerical result of the SVD is avoided.

For the factorization form (3) with a mean vector, instead of Proposition 1, we show:

Proposition 2. *The rank of the matrix* $\mathbf{Q_F}\tilde{\mathbf{G}}$ *is at most* $(n - r)(r + 1)$.

The proof is omitted here. Figure 2 shows the algorithm for the case with a mean vector. As in the above, it can be conveniently used in its implementation that the rank of $\mathbf{Q_F}\tilde{\mathbf{G}}$ is expected to be exactly $(n - r)(r + 1)$, unless the given data are not ill-conditioned.

3.5 Consideration of computational cost

In the Wiberg algorithm, there are two linear equations to be solved per iteration. As for the equation in step 2 for updating $\mathbf{u}$, we should use the block property of the matrix to reduce its computational cost; $\mathbf{F}$ has m block submatrices of at most $n \times r$ size. Then, assuming the computational cost of solving a linear equation with a $M \times N$ coefficient matrix to be $O(MN^2)$, the cost is evaluated as $O(mnr^2)$.

As for the equation in step 4, there is no way of reducing the computational cost, and thus it is evaluated as $O(pn^2r^2)$, since $\mathbf{Q_F}\mathbf{G}$ is of $p \times nr$. Since this is much larger than the cost $O(mnr^2)$ of

1. Initialize $\tilde{\mathbf{v}}$, i.e., $\mathbf{V}$ and $\mathbf{m}$.
2. Make $\mathbf{F}$ and $\mathbf{m}$ from $\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_n$ and compute $\mathbf{u}$ that minimizes $|\mathbf{F}\mathbf{u} + \mathbf{m} - \mathbf{y}|^2$.
3. Stop if converged. Otherwise go to 4.
4. Make $\tilde{\mathbf{G}}$ from $\mathbf{u}$ and determine $\Delta\tilde{\mathbf{v}}$ that minimizes $|\mathbf{Q}_\mathbf{F}\tilde{\mathbf{G}}\Delta\tilde{\mathbf{v}} - \mathbf{Q}_\mathbf{F}(\mathbf{y} - \mathbf{m})|^2$ From the possible solutions, select that with minimum length $|\Delta\tilde{\mathbf{v}}|$. Update $\tilde{\mathbf{v}} + \Delta\tilde{\mathbf{v}} \rightarrow \tilde{\mathbf{v}}$. Go to 2.

Figure 2: The Wiberg algorithm for the factorization form with a mean vector.

step 2, step 4 is the dominant part in terms of computational cost. Thus, the overall computational cost can be evaluated as $O(pn^2r^2)$ per iteration. Note that it depends on n, the number of columns of $\mathbf{Y}$, but not on m, the number of rows of $\mathbf{Y}$. Therefore, the better one between factorizing $\mathbf{Y}$ and $\mathbf{Y}^\top$ should be chosen to minimize the computational cost, whenever the choice is possible.

The computational cost for the naive Gauss-Newton algorithm that was shown earlier can similarly be evaluated as $O(p(m + n)^2r^2)$. Thus, it can be seen that the Wiberg algorithm is faster than the naive Gauss-Newton algorithm, and the ratio of the costs is given as $n^2/(m + n)^2$. Thus, if $m \approx n$, it will be 1/4 and it will further decrease when m is larger than n.

In the ALS algorithm, there are two equations to solve, whose computational cost are comparable to the solution of the equation in step 2 of the Wiberg algorithm. Thus, as for the ALS algorithm, its computational cost per iteration is by far smaller than even the Wiberg algorithm. However, it should be noted that the ALS algorithm tends to need many iterations to converge, hence the total computational time tends to be rather larger than the other two, especially for badly conditioned data.

4 Experimental results

This section presents the experimental results. Only the factorization with a mean vector is considered here; in our experience, there is little difference in numerical behavior between the algorithms for the two factorization forms.

4.1 Comparison with the Levenberg-Marquardt algorithm

We compared the described implementation of the Wiberg and Levenberg-Marquardt (LM) algorithms. Comparisons of LM with the ALS algorithm and its variants are detailed in [2]. As an implementation of the LM algorithm, the function `lmder` from the MINPACK library is used. There is the following fundamental ambiguity in the factorization: $\mathbf{U}\mathbf{V}^\top + \mathbf{1}_m\mu^\top = (\mathbf{U}\mathbf{A}^{-1} + \mathbf{1}_m\mathbf{b}^\top)(\mathbf{V}\mathbf{A}^\top)^\top + \mathbf{1}_m(\mu^\top - \mathbf{b}^\top\mathbf{A}^\top\mathbf{V})$. In order to constrain this, the necessary number of components of $\mathbf{U}$ and $\mathbf{m}$ are fixed in the implementation of the LM algorithm.

4.1.1 Synthetic data

As test data, matrices of 30×20 (i.e. $m = 30$ and $n = 20$) with rank $r = 3$ are used. Each component is randomly generated according to $y_{ij} = \mathbf{u}_i^\top\mathbf{v}_j + \mu_j + \varepsilon_{ij}$, where u_{ij}, v_{ij}, and μ_j are all random variables generated according to a normal density $N(0,1)$, and the noise ε according to $N(0, 0.05^2)$ (i.e. 5% noise corruption). The missing components are also randomly chosen in the 30×20 matrix. The two algorithms are run for these data. The simulation repeats for 500 trials, and during each trial, initial values are randomly chosen and supplied to each of the two algorithms.

Figures 3 and 4 show the results for the data with 30% and 65% missing components, respectively. The upper row shows the histograms of iteration counts for each algorithm, and the lower row shows the histogram of the residue $\sum(y_{ij} - \mathbf{u}_i\mathbf{v}_j - \mu_j)^2$. The two algorithmsx are forced to stop when the iteration count exceeds 100. It can be seen from Fig.3 that the Wiberg algorithm converges in every trial, whereas the LM algorithm does not converge (at most within 100 iteration counts) in 10% of the trials. The histograms of the residue show that both algorithms converge to a global minimum whenever they converge. Figure 4 shows the results for 65% missing components. As in the case of 30% missing components, it can be seen that the Wiberg algorithm

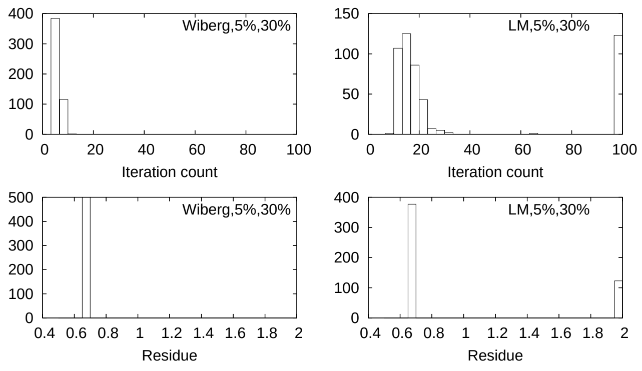

Figure 3: Results for synthetic data with 30% missing components. Upper: The iteration counts for 500 trials with random initial values. Lower: The residue after convergence. Left: Wiberg. Right: Levenberg-Marquardt.

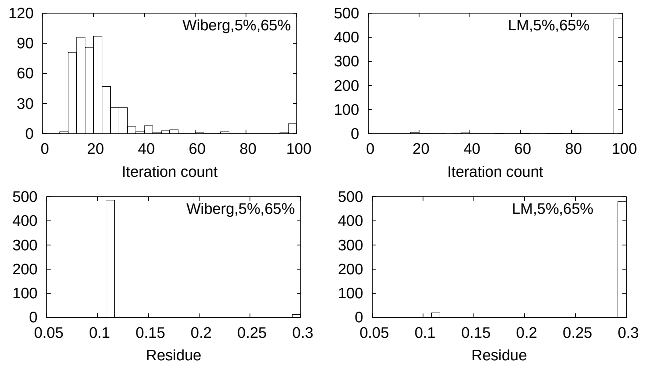

Figure 4: Results for synthetic data with 65% missing components.

converges in almost every trial, whereas the LM algorithm does so in only a few. We can conclude that the Wiberg algorithm showed a much better performance than the LM algorithm for the data used here.

4.1.2 Real data

We also compared the two algorithms using a real image sequence of a cube rotated through 360 degrees on a turntable. There are 177 images in the sequence, a few of which are shown in Fig.5. The Lucas-Kanade-Tomasi tracker is first applied to the image sequence, and trajectories of feature points are extracted, from which wrong trajectories are manually removed. A few erroneous trajectories are intentionally left unremoved to test the robustness of the algorithms. As a result, 106 trajectories survive, which are shown on the left of Fig.6. The number of observed data is 5550,

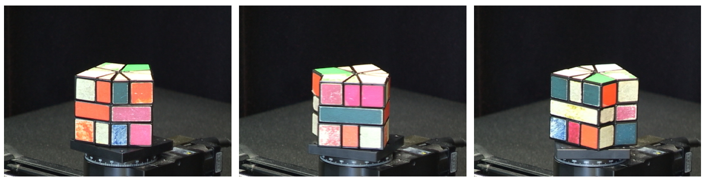

Figure 5: Real images used for the test. Selected three of 177 images.

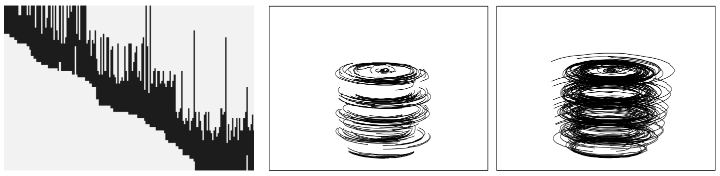

Figure 6: Trajectories of feature points. Left: Observed components (in black) in the data matrix. Middle: Initial trajectories. Right: Recovered trajectories.

which means that 70.4%$(= (106 \times 177) - 5550/(106 \times 177))$ of components are missing.

Then, the LM and the Wiberg algorithms are run for 100 trials. In the same way as above, random initial values are used for each trial. Figure 7 shows the results. It is seen that the Wiberg algorithm converged to an identical solution, which is therefore considered to be a correct solution, in every trial. On the other hand, *the LM algorithm did not converge in any of the 100 trials*. To examine the behavior of the LM algorithm, we run the LM algorithm also using *good* initial values, which are synthesized by multiplying $(1 + \alpha)$ to each component of true $\mathbf{U}$ and $\mathbf{V}$, where $\alpha \sim N(0, \sigma_1^2)$. The second and third histograms on Fig.7 show the results for $\sigma_1 = 0.1$ and 0.3, respectively. It can be seen from this that when good initial values are given, the LM algorithm converges, and also that its convergence performance deteriorates quickly as the initial values become distant from the global minimum. We can conclude that the Wiberg algorithm outperforms the LM algorithm for the real data used here.

4.2 Convergence performance vs. fraction of missing data and noise strength

Clearly there are two factors that affect the convergence performance of the Wiberg algorithm: the fraction of missing components and noise strength. Using synthetic data, we examined how these factors affect the convergence performance. The data is generated in the same way as above, except that the matrix size is set to 50×50, and the missing components are deterministically selected so

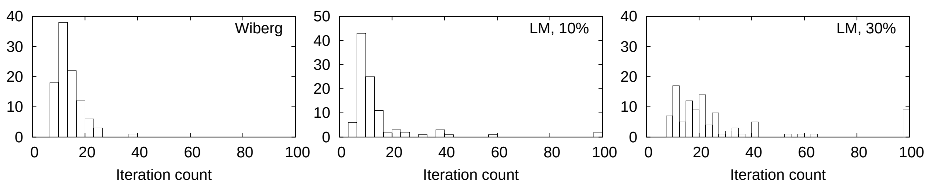

Figure 7: The number of iterations needed for SFM using the trajectories of Fig.6. From left to right, the Wiberg algorithm starting from random initial values, and the LM algorithm starting from "good" initial values that are generated by perturbing the solutions obtained by the Wiberg with 10% and 30% random noise, respectively.

that **H** is a band diagonal matrix. The bandwidth of **H** varies from 29 to 13, which corresponds to a range from 50.4% to 75.7% of the fraction of the missing components. The variance of the noise ε is varied in the range from 0.005(0.5% corruption) to 0.5(50% corruption). Then, we run the algorithm for 100 trials, for each of which random initial values are used.

Figure 8 shows the results. The left plot shows the percentage of the number of *successful* trials vs. the fraction of missing components, for each noise strength. The success/failure of a trial is identified by checking if the final residue after the convergence is the same, in a numerical sense, as the minimum of the residues for the 100 trials. The minimum of the residues for each data set is shown on the right plot, which confirms that the algorithm actually reaches the global minimum for the trials identified as successful.

It can be seen from the results that the convergence rate decreases with the number of missing components, and deteriorates very quickly over around the fraction 65% of missing components. The "theoretical ceiling" of the missing component fraction, which is given at the percentage where the number of observed data is the same as the number of parameters minus factorization ambiguity (i.e., $p = mr + (n - r)(r + 1)$), is calculated to be 86.5%. Thus, a gap is evident between the theoretical ceiling and the percentage at which the algorithm starts to fail to converge. It can also be seen that noise strength does not appear to significantly affect the convergence performance. It seems even that the number of converged trials slightly increases for larger noise strength. It should be noted that for most of the unconverged trials, the algorithm either diverged completely or was trapped in an infinite (oscillatory) loop, and only for a few trials, did it converge to another minimum.

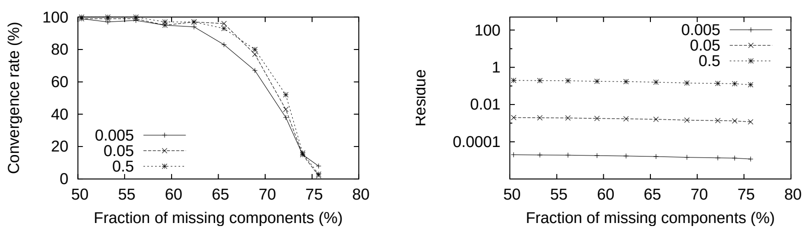

Figure 8: Effects of fraction of missing components and noise strength (0.5, 5, and 50% noise corruption) on the convergence performance. Left: The number of trials (%) for which the algorithm converges to the solution of minimum residue. Right: The minimum residue per y_{ij}.

5 Conclusion

We have shown the derivation of the Wiberg algorithm for the problem of matrix factorization in the presence of missing components. In the derivation, we prove the degeneracy of the equation for determining the Gauss-Newton update in the algorithm, which needs to be taken care of when implementing the algorithm. We show through several experiments, that our implementation of the algorithm demonstrates a relatively good convergence performance, as compared to a standard implementation of the Levenberg-Marquardt algorithm. As inferior performance of the ALS algorithm and its variants, even to the LM algorithm, is reported in [2], we believe that the Wiberg algorithm (with the described implementation) should also be used as a standard tool for the problems in computer vision.

References

[1] P. N. Belhumeur and D. J. Kriegman. What is the set of images of an object under all possible illumination conditions ? *International Journal of Computer Vision*, 28(3):245–260, 1998.

[2] A. M. Buchanan and A. W. Fitzgibbon. Damped newton algorithms for matrix factorization with missing data. In *Proceedings of IEEE Computer Vision and Pattern Recognition*, 2005.

[3] A. M. Buchanan. Investigation into matrix factorization when elements are unknown. Technical report, University of Oxford, `http://www.robots.ox.ac.uk/˜amb`, 2004.

[4] P. Chen and D. Suter. Recovering the missing components in a large noisy low-rank matrix: Application to sfm. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 26(8):1051–1063, 2004.

[5] R. Epstein, A. L. Yuille, and P. N. Belhumeur. Learning object representations from lighting variations. In *Object Representation in Computer Vision II*. Eds. J. Ponce, A. Zisserman, and M. Herbert. Springer Lecture Notes in Computer Science 1144, 179–199, 1996.

[6] H. Hayakawa. Photometric stereo under a light source with arbitrary motion. *Journal of the Optical Society of America A*, 11(11):3079–3089, 1994.

[7] D. W. Jacobs. Linear fitting with missing data for structure-from-motion. *Computer Vision and Image Understanding*, 82(1):57–81, 2001.

[8] A. Ruhe and P. ÅA. Wedin. Algorithms for separable nonlinear least squares problems. *"SIAM Review"*, 22(3):318–337, 1980.

[9] H. Shum, K. Ikeuchi, and R. Reddy. Principal component analysis with missing data and its application to polyhedral object modeling. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 17(9):855–867, 1995.

[10] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[11] T. Wiberg. Computation of principal components when data are missing. In *Proceedings of the Second Symposium of Computational Statistics*, Berlin, 229–326, 1976.