A Simultaneous Reconstruction of Missing Data in DNA Microarrays

Shmuel Friedland [*†], Amir Niknejad [*], and Laura Chihara[‡]
Institute for Mathematics and its Applications
400 Lind Hall, 207 Church St., S.E.
Minneapolis, MN 55455-0436

Final revised version: August 3, 2005

Abstract

We suggest here a new method of the estimation of missing entries in a gene expression matrix, which is done simultaneously— i.e., the estimation of one missing entry influences the estimation of other entries. Our method is closely related to the methods and techniques used for solving inverse eigenvalue problems.

2000 Mathematical Subject Classification: 15A18, 92D10
Keywords: Gene expression matrix, singular value decomposition (svd), missing values, imputation.

1 Introduction

In the last decade, molecular biologists have been using DNA microarrays as a tool for analyzing information in gene expression data. During the laboratory process, some spots on the array may be missing due to various factors (for example, machine error.) Because it is often very costly or time consuming to repeat the experiment, molecular biologists, statisticians, and computer scientists have made attempts to recover the missing gene expressions by some ad-hoc and systematic methods.

More recently, microarray gene expression data have been formulated as a gene expression matrix E with n rows, which correspond to genes, and m columns, which correspond

[*] *Department of Mathematics, Statistics and Computer Science, University of Illinois at Chicago, Chicago, Illinois 60607-7045*
[†] Corresponding author. Tel.: +1-312-996-3041; fax: +1-312-996-1491; email: friedlan@uic.edu
[‡] *Department of Mathematics and Computer Science, Carleton College, One N. College St., Northfield, MN 55057*

to experiments. Typically n is much larger than m. In this setting, the analysis of missing gene expressions on the array would translate to recovering missing entries in the gene expression matrix values.

The most common methods for recovery are [12]:
(a) Zero replacement method;
(b) Row sum mean;
(c) Clustering analysis methods such as K-nearest neighbor clustering , hierarchical clustering;
(d) SVD - Singular Value Decomposition (which is closely related to Principal Component Analysis).

In these methods, the recovery of missing data is done independently, i.e., the estimation of each missing entry does not influence the estimation of the other missing entries. The iterative method using SVD suggested in [12] takes into account implicitly the influence of the estimation of one entry on the other ones. See also [2].

We suggest a new method in which the estimation of missing entries is done simultaneously, i.e., the estimation of one missing entry influences the estimation of the other missing entries. If the gene expression matrix E has missing data, we want to complete its entries to obtain a matrix $\hat{E}$, such that the rank of $\hat{E}$ is equal to (or does not exceed) d, where d is taken to be the number of significant singular values of E. The estimation of the entries of E to a matrix with a prescribed rank is a variation of the *problem of communality* (*see* [4, p. 637].) We give an optimization algorithm for finding $\hat{E}$ using the techniques for inverse eigenvalue problems discussed in [3].

We implemented our fixed rank approximation algorithm as a Matlab procedure and ran simulations on the microarray data *Saccharomyces cerevisiae* [11]. (This data set is the benchmark for icroarray data for other methods of missing value estimations available in the recent literature. It is available on the web address
http://genome-www.stanford.edu/SVD/htmls/spie.html, under the name Elutriation data set.) We describe the results in Section 7.

We ran similar simulations on the full Cdc15 data set, available at the above web address, and on subsets of this data set (using 4 columns). We also ran a couple of simulations on one of the data sets included by [10]. The outcomes were similar to that using the Elutriation data set, with the FRAA algorithm outperforming KNN on the matrices with a small number of columns.

It is likely that our algorithm can be used to estimate missing entries in data sets other than gene expression data. Such a data set should be represented by an $n \times m$ matrix whose rank is smaller than $\min(m, n)$. To keep the paper focused we did not test our methods on non-microarray data sets.

Since we wrote the first version of this paper in Fall 2003 we became aware of [10], which uses Bayesian estimations, and a new paper [9], which use local least squares. Both papers claim to have superior results than KNN. The relatives successes of KNN and these

two methods over FRAA most likely is due to the fact that these three methods use only closely related genes to impute the missing values in each microarray data set. We believe that if we first apply FRAA to the corrupted set, then using this estimated data set, subdivide the genes into clusters of genes with similar traits, and then once again apply FRAA to the missing entries of genes in each cluster, we will obtain similar, or hopefully better results, then the above three methods. We intend to carry out this algorithm in a future paper.

2 The Singular Value Decomposition

In this section, we recall some basic facts about *Singular Value Decomposition* SVD. Let E be an $n \times m$ real-valued nonzero matrix. In this paper we assume that $n \geq m$. The SVD of E is a decomposition of E into the product $U\Sigma V^{\mathrm{T}}$ with certain properties. There are a few variations of this definition, and we give the following one which is most suitable for the applications in our context. We assume that U is $n \times m$, Σ is $m \times m$, and V is $m \times m$.

$$E = U\Sigma V^{\mathrm{T}},\ U^{\mathrm{T}}U = V^{\mathrm{T}}V = I_m,\ \Sigma = \mathrm{diag}(\sigma_1, ..., \sigma_m),\ \sigma_1 \geq ... \geq \sigma_m \geq 0. \quad (2.1)$$

The rank r of E is the number of positive singular values; the dimension of the row space, and the dimension of the column space of E is also r.

Remark. Singular value decomposition is related to principal component analysis (PCA) in statistics. If we center each column in matrix E, then $E^TE = V\Sigma^2V^T$ is proportional to the covariance matrix of the columns of E, the columns of V are the principal components, and the $\{\sigma_q^2\}$ are proportional to the variances of the principal components.

Let U_r, Σ_r, V_r be matrices obtained from U, Σ, V, respectively, as follows: U_r is an $n \times r$ matrix obtained by deleting the last $m - r$ columns of U, V_r is the $m \times r$ matrix obtained by deleting the last $m - r$ columns of V, and Σ_r is obtained by deleting the last $m - r$ columns and rows of Σ . Then

$$E = U_r\Sigma_rV_r^{\mathrm{T}},\ U_r^{\mathrm{T}}U_r = V_r^{\mathrm{T}}V_r = I_r,\ \Sigma_r = \mathrm{diag}(\sigma_1, ..., \sigma_r),\ \sigma_1 \geq ... \geq \sigma_r > 0. \quad (2.2)$$

In this setting U_r, Σ_r, V_r are all rank r matrices: the last $m - r$ columns of U and the last $m - r$ rows of V^{T} are arbitrary, up to the condition that the last $m - r$ columns U and last $m - r$ rows of V^{T} are orthonormal bases of the orthogonal complement of the column space and the row space of E respectively. Hence (2.2) is sometimes called a *reduced* SVD of E.

We now give another form of (2.2) which has a significant interpretation in microarray data. Let $\mathbf{u}_1, \cdots, \mathbf{u}_m$ denote the columns of U and $\mathbf{v}_1, \ldots, \mathbf{v}_m$ denote the columns of V. Then (2.1) and (2.2) can be written as

$$E = \sum_{q=1}^{m} \sigma_q \mathbf{u}_q \mathbf{v}_q^{\mathrm{T}} = \sum_{q=1}^{r} \sigma_q \mathbf{u}_q \mathbf{v}_q^{\mathrm{T}}. \tag{2.3}$$

If $\sigma_1 > ... > \sigma_r$ then $\mathbf{u}_q$ and $\mathbf{v}_q$ are determined up to the sign ± 1 for $q = 1, ..., r$. Namely $\mathbf{u}_q$ and $\mathbf{v}_q$ are length 1 eigenvectors of EE^{T} and $E^{\mathrm{T}}E$, respectively, corresponding to the common eigenvalue σ_q^2. (Note the choice of a sign in $\mathbf{v}_q$ forces a unique choice of the sign in $\mathbf{u}_q$.) The vectors $\mathbf{u}_1, \ldots, \mathbf{u}_r$ are called *eigengenes*, the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_r$ are called *eigenarrays* and $\sigma_1, \ldots, \sigma_r$ are called *eigenexpressions*. The rank r can be viewed as the number of different biological functions of n genes observed in m experiments. The eigenarrays $\mathbf{v}_1, \ldots, \mathbf{v}_r$ give the principle r orthogonal directions in $\mathbb{R}^m$ corresponding to $\sigma_1, \ldots, \sigma_r$. The eigengenes $\mathbf{u}_1, \ldots, \mathbf{u}_r$ give the principle r orthogonal directions in $\mathbb{R}^n$ corresponding to $\sigma_1, \ldots, \sigma_r$. The eigenexpressions describe the relative significance of each bio-function. From the data given in [1], one it seems that the number of significant singular values never exceeds $\frac{m}{2}$. See the discussion on the number of significant singular values in the beginning of §3. The essence of the FRAA algorithm, suggested in this paper, is based on this observation.

Computationally, one brings E to upper bidiagonal form A using Householder matrices. Then one applies implicitly the QR algorithm to $A^{\mathrm{T}}A$ to find the positive eigenvalues $\sigma_1^2, ..., \sigma_r^2$ and the corresponding orthonormal eigenvectors $\mathbf{v}_1, ..., \mathbf{v}_r$ of the matrix $E^{\mathrm{T}}E$ [5]. Next

$$\mathbf{u}_q := \frac{1}{\sigma_q} E\mathbf{v}_q \iff \sigma_q \mathbf{u}_q = E\mathbf{v}_q, \quad q = 1, ..., r. \tag{2.4}$$

To compute the decomposition (2.3), it is enough to know $\mathbf{v}_q$ and $\sigma_q \mathbf{u}_q$. If σ_q repeats $k > 1$ times in the sequence $\sigma_1 \geq ... \geq \sigma_r > 0$, then the choice of the corresponding k eigenvectors $\mathbf{v}_j$ is not unique: any choice of the orthonormal basis in the eigenspace of $E^{\mathrm{T}}E$ corresponding to the eigenvalue σ_q^2 is a legitimate choice.

We remark that in our applications m was relatively small: $m \leq 20$. Thus we opted to compute the "small" matrix $E^{\mathrm{T}}E$ directly, then use software to compute the positive eigenvalues $\sigma_1^2, \ldots, \sigma_r^2$ and the corresponding orthonormal eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_r$ of $E^{\mathrm{T}}E$.

Denote by $||E||_{\mathcal{F}}$ the Frobenius (ℓ_2) norm of E. It is the Euclidean norm of E viewed as a vector with nm coordinates. Each term $\mathbf{u}_q \mathbf{v}_q^{\mathrm{T}}$ in (2.3) is a rank one matrix with $||\mathbf{u}_q \mathbf{v}_q^{\mathrm{T}}||_{\mathcal{F}} = 1$. Let $\mathcal{R}(n, m, k)$ denote the set of $n \times m$ matrices of at most rank k ($m \geq k$). Then for each k, $k \leq r$, the SVD of E gives the solution to the following approximation problem:

$$\min_{F \in \mathcal{R}(n,m,k)} ||E - F||_{\mathcal{F}} = ||E - \sum_{q=1}^{k} \sigma_q \mathbf{u}_q \mathbf{v}_q^{\mathrm{T}}||_{\mathcal{F}} = \sqrt{\sum_{q=k+1}^{r} \sigma_q^2}. \tag{2.5}$$

If $\sigma_k > \sigma_{k+1}$ then $\sum_{q=1}^{k} \sigma_q \mathbf{u}_q \mathbf{v}_q^{\mathrm{T}}$ is the unique solution to the above minima problem. For the purposes of this paper, it will be convenient to assume that $\sigma_q = 0$ for any $q > m$.

In what follows we will use yet another equivalent definition of the singular values of E. Let $\mathbb{R}^{n\times m}$ denote the space of all real $n \times m$ matrices and let $\mathrm{S}_m(\mathbb{R})$ denote the space of all real $m \times m$ symmetric matrices. For $A \in \mathrm{S}_m(\mathbb{R})$, we let

$$\lambda_1(A) = \lambda_1 \geq ... \geq \lambda_m(A) = \lambda_m,\ A\mathbf{z}_q = \lambda_q \mathbf{z}_q,\ \mathbf{z}_q^{\mathrm{T}}\mathbf{z}_t = \delta_{qt},\ q, t = 1, ..., m, \tag{2.6}$$

be the eigenvalues and corresponding eigenvectors of A, where the eigenvalues are counted with their multiplicities, and the eigenvectors form an orthonormal basis in $\mathbb{R}^m$.

Consider the following $(n + m) \times (n + m)$ real symmetric matrix:

$$E^s := \begin{pmatrix} 0 & E \\ E^{\mathrm{T}} & 0 \end{pmatrix}. \tag{2.7}$$

It is known [6, §7.3.7]

$$\begin{aligned} &\sigma_q(E) := \sigma_q = \lambda_q(E^s) = -\lambda_{n+m+1-q}(E^s),\ \text{for } q = 1, ..., m, \\ &\lambda_q(E^s) = 0 \text{ for } q = m + 1, ..., n. \end{aligned} \tag{2.8}$$

The Cauchy interlacing property for E^s implies [6, §7.3.9]

Let $[n] := \{1, 2, \ldots, n\}$, and let $\mathcal{N} \subset [n],\ \mathcal{M} \subset [m]$ denote sets of cardinalities $n - n', m - m' \geq 0$ respectively.

Proposition 2.1 *Let $E \in \mathbb{R}^{n\times m}$ and denote by $E' \in \mathbb{R}^{n'\times m'}$ the matrix obtained from E by deleting all rows $i \in \mathcal{N}$ and all columns $j \in \mathcal{M}$. Then*

$$\begin{aligned} &\sigma_q(E) \geq \sigma_q(E') \text{ for } q = 1, ..., m, \\ &\sigma_q(E') \geq \sigma_{q+n-n'+m-m'}(E) \text{ for } q = 1, ..., m' + n' - n. \end{aligned} \tag{2.9}$$

The significance of this proposition is explained in §4 and §5.

3 The Gene Expression Matrix

In this section we will view $E \in \mathbb{R}^{n\times m}$, with $n \geq m$ as the gene expression matrix:

$$E = \begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1m} \\ g_{21} & g_{22} & \cdots & g_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ g_{j1} & g_{j2} & \cdots & g_{jm} \\ \vdots & \vdots & \vdots & \vdots \\ g_{n1} & g_{n2} & \cdots & g_{nm} \end{pmatrix} = \begin{pmatrix} \mathbf{g}_1^{\mathrm{T}} \\ \mathbf{g}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{g}_j^{\mathrm{T}} \\ \vdots \\ \mathbf{g}_n^{\mathrm{T}} \end{pmatrix} = [\, \mathbf{c}_1 \quad \mathbf{c}_2 \quad \ldots \mathbf{c}_m \,], \tag{3.1}$$

$$\mathbf{g}_j^{\mathrm{T}} := (g_{j1}, g_{j2}, ..., g_{jm}),\ j = 1, ..., n, \quad \mathbf{c}_i = \begin{pmatrix} g_{1i} \\ g_{2i} \\ \vdots \\ g_{ji} \\ \vdots \\ g_{ni} \end{pmatrix},\ i = 1, ..., m.$$

The row vector $\mathbf{g}_j^{\mathrm{T}}$ corresponds to the (relative) expression levels of the j^{th} gene in m experiments. The column vector $\mathbf{c}_i$ corresponds to the (relative) expression levels of the n genes in the i^{th} experiment.

Consider the SVD of the gene expression matrix $E = U\Sigma V^T$. In the terminology of [1], the columns of U are eigengenes, the columns of V are eigenarrays, and the singular values of E are eigenexpression levels.

In many microarray data sets, researchers have found that only a few eigengenes are needed to capture the overall gene expression pattern. (Here, by a "few" we mean less than half of the number of experiments m.) The number of these *significant* eigengenes is a fundamental problem in principal component analysis [7]. Let us mention explicitly three methods to estimate the number of significant eigengenes. The *fraction* criteria can be stated simply as follows. Let

$$p_q := \frac{\sigma_q^2}{\sum_{t=1}^m \sigma_t^2},\ q = 1, ..., m, \quad \mathbf{p} := (p_1, ..., p_m)^{\mathrm{T}}. \tag{3.2}$$

Thus p_q represents the fraction of the expression level contributed by the q^{th} eigengene. Then we choose the l eigengenes that contribute about $70\% - 90\%$ of the total expression level. Another method is to use scree plots for the σ_q^2. (In principal component analysis, the p_q are proportional to the variances of the principal components, so we choose the principal components of maximum variability [8].) According to [7], the most consistent estimates of the number of significant eigengenes is achieved by the broken-stick model.

If E has l significant eigenvalues, we view σ_q to be effectively equal to zero for $q > l$. We define the matrix

$$E_l := \sum_{q=1}^{l} \sigma_q \mathbf{u}_q \mathbf{v}_q^{\mathrm{T}} \tag{3.3}$$

as the *filtered* part of E and consider $E - E_l$ the *noise* part of E.

Let

$$1 \geq h(\mathbf{p}) := -\frac{1}{\log m} \sum_{q=1}^{m} p_q \log p_q \geq 0. \tag{3.4}$$

Then $h(\mathbf{p})$ is the rescaled entropy of the probability vector $\mathbf{p}$. $h(\mathbf{p}) = 1$ only when $p_q = \frac{1}{m},\ q = 1, ..., m$; in other words, all the eigengenes are equally expressed. On the other hand, $h(\mathbf{p}) = 0$ if and only if $p_q(1 - p_q) = 0,\ q = 1, ..., m$ and this corresponds to $r = 1$: in other words, the gene expression is captured by a single eigengene (and eigenarray).

The following example points out a potential weakness of SVD theory in trying to detect groups of genes with similar properties.

3.1 SVD and gene clusters

Suppose the set of genes $\mathbf{g}_j^{\mathrm{T}},\ j \in [n]$ can be grouped into $k + 1$ disjoint subsets $[n] = \cup_{q=1}^{k+1} G_q$ with $G_1, ..., G_k$ nonempty and $m \geq k$ (usually $m > k$). In particular, consider the genes in each group G_q ($q = 1, ..., k$) to have similar characteristics (in other words, G_q is a cluster). Genes that have no similar characteristics are placed in G_{k+1}. Denote by $\#G_q$ the cardinality of the set G_q for $q = 1, ..., k + 1$. Suppose that our m experiments does not distinguish between any two genes belonging to the same group G_q for $q = 1, ..., k + 1$. More precisely we assume:

$$\begin{aligned} &g_{ji} = a_{qi} \text{ for each } j \in G_q \text{ and } q = 1, ..., k,\ i = 1, ..., m, \\ &g_{ji} = 0 \text{ for each } j \in G_{k+1} \text{ and } i = 1, ..., m, \end{aligned} \tag{3.5}$$

Let $A = (a_{qi})_{q,i=1}^{k,m} \in \mathbb{R}^{k \times m}$ be the corresponding $k \times m$ matrix with the rows $\mathbf{r}_1^{\mathrm{T}}, ..., \mathbf{r}_k^{\mathrm{T}}$:

$$A = \begin{pmatrix} \mathbf{r}_1^{\mathrm{T}} \\ \mathbf{r}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{r}_k^{\mathrm{T}} \end{pmatrix}.$$

Then the row $\mathbf{r}_q$ appears exactly $\#G_q$ times in E for $q = 1, ..., k$. In addition E has $\#G_{k+1}$ zero rows. Clearly the row space of E is the row space of A. So $k \geq \operatorname{rank} E = \operatorname{rank} A$. Hence if $\operatorname{rank} A = k$ then

$$\sigma_1(E) \geq ... \geq \sigma_k(E) > \sigma_{k+1}(E) = ... = \sigma_m(E) = 0.$$

However, there is no simple formula relating the singular values of E and A. It may happen that the rows of A are linearly dependent which indicates that several groups out of $G_1, ..., G_k$ are somehow related, and the number of the significant singular values of E is less than k.

Conclusion: *The number of gene clusters is no less than the number of significant singular values of gene expression matrix.*

4 Missing Data in the Gene Expression Matrix

We now consider the problem of missing data in the gene expression matrix E. (Our analysis can be applied to any matrix E.) Let $\mathcal{N} \subset [n]$ denote the set of rows of E that contain at least one missing entry. Thus for each $j \in \mathcal{N}^c := [n]\backslash\mathcal{N}$, the gene $\mathbf{g}_j^{\mathrm{T}}$ has all of its entries. Let n' denote the size of $\mathcal{N}^c$ so that the size of $\mathcal{N}$ is $n - n'$. We want to complete the missing entries of each $\mathbf{g}_j^{\mathrm{T}}$, $j \in \mathcal{N}$, under some assumptions.

We first describe the reconstruction of the missing data in E using SVD as given in [1].

4.1 Imputation using SVD

Let E' be the $n' \times m$ matrix containing the rows $\mathbf{g}_j^{\mathrm{T}},\ j \in \mathcal{N}^c$ of E which do not have any missing entries, and l' be the number of significant singular values of E'. Let $\mathbf{X} \subset \mathbb{R}^m$ be the invariant subspace of the symmetric matrix $(E')^{\mathrm{T}}E'$ corresponding to the eigenvalues $\sigma_1(E')^2, ..., \sigma_{l'}(E')^2$. Let $\mathbf{x}_1, ..., \mathbf{x}_{l'}$ be the orthonormal eigenvectors of $(E')^{\mathrm{T}}E$ corresponding to the eigenvalues $\sigma_1(E')^2, ..., \sigma_{l'}(E')^2$. Then $\mathbf{x}_1, ..\mathbf{x}_{l'}$ is a basis of $\mathbf{X}$.

Let $\mathcal{M} \subset [m]$ be a subset of cardinality $m - m'$. Consider the projection $\pi_{\mathcal{M}} : \mathbb{R}^m \to \mathbb{R}^{m'}$ by deleting all the coordinates $i \in \mathcal{M}$ for any vector $\mathbf{x} = (x_1, ..., x_m)^{\mathrm{T}} \in \mathbb{R}^m$. Then $\pi_{\mathcal{M}}(\mathbf{X})$ is spanned by $\pi_{\mathcal{M}}(\mathbf{x}_1), ..., \pi_{\mathcal{M}}(\mathbf{x}_{l'})$.

Fix $j \in \mathcal{N}$ and let $\mathcal{M} \subset [m]$ be the set of experiments (columns) where the gene $\mathbf{g}_j^{\mathrm{T}}$ has missing entries. Let $\mathbf{y} \in \pi_{\mathcal{M}}(\mathbf{X})$ be the least square approximation to $\pi_{\mathcal{M}}(\mathbf{g}_j)$. Then any $\overline{\mathbf{g}}_j \in \pi_{\mathcal{M}}^{-1}(\mathbf{y})$ is a completion of $\mathbf{g}_j$. If $\pi_{\mathcal{M}}|\mathbf{X}$ is 1-1 then $\overline{\mathbf{g}}_j$ is unique. Otherwise one can choose $\overline{\mathbf{g}}_j \in \pi_{\mathcal{M}}^{-1}(\mathbf{y})$ with the least norm. Note that to find $\mathbf{y} \in \pi_{\mathcal{M}}(\mathbf{X})$ one needs to solve the least square problem for a subspace $\pi_{\mathcal{M}}(\mathbf{X})$. In principle, for each $j \in \mathcal{N}$ one solves a different least square problem. The crucial assumption of this method is

$$l = l'. \tag{4.1}$$

That is *the completed matrix E and its submatrix E' have the same number of significant singular values*. This follows from the observation that the completion of the row $\mathbf{g}_j, j \in \mathcal{N}$ lies in the subspace $\mathbf{X}$. Note that the inequalities (2.9) imply that the assumption (4.1) can be a very restrictive assumption.

The significant singular values of E' and of the reconstructed E are joint functions of all the rows (genes). By trying to reconstruct the missing data in each gene $\mathbf{g}_j^{\mathrm{T}}$, for $j \in \mathcal{N}$, separately, we ignore any correlation between $\mathbf{g}_j^{\mathrm{T}}$ and the genes $\mathbf{g}_q^{\mathrm{T}},\ q \in \mathcal{N}$; consequently, this will have an impact on the singular values of E. In the following section we suggest a different approach which treats the estimation problem of all the missing data simultaneously.

4.2 Reconsideration of 3.1

Let us reconsider Example 3.1. Assume that $\operatorname{rank} A = k$. Let $j \in \mathcal{N}$ and assume that the gene j is in the cluster G_q. Then we can reconstruct all missing entries of $\mathbf{g}_j^{\mathrm{T}}$ if $G_q \backslash \mathcal{N} \neq \emptyset$. Indeed, if for some gene $p \in G_q$ we have the results of m experiments, then $\mathbf{g}_j = \mathbf{g}_p$ and we reconstructed the missing entries for $\mathbf{g}_j$. In this example we can reconstruct all the missing entries in E if E' has the same rank as E. Equivalently, we can reconstruct all the missing entries in E if the equality (4.1) holds, where l and l' are the ranks of E and E' respectively.

4.3 Iterative method using SVD

In the recent papers [12] and [2], the following iterative method using SVD to impute missing values in a gene expression matrix is suggested. First, replace the missing values with 0 or with values computed from another method. Call the estimated matrix E_p, where $p = 0$. Find the l_p significant singular values of E_p, and let E_{p,l_p} be the filtered part of E_p (3.3). Replace the missing values in E by the corresponding values in E_{p,l_p} to obtain the matrix E_{p+1}. Continue this process until E_p converges to a fixed matrix (within a given precision). This algorithm takes into account implicitly the influence of the estimation of one entry on the other ones. But it is not clear if the algorithm converges, nor what are the features of any fixed point(s) of this algorithm.

5 The Optimization Problem

We now show that the estimation problem discussed in the previous section can be cast as the following optimization problem:

Problem 5.1 *Let $\mathcal{S}$ be a given subset of $[n] \times [m]$. ($\mathcal{S}$ is the set of uncorrupted entries of the gene expression matrix E given by (3.1).) Let $e(\mathcal{S}) := \{e_{ji},\ (j,i) \in \mathcal{S}\}$ be a given set of real numbers. ($e(\mathcal{S})$ is the set of uncorrupted (known) values of the entries of E.) Let $\mathrm{M}(e(\mathcal{S})) \subset \mathbb{R}^{n\times m}$ be the affine subset of all matrices $A = (a_{ji}) \in \mathbb{R}^{n\times m}$ such that $a_{ji} = e_{ji}$ for all $(j,i) \in \mathcal{S}$. ($\mathrm{M}(e(\mathcal{S}))$ all possible choices for E.) Let ℓ be a positive integer not exceeding m. Find $\hat{E} \in \mathrm{M}(e(\mathcal{S}))$ with the minimal σ_ℓ.*

Let $E = (g_{ji})$ denote the gene expression matrix with missing values. We choose the $\mathcal{S}$ in Problem 5.1 to be the set of coordinates (j,i) for which the entry g_{ji} is not missing. Recall that $\mathcal{N} \subset [n]$ denotes the set of rows of E, such that each row $j \in \mathcal{N}$ contain at least one missing entry. The cardinality of $\mathcal{N}$ is $n - n'$. Thus the set $\mathcal{S}$ contains all elements $(j,1), ...(j,m)$ for each $j \in \mathcal{N}^c$. The complement of $\mathcal{S}$ is the set of coordinates $\mathcal{S}^c = \{(j,i) \,|\, g_{ji} \text{ is missing}\} \subset \mathcal{N} \times [m]$. Let o denote the total number of missing entries in E. Then $o \geq n - n'$.

Let E' be the matrix as in §4.1 with l' significant singular values. Note that (2.9) yields $\sigma_q(E) \geq \sigma_q(E')$ for $q = 1, ..., m$. Thus if we want to complete E such that the resulting matrix still has exactly l' significant singular values, we should consider Problem 5.1 with $\ell = l' + 1$.

A more general possibility is to assume that the number of significant singular values of a possible estimation of E is $l = l' + k$ where k is a small integer, e.g. $k = 1$ or 2. That is, the group of genes $\mathbf{g}_j^{\mathrm{T}}$ for $j \in \mathbb{N}$ contributes to $l' + 1, ..., l' + k$ significant eigengenes of E. Then one considers Problem 5.1 with $\ell = l' + k + 1$.

We now consider a modification of Problem 5.1 which has a nice numerical algorithm.

Problem 5.2 *Let $\mathcal{S} \subset [n] \times [m]$ and denote by $e(\mathcal{S})$ a given set of real numbers e_{ji} for $(j, i) \in \mathcal{S}$. Let $\mathrm{M}(e(\mathcal{S})) \subset \mathbb{R}^{n\times m}$ be the affine subset of all matrices $A = (a_{ji}) \in \mathbb{R}^{n\times m}$ such that $a_{ji} = e_{ji}$ for all $(j, i) \in \mathcal{S}$. Let ℓ be a positive integer not exceeding m. Find $\hat{E} \in \mathrm{M}(e(\mathcal{S}))$ such that $\sum_{q=\ell}^{m} \sigma_q^2$ is minimal.*

Clearly, we can find $E \in \mathrm{M}(e(\mathcal{S}))$ with a "small" $\sigma_\ell^2(E)$ if and only if we can find $E \in \mathrm{M}(e(\mathcal{S}))$ with a "small" $\sum_{q=\ell}^{m} \sigma_q^2(E)$.

6 Fixed Rank Approximation Algorithm

We now describe one of the standard algorithms to solve Problem 5.2. Mathematically it is stated as follows:

Algorithm 6.1 Fixed Rank Approximation Algorithm (FRAA) *Let $E_p \in \mathrm{M}(e(\mathcal{S}))$ be the p^{th} approximation to a solution of Problem 5.2. Let $A_p := E_p^{\mathrm{T}} E_p$ and find an orthonormal set of eigenvectors for A_p, $\mathbf{v}_{p,1}, ..., \mathbf{v}_{p,m}$ as in (2.6). Then E_{p+1} is a solution to the following minimum of a convex nonnegative quadratic function*

$$\min_{E\in \mathrm{M}(e(\mathcal{S}))} \sum_{q=\ell}^{m} (E\mathbf{v}_{p,q})^{\mathrm{T}} (E\mathbf{v}_{p,q}). \tag{6.1}$$

The flow chart of this algorithm can be given as:

Fixed Rank Approximation Algorithm (FRAA)
Input: integers $m, n, L, iter$, the locations of non-missing entries $\mathcal{S}$, initial approximation E_0 of $n \times m$ matrix E.
Output: an approximation E_{iter} of E.
for $p = 0$ **to** $iter - 1$
- Compute $A_p := E_p^{\mathrm{T}} E_p$ and find an orthonormal set of eigenvectors for A_p, $\mathbf{v}_{p,1}, ..., \mathbf{v}_{p,m}$.
- E_{p+1} is a solution to the minimum problem (6.1) with $\ell = L$.

We now explain the algorithm and show that in each step, we decrease the value of the function we minimize:

$$\sum_{i=\ell}^{m} \sigma_q^2(E_p) \geq \sum_{q=\ell}^{m} \sigma_q^2(E_{p+1}). \tag{6.2}$$

For any integer $k \in [m]$, let Ω_k denote the set of all k orthonormal vectors $\{\mathbf{y}_1, ..., \mathbf{y}_k\}$ in $\mathbb{R}^m$. Let A be an $m \times m$ real symmetric matrix and assume (2.6). Then the minimal principle (the Ky-Fan characterization for $-A$) is:

$$\sum_{q=\ell}^{m} \lambda_q(A) = \sum_{q=\ell}^{m} \mathbf{z}_q^{\mathrm{T}} A \mathbf{z}_q = \min_{\{\mathbf{y}_\ell, ..., \mathbf{y}_m\} \in \Omega_{m-\ell+1}} \sum_{q=\ell}^{m} \mathbf{y}_q^{\mathrm{T}} A \mathbf{y}_q. \tag{6.3}$$

See for example [3].

Let $E = E_p + X \in \mathrm{M}(e(\mathcal{S}))$. Then $X = (x_{ji})_{j,i=1}^{n,m}$ where $x_{ji} = 0$ if $(j,i) \in \mathcal{S}$ and x_{ji} is a free variable if $(j,i) \notin \mathcal{S}$.

Let $\mathbf{x} = (x_{j_1 i_1}, x_{j_2 i_2}, \ldots, x_{j_o i_o})^{\mathrm{T}}$ denote the $o \times 1$ vector whose entries are indexed by $\mathcal{S}^c$, the coordinates of the missing values in E. Then there exists a unique $o \times o$ real valued symmetric nonnegative definite matrix $o \times o$ matrix B_p which satisfies the equality

$$\mathbf{x}^{\mathrm{T}} B_p \mathbf{x} = \sum_{q=\ell}^{m} \mathbf{v}_{p,q}^{\mathrm{T}} X^{\mathrm{T}} X \mathbf{v}_{p,q}. \tag{6.4}$$

Let $F(j,i)$ be the $n \times m$ matrix with 1 in the (j,i) entry and 0 elsewhere. Then the (s,t) entry of B_p is given by

$$b_p(s,t) = \frac{1}{2} \sum_{q=\ell}^{m} \mathbf{v}_{p,q}^{\mathrm{T}} (F(j_s, i_s)^{\mathrm{T}} F(j_t, i_t) + F(j_t, i_t)^{\mathrm{T}} F(j_s, i_s)) \mathbf{v}_{p,q}, \tag{6.5}$$
$$s, t = 1, \ldots o.$$

The proof of (6.5) is given in the Appendix. The crucial observation is that B_p can be decomposed into the direct sum of o symmetric nonnegative definite matrices indexed by $\mathcal{N}$.

Hence the function minimized in (6.1) is given by

$$\sum_{q=\ell}^{m} \mathbf{v}_{p,q}^{\mathrm{T}} E^{\mathrm{T}} E \mathbf{v}_{p,q} = \sum_{q=\ell}^{m} \mathbf{v}_{p,q}^{\mathrm{T}} (A_p + E_p^{\mathrm{T}} X + X^{\mathrm{T}} E_p + X^{\mathrm{T}} X) \mathbf{v}_{p,q} =$$
$$\mathbf{x}^{\mathrm{T}} B_p \mathbf{x} + 2 \mathbf{w}_p^{\mathrm{T}} \mathbf{x} + \sum_{q=\ell}^{m} \lambda_q(A_p) =$$

$$\sum_{i\in\mathcal{N}}(\mathbf{x}_j^{\mathrm{T}}B_{p,j}\mathbf{x}_j + 2\mathbf{w}_{p,j}^{\mathrm{T}}\mathbf{x}_j) + \sum_{q=\ell}^{m}\lambda_q(A_p), \tag{6.6}$$

where $\mathbf{w}_p := (w_{p,1},\ldots,w_{p,o})^{\mathrm{T}}$, and

$$w_{p,t} = \sum_{q=\ell}^{m}\mathbf{v}_{p,q}^{\mathrm{T}}E_p^{\mathrm{T}}F(j_t,i_t)\mathbf{v}_{p,q},\ t=1,...,o.$$

For $j\in\mathcal{N}$ the vector $\mathbf{x}_j\in\mathbb{R}^{o_j}$ contains all o_j missing entries of E in the row j of the form $x_{ji_t}, i_t\in O_j$ for the corresponding set $O_j\subset[m]$ of cardinality o_j. (See Appendix.) Since the expression in (6.1), and hence in (6.6), is always nonnegative, it follows that $\mathbf{w}_p$ is in the column space of B_p. Hence the minimum of the function given in (6.6) is achieved at the critical point

$$B_p\mathbf{x}_{p+1} = -\mathbf{w}_p, \tag{6.7}$$

and this system of equations is always solvable. (If B_p is not invertible, we find the least-squares solution).

We now show (6.2). The vector $\mathbf{x}_{p+1}$ contains the entries for the matrix X_{p+1}. Then $E_{p+1} := E_p + X_{p+1}$. ¿From the definition of $A_{p+1} := E_{p+1}^{\mathrm{T}}E_{p+1}$ and the minimality of $\mathbf{x}_{p+1}$ we obtain

$$\sum_{q=\ell}^{m}\sigma_q(E_p)^2 = \sum_{q=\ell}^{m}\mathbf{v}_{p,q}^{\mathrm{T}}(E_p+0)^{\mathrm{T}}(E_p+0)\mathbf{v}_{p,q} \geq$$
$$\sum_{q=\ell}^{m}\mathbf{v}_{p,q}^{\mathrm{T}}(E_p+X_{p+1})^{\mathrm{T}}(E_p+X_{p+1})\mathbf{v}_{p,q} = \sum_{q=\ell}^{m}\mathbf{v}_{p,q}^{\mathrm{T}}A_{p+1}\mathbf{v}_{p,q} \geq$$
$$\sum_{q=\ell}^{m}\lambda_q(A_{p+1}) = \sum_{q=\ell}^{m}\sigma_q(E_{p+1})^2.$$

□

In Appendix B, we give an algorithm to solve 6.7 efficiently. See Appendix C for the Matlab code of the algorithm. We conclude this section by remarking that to solve Problem 5.1, one may use the methods of [4].

7 Simulation

We implemented the Fixed Rank Approximation Algorithm (FRAA) in Matlab and tested it on the microarray data *Saccharomyces cerevisiae* [11] as provided at

http://genome-www.stanford.edu (the elutriation data set). The dimension of the complete gene expression matrix is 5981×14. We randomly deleted a set of entries and ran FRAA on this "corrupted" matrix to obtain estimates for the deleted entries. The FRAA requires four inputs: the matrix E with N rows and M columns with missing entries, an initial guess for the missing entries, a parameter L–the number of significant singular values, and the number of iterations. We set the initial guess to the missing data matrix with 0's replacing the missing values, the number of significant values to $L = 2$, and ran the algorithm through 5 iterations. (There was no significant change in the estimates when we replaced $L = 2$ with $L = 3$.)

We compared our estimates to estimates obtained by three other methods: replacing missing values with 0's (zeros method), row means (row means method), or the values obtained by the KNNimpute program [12]. We used a normalized root mean square as the metric for comparison: if C represents the complete matrix and E_p represents an estimate to the corrupted matrix E, then the root mean square (RMS) of the difference $D = C - E_p$ is $\frac{||D||_{\mathcal{F}}}{\sqrt{N}}$. We normalized the root mean square by dividing RMS by the average value of the entries in C.

In simulations where $1\% - 20\%$ of the entries were randomly deleted from the complete matrix C, the FRAA performed slightly better than the row means method, and significantly better than the zeros method. However, the KNNimpute algorithm (with parameters k$= 15$, d$= 0$) produced the most accurate estimates, with normalized RMS errors that were smaller than the normalized RMS errors from the other three methods. Figure 7.1 displays the results of one set of experiments estimating the elutriation matrix when each of $1, 5, 10, 15, 20\%$ of entries was removed: the normalized RMS errors are plotted against percent missing. When 25 simulations of deleting and then estimating 5% of the the entries was conducted, we found the average normalized RMS to be approximately 0.19 for KNNimpute and 0.24 for FRAA, with standard deviation to be approximately 0.02 for both methods. Not surprisingly, normalized RMS's increase with increasing percentage of missing values.

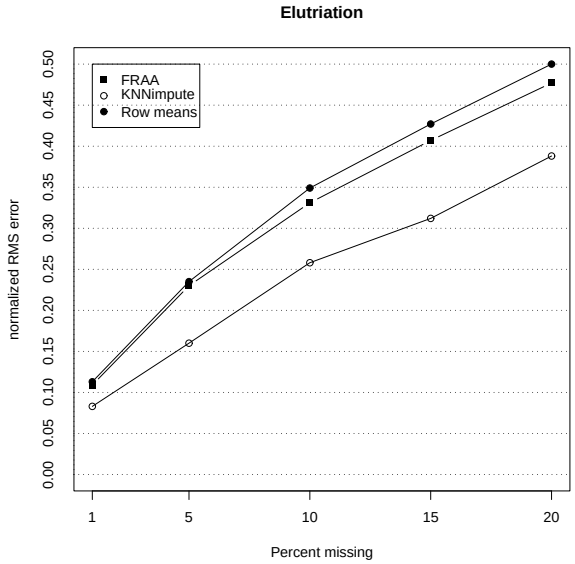

Fig. 7.1 Comparison of normalized RMS against percent missing for threee methods: FRAA, KNNimpute, and row means methods. The normalized RMS for the zeros method is not displayed, but the values are $0.397, 0.870, 1.24, 1.52, 1.76$, for $1, 5, 10, 15, 20\%$ percent missing, respectively.

In [12], the authors caution against using KNNimpute for matrices with fewer than 6 columns. We randomly selected four columns from the elutriation data set to form a truncated data set, then randomly deleted from $1\% - 20\%$ of the entries from this newly formed matrix. Figure 7.2 gives a comparison of the normalized RMS errors against percent missing in one run of the simulation at each of the percentages. When 25 simulations at 10% missing was run, we found the average normalized RMS to be approximately 0.143 for FRAA and 0.166 for KNNimpute, with standard deviations of approximately, 0.001 and 0.003, respectively.

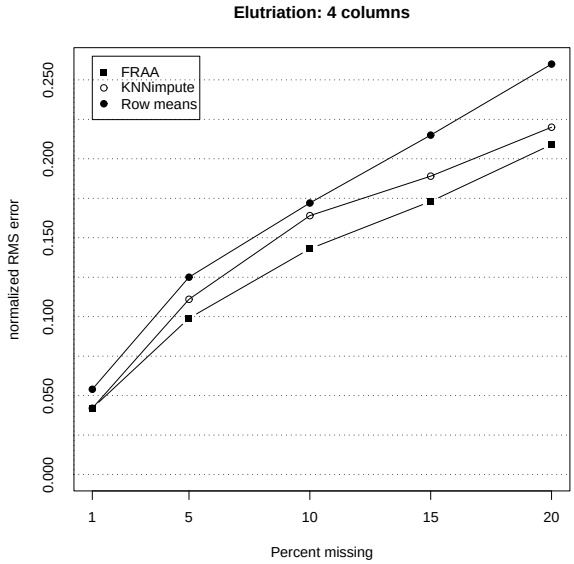

Fig. 7.2 Four columns of the full elutriation matrix were randomly selected. Entries were then randomly deleted from this truncated matrix. Plot of normalized RMS against percent missing.

For one simulation in which we randomly deleted and then estimated 10% (4200) of the entries from the full elutriation matrix,we compared the raw errors (true value - estimated value) for each of the 4200 imputed entries obtained using either KNNimpute or FRAA. Figure 7.3 shows a scatter plot of the raw errors from the estimate using KNNimpute against the raw errors from the estimate using FRAA. This plot seem to suggest that the algorithms KNNimpute and FRAA are rather consistent in how they are estimating the missing values.

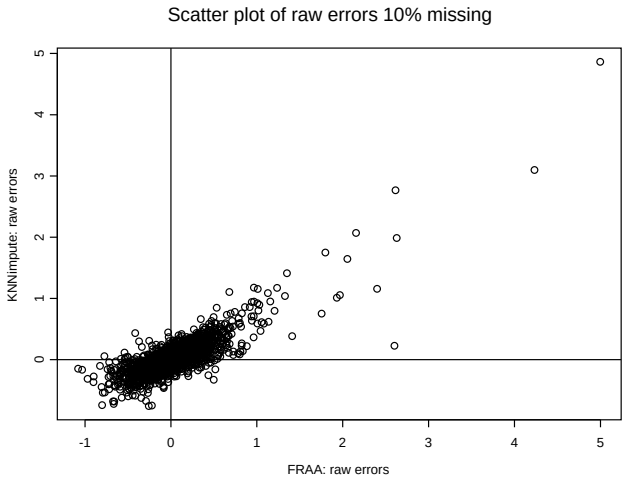

Fig. 7.3 Scatter plot of the raw errors (true - estimate) of each of the 4200 imputed entries in one simulation using KNNimpute and FRAA. The correlation between the two sets of raw errors is $.84$.

We ran similar simulations on the Cdc15 data set available on the web, (http://genome-www.stanford.edu/SVD/htmls/spie.html), and on subsets of this data set (using 4 columns). We also ran a couple of simulations on one of the data sets included by [10]. The outcomes were similar to that using the Elutriation data set, with the FRAA algorithm outperforming KNN on the matrices with a small number of columns.

8 Discussion

The Fixed Rank Approximation Algorithm uses singular value decomposition to obtain estimates of missing values in a gene expression matrix. It uses all the known information in the matrix to simultaneously estimate all missing entries. Preliminary tests indicate that, under a normalized root mean square metric, FRAA is more accurate than replacing missing values with 0's or with row means. The KNNimpute algorithm was more accurate when estimating missing entries deleted from the full elutriation matrix, but FRAA might be a feasible alternative in cases when the number of columns is small.

FRAA is another option, in addition to KNN, Bayesian estimations or local least squares imputations, for estimating missing values in gene expression data. FRAA by itself is very useful tool for gene data analysis without using clustering methods. Experimental results on various data sets shows that FRAA is robust. FRAA has been used by several computational biologists, who confirmed the accessibility of the algorithm.

To improve the results given by FRAA one needs to combine it with an algorithm for gene clustering. A possible implementation is as follows: First, apply FRAA to the cor-

rupted data set; next, using this estimated data set, subdivide the genes into clusters of genes with similar traits; now apply FRAA again to the missing entries of genes in each cluster. We intend to apply these steps in a future paper.

Our final remark is that the biology of the data should guide the researcher in determining the best method to use for imputing missing values in these data sets.

Appendix

A Proof of (6.5)

Let $\mathcal{N} \subset [n]$. Let $\mathcal{S}(j)$ denote the set of coordinates in row j with known values in E so that $\mathcal{S}(j)^{\mathbf{c}}$ denotes the set of coordinates of the missing values in row j.

$$\mathcal{S}^c = \cup_{j\in\mathcal{N}}\mathcal{S}(j)^c,\ \mathcal{S}(j)^c = \{(j,i(j,1)),...,(j,i(j,o_j))\}, \tag{A.1}$$
$$m \geq i(j,o_j) > ... > i(j,1) \geq 1 \quad \text{for } j \in \mathcal{N},$$
$$o := \sum_{j\in\mathcal{N}} o_j. \tag{A.2}$$

Note that the set O_j described just after (6.6) is given by $O_j := \{i(j,1),...,i(j,o_j)\}$.

Theorem A.1 *The $o \times o$ symmetric nonnegative definite matrix B_p given by (6.4) decomposes into a direct sum of $\#\mathcal{N} = n - n'$ symmetric nonnegative definite matrices indexed by the set $\mathcal{N}$:*

$$B_p = \oplus_{j\in\mathcal{N}} B_{p,j}, \quad B_{p,j} = (b_{p,j}(q,r))_{q,r=1}^{o_j}) \text{ is } o_j \times o_j \text{ for } j \in \mathcal{N}, \tag{A.3}$$

and

$$\mathbf{x}^{\mathrm{T}} B_p \mathbf{x} = \sum_{i\in\mathcal{N}} \mathbf{x}_j^{\mathrm{T}} B_{p,j} \mathbf{x}_j. \tag{A.4}$$

More precisely, let $\mathbf{v}_{p,k} = (v_{p,k,1},...,v_{p,k,m})^{\mathrm{T}},\ k = 1,...,m$ be given as in Algorithm 6.1. Then

$$b_{p,j}(q,r) = \sum_{k=\ell}^{m} v_{p,k,i(j,q)} v_{p,k,i(j,r)}, \quad q,r = 1,...,o_j. \tag{A.5}$$

Equivalently, let W_p be the following $m \times m$ idempotent symmetric matrix ($W_p^2 = W_p$) of rank $m - l + 1$:

$$W_p = \sum_{k=\ell}^{m} \mathbf{v}_{p,k}\mathbf{v}_{p,k}^{\mathrm{T}} = T_p T_p^{\mathrm{T}}, \quad T_p = [\mathbf{v}_{p,\ell},...,\mathbf{v}_{p,M}] \in \mathbb{R}^{m\times(m-\ell+1)}. \tag{A.6}$$

Then $B_{p,j}$ is the submatrix of W_p of order o_j with respect to the rows and columns in the set O_j for $j \in \mathcal{N}$. In particular, if in each row of E there is at most one missing entry then B_p is a diagonal matrix.

Proof. View the rows and the columns of B_p as indexed by $(s,i(s,q))$ and $(t,i(t,r))$ respectively, where $s,t \in \mathcal{N}$ and $q = 1,...,o_s,\ r = 1,...,o_t$. (For the purposes of this proof, the notation here is slightly different from that in the body of the paper.) So $B_p =$

$(b_p((s,i(s,q)),(t,i(t,r))))$. Let $F(j,i)$ be the $n\times m$ matrix which has 1 on the (j,i) place and all other entries are equal to zero. Then

$$b_p((s,i(s,q)),(t,i(t,r))) = \\ \frac{1}{2}\sum_{k=\ell}^{m}\mathbf{v}_{p,k}^{\mathrm{T}}(F(s,i(s,q))^{\mathrm{T}}F(t,i(t,r))+F(t,i(t,r))^{\mathrm{T}}F(s,i(s,q)))\mathbf{v}_{p,k}, \quad \text{(A.7)}$$
$$s,t\in\mathcal{N},\ q=1,...,o_s, r=1,...,o_t.$$

It is straightforward to show that $F(s,i(s,q))^{\mathrm{T}}F(t,i(t,r))=0$ if $s\neq t$. Furthermore, for $s=t$ the matrix $F(s,i(s,q))^{\mathrm{T}}F(t,i(t,r))+F(t,i(t,r))^{\mathrm{T}}F(s,i(s,q))$ has 1 in the places $(i(s,q),i(t,r))$ and $(i(t,r),i(s,q))$ for $r\neq q$, and has 2 in the place $(i(s,q),i(s,q))$ if $r=q$ and zero in all other positions. Hence
$b_p((s,i(s,q)),(t,i(t,q)))=0$ unless $s=t$. If $s=t$ then a straightforward calculation yields (A.5). Other claims of the theorem follow straightforward from the equality (A.5). □

B Algorithm for (6.7)

From Theorem A.1, the system of equations $B_p\mathbf{x}=-\mathbf{w}_p$ in o unknowns is equivalent to $n-n'$ smaller systems

$$B_{p,j}\mathbf{x}_{p+1,j}=-\mathbf{w}_{p,j}\quad j\in\mathcal{N}. \quad \text{(B.1)}$$

Thus the big system of equations in o unknowns, the coordinates of $\mathbf{x}_{p+1}$, given (6.7) splits to $n-n'$ independent systems given in (B.1). That is, in the iterative update of the unknown entries of E given by the matrix E_{p+1}, the values in the row $j\in\mathcal{N}$ in the places $\mathcal{S}(j)^c$ are determined by the values of the entries of E_p in the places $\mathcal{S}(j)^c$ and the eigenvectors $\mathbf{v}_{p,\ell},...,\mathbf{v}_{p,m}$ of $E_p^{\mathrm{T}}E_p$.

We now show how to efficiently solve the system (6.7).

Algorithm B.1 *For $j\in\mathcal{N}$ let $T_{p,j}$ is the $o_j\times(m-\ell+1)$ matrix obtained from T_p, given by (A.6), by deleting all rows except the rows $i(j,1),...,i(j,o_j)$. Then (B.1) is equivalent to*

$$T_{p,j}T_{p,j}^{\mathrm{T}}\mathbf{x}_{p+1,j}=-\mathbf{w}_{p,j},\quad i\in\mathcal{N}, \quad \text{(B.2)}$$

which can be solved efficiently by the QR algorithm as follows. Write $T_{p,j}$ as $Q_{p,j}R_{p,j}P_{p,j}$, where $Q_{p,j}$ is an $o_j\times d_{p,j}$ matrix with $d_{p,j}$ orthonormal columns, $R_{p,j}$ is an upper triangular $d_{p,j}\times o_j$ matrix of rank $d_{p,j}$ nonzero rows, where the rank $V_{p,j}=d_{p,j}$, and $P_{p,j}$ is a permutation matrix. (The columns of $Q_{p,j}$ are obtained from the columns of $V_{p,j}$ using Gram-Schmidt process.) Then

$$Q_{p,j}^{\mathrm{T}}\mathbf{x}_{p+1,j}=-(R_{p,j}R_{p,j}^{\mathrm{T}})^{-1}Q_{p,j}^{\mathrm{T}}\mathbf{w}_{p,j}$$

and

$$\mathbf{x}_{p+1,j} = -Q_{p,j}(R_{p,j}R_{p,j}^{\mathrm{T}})^{-1}Q_{p,j}^{\mathrm{T}}\mathbf{w}_{p,j}, \quad j \in \mathcal{N} \tag{B.3}$$

is the least square solution for $\mathbf{x}_{p+1,j}$.

C Matlab code

```
function Ep1 = fraa(E,Ep,L,iter)
%Fixed rank algorithm -- estimate missing values
%Usage: fraa(E,Ep,L,iter)
%E: matrix with missing values
%Ep: initial solution
%L: parameter (number of significant singular values + 1)
%iter: number of iterations to perform
%Note: Any rows with all missing values must be removed
%%%%%%%%%% THIS IS  THE SET-UP
%Get size of E
    [N,M]=size(E);
       if (L > M)
        error('need L<=#columns of E ')
      end;
  %get index of missing values
   missing=find(isnan(E));
   %Number of missing values
    m=length(missing);
    m2=m*m;
%%%%%%%%%%% NOW WE WORK WITH THE ALGORITHM
  Xp1=zeros(N,M);
   track=iter;
 while(iter > 0)
   A=Ep'*Ep;
   %Find singular value decomposition of A
   [U,S,V]=svd(A);
   %Singular values of Ep
   sigma2=S(S~=0);
   singular=sqrt(sigma2);
   partial_sig2=sum(sigma2(L:M));
   total_sig2=sum(sigma2(1:M));
   fprintf('\n iteration %3.0f \n', track-iter+1)
   fraction=partial_sig2/total_sig2;
```

```
   fprintf(' partial sum/total sum of sq. singular values
      \n %1.8f', fraction)
   fprintf('\n')
%Construct B=Bp
     B=sparse(m,m);  %pre-allocate space
      [is,js]=ind2sub([N,M],missing(1:m));
       for s=1:m
         for t=s:m
            if (i(s)==i(t))
            B(s,t)=sum(U(js(s),L:M)*U(js(t),L:M)');
            B(t,s)=B(s,t);    %B is symmetric
           end  %end if
        end %end For t
      end %end for s
%%%NOW CONSTRUCT THE VECTOR Wp
W=sparse(m,1); %pre-allocate space
    for t=1:m
     K=sparse(N,M);
     K(missing(t))=1;
     W(t)=sum(diag(U(:,L:M)'*Ep'*K*U(:,L:M)));
   end %end for
  %Solve Bx_(p+1)= -W
   xp1=-B\W;
  %Create matrix B_{p+1}
   Xp1(missing)=xp1;
  %Update solution
   Ep=Ep+Xp1;
  %set counter
   iter=iter-1;
  end %End while
   fprintf('\n')
   fprintf(' singular values (final iteration):\n')
   fprintf('%16.6f',singular)
   Ep1=Ep;
```

For the Matlab m file or a version of this algorithm for R, see http://people.carleton.edu/~lchihara/LMCProf.html

References

[1] O. Alter, P.O. Brown and D. Botstein, Processing and modelling gene expression expression data using singular value decomposition, Proceedings SPIE, vol. 4266 (2001), 171-186.

[2] H. Chipman, T.J. Hastie and R. Tibshirani, Clustering micrarray data in: T. Speed, (Ed.), Statistical Analysis of Gene Expression Microarray Data, , Chapman & Hall/CRC, 2003 pp. 159-200.

[3] S. Friedland, Inverse eigenvalue problems, Linear Algebra Appl., 17 (1977), 15-51.

[4] S. Friedland, J. Nocedal and M. Overton, The formulation and analysis of numerical methods for inverse eigenvalue problems, SIAM J. Numer. Anal. 24 (1987), 634-667.

[5] G.H. Golub and C.F. Van Loan, *Matrix Computations*, John Hopkins Univ. Press, 1983.

[6] R.A. Horn and C.R. Johnson, Matrix analysis, Cambridge Univ. Press, 1987.

[7] D.A. Jackson, Stopping rules in principal component analysis: a comparison of heuristical and statisitical approaches, *Ecology* 74 (1993), 2204-2214.

[8] R.A. Johnson, D. W. Wichern, Applied Multivariate Statistical Analysis, Prentice Hall, New Jersey, 4th edition (1998).

[9] H. Kim, G.H. Golub and H. Park, Missing value estimation for DNA microarray gene expression data: local least squares imputation, *Bioinformatics* 21 (2005), 187-198.

[10] S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubara and S. Ishii, A Baesian missing value estimation method for gene expression profile data, *Bioinformatics* 19 (2003), 2088-2096.

[11] P.T. Spellman, G. Sherlock, M.Q. Zhang, V.R. Iyer, K. Anders, M.B. Eisen, P.O. Brown, D. Botstein and B. Futcher, Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization, Mol. Biol. Cell, **9** (1998), 3273-3297.

[12] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein and R. Altman, Missing value estimation for DNA microarrays, Bioinformatics 17 (2001), 520-525.

Acknowledgement. We thank Dr. Shigeyuki Oba for providing data sets and the referee for his remarks.