Manuscript 1

Published online February 20, 2004

Nucleic Acids Research, 2004, Vol. 32, No. 3 ***e34***
DOI: 10.1093/nar/gnh026

LSimpute: accurate estimation of missing values in microarray data with least squares methods

Trond Hellem Bø[1,*], Bjarte Dysvik[1] and Inge Jonassen[1,2]

[1]Department of Informatics and [2]Computational Biology Unit, BCCS, University of Bergen, HIB, N5020 Bergen, Norway

Received October 1, 2003; Revised December 19, 2003; Accepted January 9, 2004

ABSTRACT

Microarray experiments generate data sets with information on the expression levels of thousands of genes in a set of biological samples. Unfortunately, such experiments often produce multiple missing expression values, normally due to various experimental problems. As many algorithms for gene expression analysis require a complete data matrix as input, the missing values have to be estimated in order to analyze the available data. Alternatively, genes and arrays can be removed until no missing values remain. However, for genes or arrays with only a small number of missing values, it is desirable to impute those values. For the subsequent analysis to be as informative as possible, it is essential that the estimates for the missing gene expression values are accurate. A small amount of badly estimated missing values in the data might be enough for clustering methods, such as hierachical clustering or K-means clustering, to produce misleading results. Thus, accurate methods for missing value estimation are needed. We present novel methods for estimation of missing values in microarray data sets that are based on the least squares principle, and that utilize correlations between both genes and arrays. For this set of methods, we use the common reference name LSimpute. We compare the estimation accuracy of our methods with the widely used KNNimpute on three complete data matrices from public data sets by randomly knocking out data (labeling as missing). From these tests, we conclude that our LSimpute methods produce estimates that consistently are more accurate than those obtained using KNNimpute. Additionally, we examine a more classic approach to missing value estimation based on expectation maximization (EM). We refer to our EM implementations as EMimpute, and the estimate errors using the EMimpute methods are compared with those our novel methods produce. The results indicate that on average, the estimates from our best performing LSimpute method are at least as accurate as those from the best EMimpute algorithm.

INTRODUCTION

Microarrays are used to obtain simultaneous measurements of the transcript abundance of thousands of genes in cell samples, revealing snapshots of the transcriptional state under a variety of conditions. Examples of application of microarrays are in studies of differential expression in different cell states (1–4), and for time series studies (5,6). The data produced by microarray experiments can be analyzed by various methods to order and visualize the information inherent in the data (7). For a number of reasons, it is not always possible to obtain a usable quantitation of all spots on an array; typical reasons include spotting problems, scratches on the slide, dust or hybridization failures. This in turn results in values missing from the gene expression matrix. Thus in every microarray project, one needs to determine how to treat missing values. Repeating the experiment is often not a realistic option, for economic reasons or because of limitations in available biological material.

Analysis results obtained using clustering algorithms, such as hierarchical clustering (8), K-means clustering and self-organizing maps (9), or data dimension reduction and projection methods such as singular value decomposition (10) or principal component analysis (11), will be influenced by the estimates replacing the missing values. Thus it is desirable to have accurate estimates of the missing values to get results from the analysis methods that are as realistic as possible. The data from microarray experiments are normally given as a matrix of expression levels of genes (rows) under different experimental conditions (columns). In two-channel experiments involving competitive hybridization, the levels are most often given as log2 transformed ratios. In such data sets, missing values are sometimes replaced by zeros (2) or, less often, by the average expression level for the gene ('row average'). The comparative study by Troyanskaya *et al*. (12) demonstrated that these simple approaches are far from optimal, as they do not utilize the correlation structure in the data.

In this paper, we propose specialized methods utilizing the least squares principle to estimate missing values using correlations between genes and between arrays. In the

*To whom correspondence should be addressed. Tel: +47 55584067; Fax: +47 55584199; Email: trondb@ii.uib.no

following, we refer to these methods collectively as LSimpute. The least squares principle is based on minimizing the sum of squared errors of a regression model. We describe two basic LSimpute methods, one estimation method utilizing correlations between genes (LSimpute_gene) and the other using correlations between arrays as a basis for the estimation (LSimpute_array). The motivation for using gene correlations as the basis for the estimation is the cellular co-regulation of genes in functional processes. The expression profiles obtained from the different arrays (represented by columns in the expression matrix) may also be correlated as our data set might contain array hybridizations of biological samples obtained from similar tissues or from neighboring time points in time series experiments. If different columns represent measurements obtained from biologically similar samples, we expect the corresponding columns in the gene expression matrix to be correlated. This can be exploited when missing values are to be estimated. However, in experiments where the samples are biologically very diverse, array expression patterns may provide a poor basis for estimation. Furthermore, we describe procedures for making weighted averages of the estimates from LSimpute_gene and LSimpute_array into combined estimates. We examine two variants of estimate combination that uses a bootstrapping approach for parameter (weight) estimation. The first, LSimpute_combined, uses a fixed global weighting of the estimates from the basic LSimpute methods, while the second, LSimpute_adaptive, uses an adaptive weighting scheme taking the data correlation structure into consideration to determine an appropriate weighting. The bootstrapping gives information on the relative strength of LSimpute_gene and LSimpute_array by re-estimating the same non-missing values with both methods, thereby estimating the joint error distribution associated with these methods. This information is in turn used to assign weights to the estimates. Additionally, we examine a classic approach to missing value estimation using expectation maximization (EM). We refer to our implementation of EM-based imputation as EMimpute, and we evaluate two variants, EMimpute_gene and EMimpute_array, utilizing, respectively, gene and array covariance structure for estimation of missing values.

We present a comparative study of the methods described above including the KNNimpute method. The methods are tested on three public data sets, taken from a *Listeria monocytogenes* infection time series study (13), a study of gene expression in several cancer cell lines also known as the NCI60 data set (14), and a study of gene expression in diffuse large B-cell lymphomas (2). The tests are performed on subsets of these data sets where no values are missing, generated by removing genes and arrays until no missing values remain. Missing values are simulated by randomly labeling a percentage of the non-missing elements in the matrix as missing. The ability of an imputation algorithm to produce accurate estimates of these elements is given a score, the root mean squared deviation (RMSD) between the true values and the estimated values, that allows for comparison of its success in predicting the missing values. In addition, we study the distribution of actual missing values in the three example data sets, assessing the influence this might have on real cases of missing value estimation.

The rest of the paper is structured as follows. In Materials and Methods, we give a detailed description of the methods and describe how the empirical evaluation was performed. In the Results, we study the distrubution of missing values in real cases, exemplified by the three test data sets mentioned above. We also describe the results of a comparison between our LSimpute methods, the EMimpute and the KNNimpute methods on the three test data sets. In the Discussion, we sum up our findings.

MATERIALS AND METHODS

Here we describe the two basic methods, LSimpute_gene and LSimpute_array, for imputation of missing data based on the least squares principle. LSimpute_gene is based on correlation between genes, while LSimpute_array is based on correlation between arrays. Furthermore, we propose two procedures for weighted combination of the estimates from the basic methods into combined estimates. The combined estimates are designed so that they are expected to be (e.g. on average) as least as accurate as each of the component estimates. Software implementing the novel methods described in this paper can be downloaded from http://www.ii.uib.no/~trondb/imputation/.

It is common to write the linear regression model for y given x as $y = \alpha + \beta x + e$, where e is the error term for which the variance is minimized when estimating the model (parameters α and β) with least squares. In single regression, the estimate of α and β is $\hat{\alpha} = \bar{y} - \hat{\beta}\bar{x}$ and

$$\hat{\beta} = \frac{s_{xy}}{s_{xx}}, \qquad \text{where} \qquad s_{xy} = \frac{1}{n-1}\sum_{j=1}^{n}(x_j - \bar{x})(y_j - \bar{y})$$

is the empirical covariance between x and y,

$$s_{xx} = \frac{1}{n-1}\sum_{j=1}^{n}(x_j - \bar{x})^2$$

is the empirical variance of x, and n is the number of observations (number of times x and y have been observed together). Here $\bar{x}$ and $\bar{y}$ are the averages over $x_1,...,x_n$ and $y_1,...,y_n$. Thus the least squares estimate of a variable y given a variable x can be written as

$$\hat{y} = \bar{y} + \frac{s_{xy}}{s_{yy}}(x - \bar{x}).$$

The corresponding model for multiple regression, e.g. a model for $y_1,...,y_l$ given $x_1,...,x_k$, is $y_i = \alpha_i + \beta_{i1}x_1 + \beta_{i2}x_2 + ... + \beta_{ik}x_k + e$. It can be shown that the least squares estimate for this model can be formulated as $\hat{y}_i = \bar{y}_i + \mathbf{S}_{y_i\mathbf{x}}\mathbf{S}_{\mathbf{xx}}^{-1}(\mathbf{x} - \bar{\mathbf{x}})$ (15), where $\mathbf{x} = [x_1, x_2,...,x_k]^T$, $\bar{\mathbf{x}} = [\bar{x}_1,\bar{x}_2,...\bar{x}_k]^T$, $\mathbf{S}_{y_i\mathbf{x}} = [s_{y_ix_1},s_{y_ix_2},...s_{y_ix_k}]$ and

$$\mathbf{S}_{\mathbf{xx}} = \begin{bmatrix} s_{x_1x_1} & s_{x_1x_2} & \cdots & s_{x_1x_k} \\ \cdots & & & \cdots \\ \cdots & & & \cdots \\ s_{x_kx_1} & s_{x_kx_2} & \cdots & s_{x_kx_k} \end{bmatrix}.$$

The single regression model has two parameters to be estimated, while the multiple regression model has $l(k + 1)$ parameters. It is essential for good estimation of the parameters that many observations are available. The number of parameters in a model should only be a fraction of the

number of observations and, as a rule of thumb, there should be at least 5–10 times as many observations as parameters. When it comes to microarray data, it is common to have measurements for thousands of genes and a limited set of arrays, normally between 20 and 100. Given that we want to use correlations between genes as the basis for missing value estimation, the observations are the arrays. Thus multiple regression using gene correlations is only feasible when a small number of genes is included in the model. In addition, missing values in the data cause problems for estimation of the covariances. If missing values are present, there will often be few arrays where none of the genes in the model have missing values. To use all arrays in the covariance estimation, each missing value must be set to a value before estimation, for instance the row/gene mean.

Viewing the problem from another perspective, using correlations between arrays as the basis for missing value estimation, using a multiple regression model does not represent a problem as the genes normally outnumber the arrays by a large margin. However, before the covariance matrix is to be computed, one needs to perform an (intermediate) imputation of missing values. For this imputation, a row/gene mean can be used to substitute missing values.

The LSimpute_gene method

Since multiple regression for gene correlations is not feasible for more than a few genes, we propose using a weighted average of several single regression estimates of the same missing value. Given a missing value in the data matrix for gene y, only the k genes $x_1,...,x_k$ most correlated with y are included in the prediction model. In addition, none of $x_1,...,x_k$ is allowed to have a missing value in the same array as the missing value to be estimated. When determining which are the most correlated genes, we use the absolute correlation values since both positive and negative correlation between genes is equally well suited for regression. The correlation between genes x_i and y is determined by only including arrays where both genes have non-missing values in the computation. Given the k closest correlated genes, k estimates $\hat{y}_1,...,\hat{y}_k$ of the missing value are computed by single regression from each of $x_1,...,x_k$. Experimentation with different numbers of genes to be included in the estimation indicated that 10 was a suitable number for k in the LSimpute_gene method (data not shown). However, for other data sets, another value for k might be more appropriate. For each single regression estimate $\hat{y}_i$, the parameters α_i and β_i are based only on arrays where neither y nor x_i have missing values. Finally, a weighted average of the estimates is computed. The weighting is designed to give the genes most correlated with y the largest weights, as these are expected to give the best estimates of the missing value. Given the estimated correlation r_{yx_i} between genes y and x_i, the weight w_i assigned to the estimate $\hat{y}_i$ is

$$w_i = \left(\frac{r_{yx_i}^2}{1 - r_{yx_i}^2 + \varepsilon} \right)^2,$$

where $\varepsilon = 10^{-6}$. In this formula, the numerator approaches 1 with increasing absolute correlation, while the denominator approaches ε. Thus strong correlations will give large weights, and weak correlations will give small weights. The constant ε (arbitrarily set to 10^{-6}) is added to the denominator to avoid division by zero. The weights are scaled so that they sum to 1. We arrived as this specific weight formula by experimentation rather than theoretical arguments.

The LSimpute_array method

Since there are normally a lot more genes than arrays in a microarray study, multiple regression models can easily be applied to estimate missing values based on correlation between arrays. Given a gene y that has l missing values $y_{m_1},...,y_{m_l}$ and k non-missing values $y_{n_1},...,y_{n_k}$, define $\mathbf{y}_{(1)} = [y_{m_1},...,y_{m_l}]^T$ and $\mathbf{y}_{(2)} = [y_{n_1},...,y_{n_k}]^T$. Then the least squares estimate of $\mathbf{y}_{(1)}$ given $\mathbf{y}_{(2)}$ is $\hat{\mathbf{y}}_{(1)} = \bar{\mathbf{y}}_{(1)} + \mathbf{S}_{\mathbf{y}(1)\mathbf{y}(2)}\mathbf{S}^{-1}_{\mathbf{y}(2)\mathbf{y}(2)}(\mathbf{y}_{(2)} - \bar{\mathbf{y}}_{(2)})$, where $\bar{\mathbf{y}}_{(1)} = [\bar{y}_{m_1},...,\bar{y}_{m_l}]^T$ are the array mean expression levels, i.e. the average log ratio on each array, and the elements of $\mathbf{S}_{\mathbf{y}(1)\mathbf{y}(2)}$ and $\mathbf{S}_{\mathbf{y}(2)\mathbf{y}(2)}$ are empirical covariances between arrays. To estimate the array mean expression levels and covariances, we first substitute all missing values in the data by the estimates from the gene-based method, and then compute the covariances as if the data are complete. We found that this approach gave better estimates than when substituting the missing values with row mean or column mean. Column mean substitution of missing values resulted in the worst estimates (data not shown).

Combining the gene and array based estimates

Since LSimpute_gene and LSimpute_array take different correlations into consideration when estimating missing values, the deviations from the true values will not be completely correlated. We therefore propose combining the methods by taking weighted averages of the estimates from LSimpute_gene and LSimpute_array, such that on average we expect the new estimates to be at least as good as the best estimates from the two component methods.

Given a missing value y that we want to estimate using the LSimpute_gene estimate $\hat{y}_g$ and the LSimpute_array estimate $\hat{y}_a$, we must determine the mixing coefficient $p \in [0,1]$ for the combined estimate $\hat{y}_c = p\hat{y}_g + (1-p)\hat{y}_a$. First we explore the possibility of using one p for all estimates, referred to as LSimpute_combined, assuming that the ratio of the squared errors is the same for all estimated elements. Using this global model, with one p for all combined estimates, we determine p by re-estimating 5% of the known values by marking them as missing. Taking the deviations between estimated values and the known values $e_a = \hat{y}_a - y$ and $e_g = \hat{y}_g - y$, the best global mixing coefficient between the methods in order to minimize the sum of squared errors can be determined. The best global mixing coefficient is the value p^* that minimizes the sum of squared errors $\Sigma e_c^2 = \Sigma[p^2 e_g^2 + 2p(1-p)e_g e_a + (1-p)^2 e_a^2]$ for the re-estimated data; thus we expect this global p to minimize the sum of squared errors for the missing data as well. Note that $\Sigma e_c^2 \leqslant \min(\Sigma e_a^2, \Sigma e_g^2)$. So under the assumption that the joint distribution of e_a and e_g is the same for the re-estimated data as for the missing values, the average squared error of LSimpute_combined will be smaller or equal to that of the best of its two component methods.

The global model does not take into consideration that the ratio of squared errors may vary depending on the correlation structure in the data. Specifically, we expect the estimates from LSimpute_gene to be much more accurate for genes in strongly correlated clusters than for other genes. Plotting the

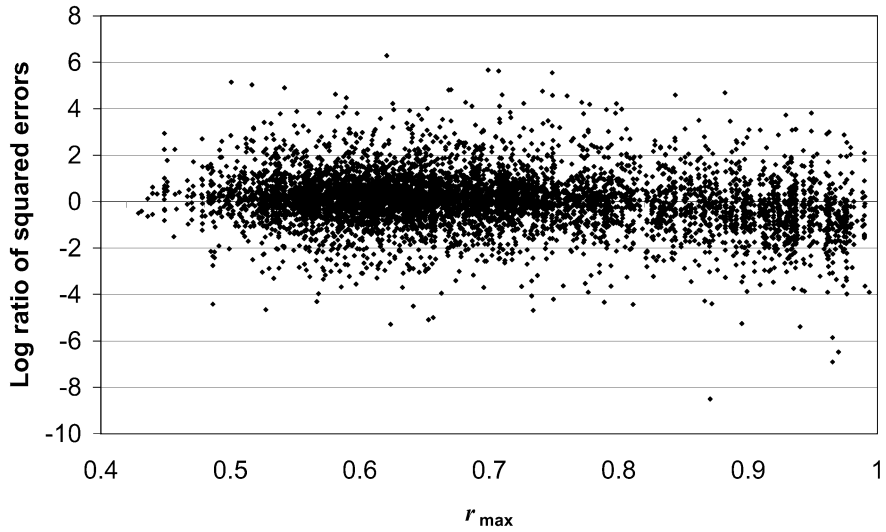

Figure 1. Maximum gene correlation versus log ratio of squared errors.

absolute correlation of the highest correlated gene used in gene-based estimation against $\log(e_g^2) - \log(e_a^2)$, i.e. the log ratio of the squared errors, we observe that the gene-based squared errors are smaller relative to array-based squared errors when gene correlation is high (Fig. 1). We therefore propose an adaptive weighting that takes into consideration the relationship between the squared errors of the LSimpute_gene and LSimpute_array estimates given the gene correlation strength. This method is referred to as LSimpute_adaptive. The adaptive weighting is determined by re-estimating 5% of the known values in the data matrix. From these estimates and for each maximum gene absolute correlation used in estimation $r_{\max}$, the best weighting p given $r_{\max}$ can be calculated. Thus for each pair of estimates $\hat{y}_a$ and $\hat{y}_g$ and the corresponding $r_{\max}$ for LSimpute_gene, the weights for combining these estimates are determined by looking at the best weighting given the same $r_{\max}$ for the re-estimated data. The best p given a specific $r_{\max}$ is determined by collecting all observations of errors e_a and e_g for the re-estimated data in the interval $r_{\max} \pm 0.05$, and then calculating p for these observations. For extreme values of $r_{\max}$, we demand that there are at least 100 observations as a basis for estimation of p. Thus we expand the interval until 100 observations are included if the interval $r_{\max} \pm 0.05$ contains too few observations.

Estimating missing values with the EM algorithm

Estimating missing values is a classical problem in statistics, and iterative algorithms based on the EM algorithm are widely used. We adopt an implementation of the EM algorithm for missing value estimation described by Johnson and Wichern (15). This algorithm uses the same estimation model for prediction as the multiple regression model described previously. The difference is that instead of the empirical covariance matrix S, we use Σ, the maximum likelihood estimate of the covariance matrix. The EM algorithm iterates while updating estimates of missing values and covariance matrix until the estimates stabilize. Thus the estimates are found by gradual convergence of covariance and missing value estimates.

We apply the algorithm for both gene-based and array-based estimation. The EM algorithm for gene-based estimation, referred to as EMimpute_gene, first selects 10 genes by the procedure that is used by LSimpute_gene. Thus EMimpute_gene will use the same genes for estimation as our gene-based method, and direct comparison of the RMSD with our non-iterative gene-based method is reasonable. Missing values are initialized to row/gene mean. Since the number of arrays in our three test data sets varies from 39 to 65, we have from four to six times as many arrays as genes in the estimation model. This is an absolute minimum for gene-based multiple regression, and ideally we should have at least twice as many arrays. If there are fewer arrays than in the test data sets used here, a reasonable multiple regression model cannot be computed.

Array-based estimation using EM, referred to as EMimpute_array, is done in the same manner as for LSimpute_array, except that the EM algorithm iteratively updates the estimates of the covariance matrix and missing values. Missing values are initialized to row/gene mean, as this results in faster convergence than starting with column/array mean (data not shown).

Evaluating imputation methods

To evaluate missing value imputation methods, we use data sets with no missing values. Some elements from the complete data are then marked as missing, and re-estimated using an imputation method. The accuracy of a set of imputation methods can then be compared by computing a statistic quantifying the deviation between the estimated and the true values for each imputation method. To evaluate the estimates made by each imputation method, the RMSD is computed. This gives small values for the method that best minimizes the squared deviations between the estimated values and the real values. The imputation method achieving the smallest RMSD gives the most correct picture of the complete data matrix when estimated values are included.

Table 1. Number of genes with different percentages of missing values in three example data sets

Missing values (%)	Lymphoma	NCI60	Time series
0	854	2069	6850
>0–5	1560	3734	3272
>5–10	797	581	2182
>10–15	530	174	1143
>15–20	285	106	867
>20	0	166	2524

Troyanskaya *et al*. (12) report the best results for K between 10 and 20. As a consequence of this, we choose to perform mutiple tests with KNNimpute using K = 5, 10, 15, 20 and 25.

We perform the tests by randomly removing (marking as missing) 5–25% (5, 10, 15, 20 and 25) of the data and re-estimating the missing values using the alternative imputation methods. Thus we get several estimates for each missing value, one for each imputation method used. To obtain results unbiased with regard to which portion of the data is missing, we run 100 independent rounds of this procedure and use the average RMSD from these as the performance metric for each imputation method.

Data sets

Data sets are chosen to represent different types of experiments in order to obtain results likely to hold for microarray experiment data in general. Specifically, we select three data sets from two cancer studies and one time series study. One data set comes from the NCI60 study (14). Here we selected the data from figure 3 in Ross *et al*. (14), and removed all genes with missing values, resulting in a 2069 × 64 data matrix. The second data set comes from a lymphoma study (2). Here we selected the data from Figure 1, and we first removed all arrays with >5% missing values, and then all genes with missing values, resulting in a 2317 × 65 data matrix. The third data set is from an infection time series study (13). Here we downloaded all the time course data (the file allTimeCourses.pcl), and removed all genes with missing values, resulting in a 6850 × 39 data matrix. The data sets were downloaded from the supplementary web pages accompanying the papers: http://llmpp.nih.gov/lymphoma/, web supplement to the lymphoma study, http://genome-www.stanford.edu/listeria/gut/, web supplement to the time series study, and http://genome-www.stanford.edu/nci60/index.shtml, web supplement to the NCI60 study.

RESULTS

Troyanskaya *et al*. (12) have shown that correlation between gene expression profiles is useful in the imputation of missing values. This will only be the case for gene profiles where the proportion of values that are missing is relatively low. We therefore analyzed the three data sets referred to above to find out how common it is for genes to have only a few values missing. As Table 1 shows, for most genes with missing values, only a few percent are missing. The time series data set has 6850 out of 16 838 genes without missing values in 39 arrays, which is the set used for testing the imputation methods. It is interesting to observe that for 6597 of the genes having missing values, <15% of the values are missing per gene. In the lymphoma data set, only 854 out of 4026 genes are without missing values in all 96 arrays. However, no genes have >20% of the values missing. The NCI60 data set shows a similar pattern, where 2069 out of 6830 genes are without missing values, but 4489 of the remaining 4761 genes have <15% missing values.

These results indicate first of all that missing values is a common problem that has to be addressed. At the same time, they show that the structure of the missing values in these three data sets is such that it is likely to allow imputation methods to make reliable estimates of the missing values for most of the genes. If missing values are dominating an expression profile for a gene, for instance if a gene has 70% missing values, few measurements remain to determine how the gene is correlated with other genes in the data set. However, since the example data sets show that most values are present for each gene, the basis for determining the correlation structure between genes is relatively good. Determining how the arrays are correlated appears to be a smaller problem since typically measurements are present for thousands of genes. However, if we are to impute values for a gene with many missing values, fewer arrays can be included in the array-based multiple regression model used for estimation in LSimpute_array, most probably leading to less accurate estimates.

We compare the LSimpute and EMimpute methods with KNNimpute (12), to see whether these methods represent an improvement over previously proposed methods. At present, KNNimpute is a widely used method for missing value imputation. The estimates from the KNNimpute method are sensitive to the choice of the parameter K, the number of gene neighbors used to estimate the missing values. Because of this, we have tested KNNimpute over a range of values for K, and only report the best results obtained here. Comparing the results obtained using the methods LSimpute_gene, LSimpute_array, EMimpute_gene and EMimpute_array with the results obtained using KNNimpute, we found that all these methods give a smaller RMSD than KNNimpute when 5% of the data are missing. The results are summarized in Figure 2 and Table 2. Due to the time it takes to run one iteration of EMimpute_gene (see below), this method is only tested with 5 and 10% missing values. Table 2 also lists the ratio of the RMSD between KNNimpute and the other methods for easier assessment of relative improvement compared with KNNimpute. LSimpute_gene gives a 4.4–9.7% smaller RMSD than KNNimpute with 5% missing values, while LSimpute_array gives a 6.8–19.8% smaller RMSD EMimpute_gene gives a 2.6–8.5% smaller RMSD than KNNimpute with 5% missing values, while EMimpute_array gives a 5.0–21.7% smaller RMSD. The results clearly favor the array-based estimation methods, as in two of three data sets they give markedly more accurate estimates than gene-based estimation methods. For the lymphoma data set, the RMSD obtained using array-based estimation is marginally worse. The difference is 1.3% comparing LSimpute_gene with LSimpute_array, and 3.5% comparing EMimpute_gene with EMimpute_array. In this case, LSimpute_gene and EMimpute_gene have approximately the same RMSD value, with only a 0.3% difference in favor of EMimpute. Overall, these results indicate that LSimpute_gene performs better than

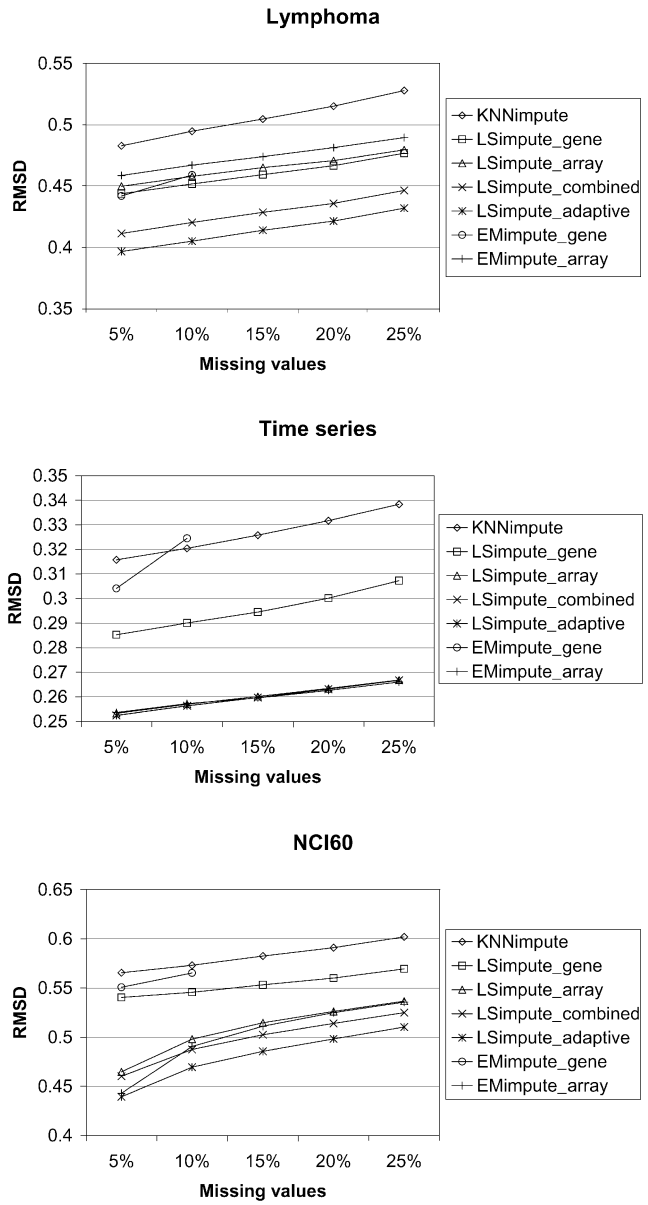

Figure 2. Comparison of estimation error (RMSD) for the methods on three data sets.

EMimpute_gene, while EMimpute_array may be a bit better overall than LSimpute_array. Somewhat surprisingly, we found that for two of the data sets, KNNimpute gave the best performance using K = 5 (NCI60 and lymphoma), while K = 10 gave the best performance for the last data set (time series). Troyanskaya *et al.* (12) reported that KNNimpute produced the most accurate estimates when K had a value in the range 10–20.

We performed a set of tests to evaluate how the methods for combining array- and gene-based estimates perform relative to the other methods. The results are summarized in Figure 2 and Table 3 and show that LSimpute_combined gives a smaller RMSD than each of its two component methods (LSimpute_gene and LSimpute_array), although only marginally for two of the data sets (NCI60 and time series). The marginal improvement over the best of the two, the array-based method, is an effect of the relatively large difference in RMSD between LSimpute_gene and LSimpute_array. For the lymphoma data, the accuracy (RMSD) of the two component methods is more even, and therefore a relatively large improvement compared with the better of the two is obtained by combining them. Our empirical results indicate that by combining the gene- and array-based methods, we obtain estimates that are at least as good as when using the best of the two. The lack of significant improvement for LSimpute_combined over its component methods in the time series data set is a result of the nature of this data set. The data set contains several similar infection time series with different mutants of *L.monocytogenes*. Baldwin *et al.* (13) report that a difference in host response was undetectable using different mutants. From this, we can practically view the time series as replicated experiments, and therefore we are not surprised by the superiority of the LSimpute_array over LSimpute_gene in estimating the missing values. Still, we expect that LSimpute_combined will give an estimate at least as accurate as each of the component methods, as it takes into consideration the relative strengths of the two underlying methods.

Using the adaptive estimation model implemented in LSimpute_adaptive, we get an additional improvement for the NCI60 and lymphoma data sets compared with LSimpute_combined. Thus by performing an adaptive weighting of the estimates from LSimpute_gene and LSimpute_array based on the structure of the data, we obtain the most accurate estimates (lowest RMSD) of all methods tested in this study. Performance for the time series data set is equal to that of LSimpute_combined, and approximately equal to LSimpute_array. Thus the nature of the time series data set, containing several closely related cell samples on different arrays, causes array correlations to be the best basis for missing value prediction. Overall, using LSimpute_adaptive, we see an improvement in RMSD of 18–20% compared with KNNimpute for all three data sets.

We want to test whether the prediction errors obtained using our most successful method, LSimpute_adaptive, are significantly smaller on average compared with the prediction errors we obtained using KNNimpute. For this purpose, we use a paired *t*-test, where the observations are the differences in the size of the errors made by the two methods. By taking the difference $d_i = |e_{i,\mathrm{KNN}}| - |e_{i,\mathrm{adaptive}}|$ for each missing value ($i = 1, 2, \ldots, n$, where n is the number of missing values), we can test whether the average difference in size of prediction error

$$\overline{d} = \frac{1}{n}\sum d_i$$

is significantly larger than 0. Here $e_{i,\mathrm{KNN}}$ and $e_{i,\mathrm{adaptive}}$ are the errors made by KNNimpute and LSimpute_adaptive, respectively, when estimating missing value number i. The formula for t:

$$t = \frac{\overline{d}}{s_d/\sqrt{n}},$$

where s_d is the empirical standard deviation of the d_is, is t-distributed with $n - 1$ degrees of freedom (df) under the null

Table 2. Comparison of basic LSimpute methods and EMimpute methods against KNNimpute with 5% missing data

	Lymphoma K = 5		NCI60 K = 5		Time series K = 10	
	RMSD	RMSD ratio versus KNNimpute	RMSD	RMSD ratio versus KNNimpute	RMSD	RMSD ratio versus KNNimpute
KNNimpute	0.4828		0.5656		0.3157	
LSimpute_gene	0.4437	0.9189	0.5404	0.9555	0.2853	0.9035
LSimpute_array	0.4498	0.9316	0.4648	0.8218	0.2533	0.8022
EMimpute_gene	0.4419	0.9152	0.5507	0.9736	0.3041	0.9633
EMimpute_array	0.4586	0.9500	0.4430	0.7832	0.2536	0.8034

Table 3. Comparison of basic and combined LSimpute methods and EMimpute methods against KNNimpute with 10% missing data

	Lymphoma K = 5		NCI60 K = 5		Time series K = 10	
	RMSD	RMSD ratio versus KNNimpute	RMSD	RMSD ratio versus KNNimpute	RMSD	RMSD ratio versus KNNimpute
KNNimpute	0.4947		0.5732		0.3204	
LSimpute_gene	0.4518	0.9132	0.5456	0.9520	0.2900	0.9050
LSimpute_array	0.4578	0.9254	0.4980	0.8689	0.2570	0.8019
LSimpute_combined	0.4202	0.8494	0.4874	0.8504	0.2563	0.7998
LSimpute_adaptive	0.4058	0.8202	0.4694	0.8190	0.2563	0.7999
EMimpute_gene	0.4590	0.9278	0.5654	0.9865	0.3245	1.0128
EMimpute_array	0.4671	0.9442	0.4906	0.8559	0.2572	0.8028

Table 4. Summary of time usage of all methods with different percentages of missing values; all results are in seconds

Method\%missing	Lymphoma					NCI60					Time series				
	5	10	15	20	25	5	10	15	20	25	5	10	15	20	25
KNNimpute	150	259	338	396	417	113	200	264	302	323	482	830	1079	1239	1325
LSimpute_gene	30	30	29	28	27	24	24	23	23	21	155	194	177	167	158
LSimpute_array	36	35	33	32	31	29	28	27	26	25	159	199	182	172	162
LSimpute_combined	68	67	66	61	56	56	54	52	50	48	375	378	375	356	333
LSimpute_adaptive	141	137	131	124	112	114	108	102	98	92	818	831	792	766	716
EMimpute_gene	221	504	964	1818	3149	186	439	865	1684	3303	510	1297	3150	5027	6600
EMimpute_array	24	25	26	29	29	75	50	41	34	29	16	16	18	19	20

hypothesis. Our null hypothesis states that $d = 0$, while the alternative hypothesis states that $\bar{d} > 0$, e.g that KNNimpute on average makes larger prediction errors.

We test whether the average difference in size of prediction error is significant for our three data sets by marking 5% of the values in each of our three data sets as missing, and taking the corresponding d_is as our observations. Given these observations, the *t*-statistic for the lymphoma data set scored 22.89, corresponding to a *P*-value of 1.86×10^{-112} (df = 7529). For the NCI60 data set, the *t*-statistic scored 25.68, corresponding to a *P*-value of 5.31×10^{-139} (df = 6620). For the time series data set, the *t*-statistic scored 34.22, corresponding to a *P*-value of 2.16×10^{-246} (df = 13 356). Thus we can conclude that the average estimation error is significantly larger using KNNimpute than that obtained using LSimpute_adaptive.

Finally, we comment on the running time required by the methods we study. All methods have been tested on a computer with a 2.8 GHz Pentium4 CPU running under Linux. For KNNimpute, we used an optimized compilation of the original C++ code made by Troyanskaya *et al*. (12). All other methods have been implemented in Java, and Java is started with the option –server that optimizes Java applications. The time required to run one round of missing value imputation is recorded for all methods and summarized in Table 4. Note that the running time for LSimpute_array includes the time it takes to run LSimpute_gene, which is done in order to initialize the missing values before array-based estimation. Our most CPU-intensive LSimpute method, LSimpute_adaptive, runs equally fast or faster than KNNimpute in all cases except one. We also note that EMimpute_array is relatively fast, considering that it is an iterative method, while EMimpute_gene is by far the slowest method.

DISCUSSION

The process of replacing missing values in a data matrix is an important part of the analysis of every microarray experiment, as most analysis methods require that the input data matrix is complete. Thus the quality of the missing value estimates is essential to get a picture of the complete data that is as realistic as possible when performing clustering or using other analysis methods. As demonstrated by our three example data sets, a large portion of the genes have only a few values missing. Thus the distribution of missing values in real data is not likely to prevent determination of the correlation structure between genes or arrays, and estimation of the missing values is likely

to be accurate. However, the experimenter should be cautious of the missing value structure in the data given as input to imputation methods. Any arrays or genes having too many missing values should be removed before the missing values are estimated for the remaining data.

Here we demonstrate that least squares-based methods taking advantage of both gene and array correlations provide fast and accurate methods for estimating missing values in microarray data. The studied methods are demonstrated to perform better than KNNimpute on three example data sets with 5–25% of the data missing. While KNNimpute finds positively correlated genes by Euclidean distance, the LSimpute methods are able to include negative correlation between genes in the estimation model. In addition, we explore the possibility of exploiting correlation between arrays in estimation and show that this produces superior results in some cases. The success of array correlation-based estimates depends on the similarity of the samples. The stronger the similarity, the more successful we expect this approach to be. With samples taken from very diverse tissues, array correlations will probably provide a weak basis for missing value estimation. Furthermore, we demonstrate that the strengths of the gene- and array-based approaches to estimation can be combined, exemplified by the methods LSimpute_combined and LSimpute_adaptive. Our tests with LSimpute_combined and LSimpute_adaptive on data sets with 10% missing values reveals an RMSD between missing value estimates and the real values that is 15–20% smaller than that obtained using KNNimpute.

To study the impact missing value imputation has on downstream analysis, e.g. clustering, one can use data sets with no missing values and compare clustering results with original data with those obtained using re-estimated data. In order to evaluate the performance of alternative imputation methods on data sets that do include missing values, clustering can be done based on data obtained using the alternative imputation methods and the results evaluated for instance using the method suggested by Gibbons and Roth (16) for comparing gene annotations with clustering systems. Such studies would be informative about how large an impact the choice of imputation method has on cluster analysis.

ACKNOWLEDGEMENTS

We thank Hans A. Karlsen at the Department of Statistics, University of Bergen, for helpful comments. We are also grateful to two anonymous referees for valuable suggestions and comments.

REFERENCES

1. Perou,C.M., Sørlie,T., Eisen,M.B., van de Rijn,M., Jeffrey,S.S., Rees,C.A., Pollack,J.R., Ross,D.T., Johnsen,H., Akslen,L.A. *et al*. (2000) Molecular portraits of human breast tumors. *Nature*, **406**, 747–752.
2. Alizadeh,A.A., Eisen,M.B., Davis,R.E., Ma,C., Lossos,I.S., Rosenwald,A., Boldrick,J.C., Sabet,H., Tran, T, Powell,J.L. *et al*. (2000) Distinct types of diffuse large B-cell lymphoma identified by gene-expression profiling. *Nature*, **403**, 503–511.
3. Golub,T.R., Slonim,D.K., Tamayo,P., Huard,C., Gaasenbeeck,M., Mesirov,J.P., Coller,H., Loh,M.L., Downing,J.R., Caligiuri,M.A. *et al*. (1999) Molecular classification of cancer: class discovery and class prediction by expression monitoring. *Science*, **286**, 531–537.
4. Alon,U., Barkai,N., Notterman,D.A., Gish,K., Ybarra,S., Mack,D. and Levine,A.J. (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl Acad. Sci. USA*, **96**, 6745–6750.
5. Chu,S., DeRisi,J., Eisen,M.B., Mulholland,J., Botstein,D., Brown,P.O. and Hesrkowitz,I. (1998) The transcriptional program of sporulation in budding yeast. *Science*, **278**, 680–686.
6. Gasch,A.P., Spellman,P.T., Kao,C.M., Carmel-Harell,O., Eisen,M.B., Storz,G., Botstein, D and Brown,P.O. (2000) Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Biol. Cell*, **11**, 4241–4257.
7. Brazma,A. and Vilo,J. (2000) Gene expression data analysis. *FEBS Lett.*, **480**, 17–24.
8. Eisen,M.B., Spellman,P.T., Brown,P.O. and Botstein,D. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl Acad. Sci. USA*, **95**, 14863–14868.
9. Tamayo,P., Slonim,D., Mesirov,J., Zhu,Q., Kitareewan,S., Dmitrovsky,E., Lander,E.S. and Golub,T.R. (1999) Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proc. Natl Acad. Sci. USA*, **96**, 2907–2912.
10. Alter,O., Brown,P.O. and Botstein,D. (2000) Singular value decomposition for genome-wide expression data processing and modeling. *Proc. Natl Acad. Sci. USA*, **97**, 10101–10106.
11. Raydachauri,S., Stuart,J.M. and Altman,R.B. (2000) Principal components analysis to summarize microarray experiments: application to sporulation time series. *Pac. Symp. Biocomput.*, 455–466.
12. Troyanskaya,O., Cantor,M., Sherlock,G., Brown,P., Hastie,T., Tibshirani,R., Bostein,D. and Altman,R.B. (2001) Missing value estimation methods for DNA microarrays. *Bioinformatics*, **17**, 520–525.
13. Baldwin,D.N., Vanchianathan,V., Brown,P.O. and Theriot,J.A. (2002) A gene expression program reflecting the innate immune response of intestinal ephithelial cells to infection by *Listeria monocytogenes*. *Genome Biol.*, **4**, R2 (http://genomebiology.com/2002/4/1/R2).
14. Ross,D.T, Scherf,U., Eisen,M.B., Perou,C.M., Rees,C., Spellman,P., Iyer,V., Jeffrey,S.S., Van de Rijn,M., Waltham,M. *et al*. (2000) Systematic variation in gene expression patterns in human cancer cell lines. *Nature Genet.*, **24**, 227–235.
15. Johnson,R.A. and Wichern,D.W. (2002) *Applied Multivariate Statistical Analysis*, 5th edn. Prentice Hall, NJ.
16. Gibbons,F.D. and Roth,F.P. (2002) Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Res.*, **12**, 1574–1581.