BIOINFORMATICS ORIGINAL PAPER Vol. 21 no. 23 2005, pages 4272–4279
doi:10.1093/bioinformatics/bti708

Gene expression

The influence of missing value imputation on detection of differentially expressed genes from microarray data

Ida Scheel[1,*], Magne Aldrin[2], Ingrid K. Glad[1], Ragnhild Sørum[1], Heidi Lyng[3] and Arnoldo Frigessi[2,4]

[1]Department of Mathematics, University of Oslo, PO Box 1053, Blindern, NO-0316 Oslo, Norway, [2]Department of Statistical Analysis, Image Analysis and Pattern Recognition, Norwegian Computing Center, NO-0314 Oslo, Norway, [3]Department of Radiation Biology, The Norwegian Radium Hospital, NO-0310 Oslo, Norway and [4]Department of Biostatistics, University of Oslo, NO-0317 Oslo, Norway

Received on June 9, 2005; revised on September 20, 2005; accepted on October 5, 2005
Advance Access publication October 10, 2005

ABSTRACT

Motivation: Missing values are problematic for the analysis of microarray data. Imputation methods have been compared in terms of the similarity between imputed and true values in simulation experiments and not of their influence on the final analysis. The focus has been on missing at random, while entries are missing also not at random.

Results: We investigate the influence of imputation on the detection of differentially expressed genes from cDNA microarray data. We apply ANOVA for microarrays and SAM and look to the differentially expressed genes that are lost because of imputation. We show that this new measure provides useful information that the traditional root mean squared error cannot capture. We also show that the type of missingness matters: imputing 5% missing not at random has the same effect as imputing 10–30% missing at random. We propose a new method for imputation (LinImp), fitting a simple linear model for each channel separately, and compare it with the widely used KNNimpute method. For 10% missing at random, KNNimpute leads to twice as many lost differentially expressed genes as LinImp.

Availability: The R package for LinImp is available at http://folk.uio.no/idasch/imp

Contact: idasch@math.uio.no

Supplementary information: http://folk.uio.no/idasch/imp

1 INTRODUCTION

Missing values are a predominant problem for the analysis of microarray data, a high throughput technology to evaluate the expression of thousands of genes simultaneously (Lee, 2004). Missing values arise due to technical failure, low signal-to-noise ratio and measurement error (Lee, 2004; Wit and McClure, 2004). Typically ∼1–10% of the data are missing (de Brevern *et al*., 2004), affecting up to 95% of the genes. Many available algorithms for the statistical analysis of microarray data require a full dataset (Wit and McClure, 2004), because the underlying statistical methodology is based on balanced data. This includes for example SAM (Troyanskaya *et al*., 2001, www-stat.stanford.edu/~tibs/SAM), PAM (Tibshirani *et al*., 2001, www-stat.stanford.edu/~tibs/PAM) and ANOVA for microarrays (Kerr *et al*., 2000), implemented in the software MAANOVA (www.jax.org/staff/churchill/labsite/software). Hence all missing values need to be imputed before, e.g. testing for differential gene expression between biological samples. The output of the analysis is seriously influenced by the quality of the applied imputation method: many of the differentially expressed genes are lost, and falsely new differentially expressed genes are generated, compared with the analysis of the true full dataset. Even the robust measures of family wise errors and false discovery rates cannot take this into consideration. In his comment to Sebastiani *et al*. (2003), Gary A. Churchill wrote 'Among the many small problems that have yet to be addressed in microarray analysis, missing data methods stand out in my mind as one of the more pressing'. de Brevern *et al*. (2004) studied the extent of missing values in eight published microarray experiments. There were between 0.8 and 10.6% missing values. Genes with at least one missing value ranged from 3.8 to 94.7%. Kim *et al*. (2005) studied a dataset from Gasch *et al*. (2001) originally containing 6361 genes and 156 experiments. After removing columns that had >8% missing values and removing the genes with one or more missing values, a matrix of dimension 2641×44 remained, a reduction of 88% of the data.

In this paper we concentrate on cDNA microarrays, which are microchips with more than ten thousands of spots each corresponding to a gene. On each spot hybridization of two samples happens, resulting in signals from two channels, one dyed red and the other green. Microarrays are then optically scanned: spots are detected from the background and red and green signal intensities are measured. Missing values originate from imperfections at the level of chip production and treatment, hybridization and scanning. Dust present on the chip, irregularities in the spot production and inhomogeneous hybridization all lead to spots which are manually or automatically flagged, and corresponding signals are then considered as missing. Because probes are printed on spots in random order, without consideration of their expected intensities, spatial noise effects, which are present, cannot be translated into spatially smooth expected intensities. In addition to such signals missing at random, available software flags out signals which cannot be distinguished from the background or have a too irregular form because the signal itself is too low. In these cases, values are missing not at random, the missingness depending on the signal intensity.

*To whom correspondence should be addressed.

 © The Author 2005. Published by Oxford University Press. All rights reserved. For Permissions, please email: journals.permissions@oxfordjournals.org

As a typical example, the AGILENT feature extraction software G2567AA flags out signals when the intensity is extremely low with respect to signal intensities in other spots on the same array (called 'population outliers') or when the local background is highly irregular. Normally a mixture of missing at random and not at random will be present.

K-nearest neighbors (KNNimpute) (Troyanskaya *et al*., 2001) is the most commonly used imputation method. It is the only imputation method implemented in SAM, PAM and MAANOVA, and is therefore routinely applied. KNNimpute has been shown to impute values in a satisfactory way for up to 20% of missing log ratios if missingness is at random, see Troyanskaya *et al*. (2001). Their paper compared the imputed values with the true values in a simulated experiment, where spot ratios were erased at random. The same simulation and validation setup is used to investigate other competing imputation methods, among which are BPCA (Oba *et al*., 2003), LSimpute (Bø *et al*., 2004), GMCimpute (Ouyang *et al*., 2004) and LLSimpute (Kim *et al*., 2005). Feten *et al*. (2005) also compare imputed values with the true values, though in a more refined way than the common root mean squared error (RMSE). While comparing imputed values with the true values is an important measure of performance, it fails to address the more fundamental question of what is the effect of such imputations on the final output of the statistical analysis. Only Ouyang *et al*. (2004) compared the number of mis-clustered genes for different methods.

In this paper we propose a simple and natural imputation method, LinImp, based on a linear model for each channel separately. We investigate its performance and compare it with KNNimpute when values are missing both at random and not at random. In the last case, we model the missingness depending on the signal. We evaluate the method by comparing the resulting list of differentially expressed genes based on the imputed dataset with the same list based on an analysis of the true full dataset. Hence we count how many of the genes in the list are lost and added when analyzing the imputed dataset. In our experiments 47–97% of the differentially expressed genes are lost if nothing is done when 10% of the data are missing. Up to 90% of this is recovered when imputing. KNNimpute shows up to three times as many lost differentially expressed genes as LinImp.

2 SYSTEMS AND METHODS

2.1 LinImp: linear model-based imputation

The imputation method we propose, LinImp, is based on the linear model for y_{ijkg}, the base 2 logarithm of the intensity in array i, channel (dye) j, variety k and gene g

$$y_{ijkg} = \mu + A_i + D_j + G_g + AD_{ij} + AG_{ig} + DG_{jk} + VG_{kg} + \varepsilon_{ijkg}, \quad (1)$$

where ε_{ijkg} are independent normally distributed error terms with mean zero and variance σ^2. For simplicity we assume that each gene is printed only once on each array, such that one gene is represented by only one spot on each array. The varieties are the experimental conditions under study, such as for example type of tissue. If we have a arrays, 2 channels (dyes) on each array, v varieties and N genes, then $i = 1, \ldots, a$, $j = 1, 2$, $g = 1, \ldots, N$ and $k = 1, \ldots, v$ and there are $2aN$ observations. μ is the overall mean, A_i is the effect of array i, D_j is the effect of dye j, G_g is the overall effect of gene g, AD_{ij} is the interaction between array i and dye j, AG_{ig} is the interaction between array i and gene g, DG_{jg} is the interaction between dye j and gene g and VG_{kg} is the interaction between variety k and gene g. Model (1) was proposed in Kerr *et al*. (2000) to find differentially expressed genes. Written in matrix form the model is

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \quad (2)$$

where $\boldsymbol{y}$ is a vector of length $2aN$ and $\boldsymbol{X}$ is a matrix of zeros and ones. Denote $\boldsymbol{A}^{\mathrm{T}} = (A_1, \ldots, A_a)^{\mathrm{T}}$, $\boldsymbol{D}^{\mathrm{T}} = (D_1, D_2)^{\mathrm{T}}$ etc., then $\boldsymbol{\beta} = (\mu, \boldsymbol{A}^{\mathrm{T}}, \boldsymbol{D}^{\mathrm{T}}, \boldsymbol{G}^{\mathrm{T}}, \boldsymbol{AD}^{\mathrm{T}}, \boldsymbol{AG}^{\mathrm{T}}, \boldsymbol{DG}^{\mathrm{T}}, \boldsymbol{VG}^{\mathrm{T}})^{\mathrm{T}}$. Each pair of i and j corresponds to only one variety k. Because of this, the effects V and AD are confounded, so it is wise to have only one of the two in the model. We chose (1) with AD instead of V because it saturates the design space (Kerr *et al*., 2002).

LinImp works as follows. Let $\boldsymbol{Y}$ be the $N \times 2a$ observed data matrix with observed and missing values. We initialize the imputation (e.g. we used KNNimpute) with a full data matrix $\boldsymbol{Y}^0$. Then we estimate the parameter vector $\boldsymbol{\beta}$ in model (2) using the dataset $\boldsymbol{Y}^0$. Denote this estimated parameter vector as $\hat{\boldsymbol{\beta}}^0$. Next, we impute the missing values in $\boldsymbol{Y}$ with their expected values using (2) and $\hat{\boldsymbol{\beta}}^0$ to obtain the new full data matrix $\boldsymbol{Y}^1$. We iterate the procedure until convergence, for example until at iteration M in some norm $\|\boldsymbol{Y}^M - \boldsymbol{Y}^{M-l}\| < \delta$, where δ is a fixed small value. The full data matrix $\boldsymbol{Y}^{\mathrm{imp}} = \boldsymbol{Y}^M$ is then the final imputed dataset. The pseudocode of the algorithm is given in Figure 1. Running LinImp requires a couple of minutes for the largest dataset in this paper. LinImp is practically an automatic imputation method. Of course the small value δ must be chosen, but to our experience the results are very robust with respect to this choice. Also, KNNimpute is a reasonable and simple choice for initial imputation. Linear model imputation in general has been proposed before, see for instance Pyle (1999). Notice that estimating $\boldsymbol{\beta}$ in (2) is easy when data are complete because then it is possible to simplify the estimation algorithm. In the case of missing values, and hence unbalanced design, the estimation of $\boldsymbol{\beta}$ becomes a formidable computational task.

1: Linear model: $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \varepsilon$, where $\varepsilon \sim N(0, \sigma^2)$
   $\boldsymbol{Y}$: Observed data matrix of dimension $N \times 2a$ with $2aN - n$ observed values and n missing values.
   k, l: Vectors such that the jth missing value in $\boldsymbol{Y}$ is located in the k_jth row and the l_jth column.
   δ: The convergence criterion, preset in the code.
2: Initialize: $\boldsymbol{Y}^0$ (imputed data set)
   (Initial imputation done by for example KNNimpute)
3: Initialize: $i \leftarrow 1$
4: Initialize: convergence=FALSE
5: **while** convergence=FALSE **do**
6: $\quad \boldsymbol{Y}^i \leftarrow \boldsymbol{Y}$
7: $\quad$ Fit the linear model for $\boldsymbol{Y}^{i-1}$ to obtain $\widehat{\boldsymbol{\beta}}^{i-1}$
8: $\quad$ **for** $j = 1$ to n **do**
9: $\quad\quad \boldsymbol{Y}^i_{k_j l_j} \leftarrow E[\boldsymbol{Y}^{i-1}_{k_j l_j} | \boldsymbol{\beta}^{i-1} = \widehat{\boldsymbol{\beta}}^{i-1}]$
10: $\quad$ **end for**
11: $\quad$ **if** $\|\boldsymbol{Y}^i - \boldsymbol{Y}^{i-1}\| < \delta$ then
12: $\quad\quad$ convergence=TRUE
13: $\quad\quad \boldsymbol{Y}^{\mathrm{imp}} \leftarrow \boldsymbol{Y}^i$
14: $\quad$ **else**
15: $\quad\quad i \leftarrow i + 1$
16: $\quad$ **end if**
17: **end while**

Fig. 1. Pseudocode for the algorithm for LinImp.

2.2 KNNimpute

KNNimpute (Troyanskaya *et al*., 2001) is widely used, for instance it is the only imputation method available in SAM, PAM and MAANOVA. Hence it is important to analyze the effect KNNimpute has on detecting differentially

expressed genes. KNNimpute works as follows. For each row i in the data matrix, corresponding to gene g, with one or more missing values, the k nearest neighbor rows are found. It is necessary that the k nearest neighbors have data in the columns where row g had missing data. To define a neighborhood structure between rows, a metric is necessary. The distance $d_{gg'}$ from row g to row g' is the Euclidean distance of the two vectors omitting the entries for which row g and row g' have missing values. If there are one or more missing entries in row g' in places where row g has non-missing entries, the squared difference for these entries is set to the average of the squared difference for the non-missing entries. When the k nearest neighbors are found, the missing entry in column c in row g is imputed as the weighted average of the values in column c in the k-nearest neighbor rows, the weights being the inverse distances.

There seems to be some confusion in the literature about the KNNimpute algorithm. Some authors (Lee, 2004; Ouyang *et al*., 2004) describe an algorithm where neighbors are not allowed to have any missing values. This can create problems in datasets with a lot of missing values because only a few, or none, neighbors actually are free of missing values and the imputation becomes impossible or poor. Others, for instance Oba *et al*. (2003), describe algorithms where the neighbors are allowed to have missing values, but the corresponding missing differences are not imputed when calculating the distance $d_{gg'}$. This will cause falsely low distances for neighbors with a lot of missing values. These versions of KNNimpute are too simplistic and less efficient than full KNNimpute, which is used in our comparisons. In this paper we use the KNNimpute implementation available in the R package `impute` (by Hastie, T., Tibshirani, R., Narasimhan, B. and Chu, G., available at http://cran.r-project.org/). This implementation does not suffer from any of the abovementioned weaknesses.

While traditionally KNNimpute is applied to the $N \times a$ data matrix of log ratios of intensities, in this paper we apply KNNimpute to the $N \times 2a$ data matrix of log intensities.

2.3 Detecting differentially expressed genes

There are various ways of detecting differentially expressed genes. In this paper we use two such common approaches to evaluate imputation. When using the linear model (1), the quantity of interest is $VG_{1g} - VG_{2g}$ for determining if gene g is differentially expressed between varieties 1 and 2. For example, $VG_{1g} - VG_{2g} = 1$ equals a 2-fold change between the two tissues, because y_{ijkg} is the base 2 logarithm of the intensity. The linear model (1) is presented in Kerr *et al*. (2000) as ANOVA for microarrays, and it is implemented in MAANOVA. An alternative approach is based on hypothesis testing directly on the matrix of log ratios, as done by Significance Analysis of Microarrays (SAM) (Tusher *et al*., 2001).

2.4 Data

We have used two spotted cDNA microarray datasets for exploring the new imputation method and the overall recovery rate of missing differentially expressed genes. LinImp imputes missing values separately in each channel, and hence uses the channel data directly, not their log ratios. Such data are more rarely published. The first dataset is based on a study of human cell lines and the other dataset is typical for clinical studies on primary tumors. The intensities and log ratios are generally higher in experimental studies on cell lines than in clinical studies based on primary tumors.

The dataset based on human cell lines is composed of three dye-swaps, thus six arrays. The data are from the NIEHS experiment comparing treated and control human cell lines, as described in Kerr *et al*. (2002). It is publicly available at http://www.jax.org/staff/churchill/labsite/datasets/expression/niehs. The dataset based on primary tumors is based on samples from cervical tumors before and after radiotherapy and is composed of 16 dye-swaps and thus 32 arrays and is available from our Supplementary information web page. In the NIEHS dataset there were 1907 genes and no missing values, thus a full intensity data matrix of dimension 1907×12. In the original cervical cancer dataset 22% of the data were missing, affecting 70% of the 14 229 genes. We have removed the genes with one or more missing values, leaving data from 4246 genes. The resulting intensity data matrix is of dimension 4246×64. These truths are then used as bases for simulating missing values.

When genes with at least one missing value are removed from the analysis, the effect can be dramatic. When missing at random, the percentage of lost differentially expressed genes will be approximately the same as the percentage of genes with one or more missing values. When missing not at random, the percentage of lost differentially expressed genes can be even higher since genes that are differentially expressed are more likely than others to have missing values.

2.5 More realistic models of the missingness

A value in a data matrix is missing at random, MAR, or missing completely at random (MCAR), if the probability of it being missing does not depend on the value that is missing. A value is missing not at random, MNAR, if the probability of it being missing is dependent on the value that is missing. When missing at random, usually both channel signals from the spot are missing, which means that we are in the MAR situation. A basic reason for microarray data to be missing not at random is that the foreground intensity is lower than the background intensity. Another reason is that low intensities are per se sometimes considered too noisy and conservatively flagged out. Missing not at random happens most often for just one of the two channel signals in a spot. Of course this means that when analyzing log ratios the whole spot is missing. We have separated the analysis of imputation of values missing at random and not at random in order to see differences in the effects of imputation for the two types of missing.

When simulating datasets with values missing at random, we have assumed both channels from the spot to be missing. Therefore, to simulate a total of r% of the entries missing at random, we have drawn $r/2$% of the spots at random and made both channel signals missing. For both the cervical cancer data and the NIEHS data we have chosen the missing percentages r to be 1, 5, 10, 15, 20, 25, 30, 35 and 40. We have simulated 50 independent missing datasets for each percentage missing.

Our mechanism creating values missing not at random favors missingness of low intensities. We proceed as follows. For each spot the lowest of the two signals is considered. These lowest signals are ordered and the s% percentile is found, say 5%. We then produce a dataset with r% of the total number of entries missing (say 1%) by drawing at random exclusively from below the s% percentile and making the lowest channel signal from these spots missing. A histogram of the lowest base 2 log intensity for each spot for both datasets can be seen in Figure 2. On the basis of this we have chosen the threshold $s = 25$ for the NIEHS dataset and for the cervical cancer dataset the threshold $s = 5$. For the NIEHS dataset the missing percentages r run over 1, 2.5, 4, 5.5, 7, 8.5, 10, 11.5 and 13. For the cervical cancer dataset the missing percentages r run over 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5 and 5.

2.6 Measures of performance of imputation methods

Imputation methods for microarray data are discussed in the literature in terms of the RMSE, where the error is the difference between the imputed value and the true one. Troyanskaya *et al*. (2001) normalize the RMSE by dividing it by the average value over all observations in the true full dataset, which is useful because it enables comparisons across different datasets. This is denoted by NRMSE and is adopted in this paper. Oba *et al*. (2003) and Kim *et al*. (2005) normalize the RMSE by dividing it by the standard deviation of the values in the true full dataset. This measure is denoted here by NRMSE2. Ouyang *et al*. (2004) normalize the RMSE by dividing it by the root mean square of all the observations in the true full dataset. This measure is denoted here by NRMSE3. Bø *et al*. (2004) do not normalize.

Unfortunately none of the various RMSE measures describe the real effect of imputation on the final analysis. We are interested in evaluating the effect imputation has on the final output of the statistical analysis in question, and a different measure is needed. A typical end-product of a statistical analysis is a list of interesting genes. How is such a list affected by the errors of imputation? A way to produce such a list is by hypothesis

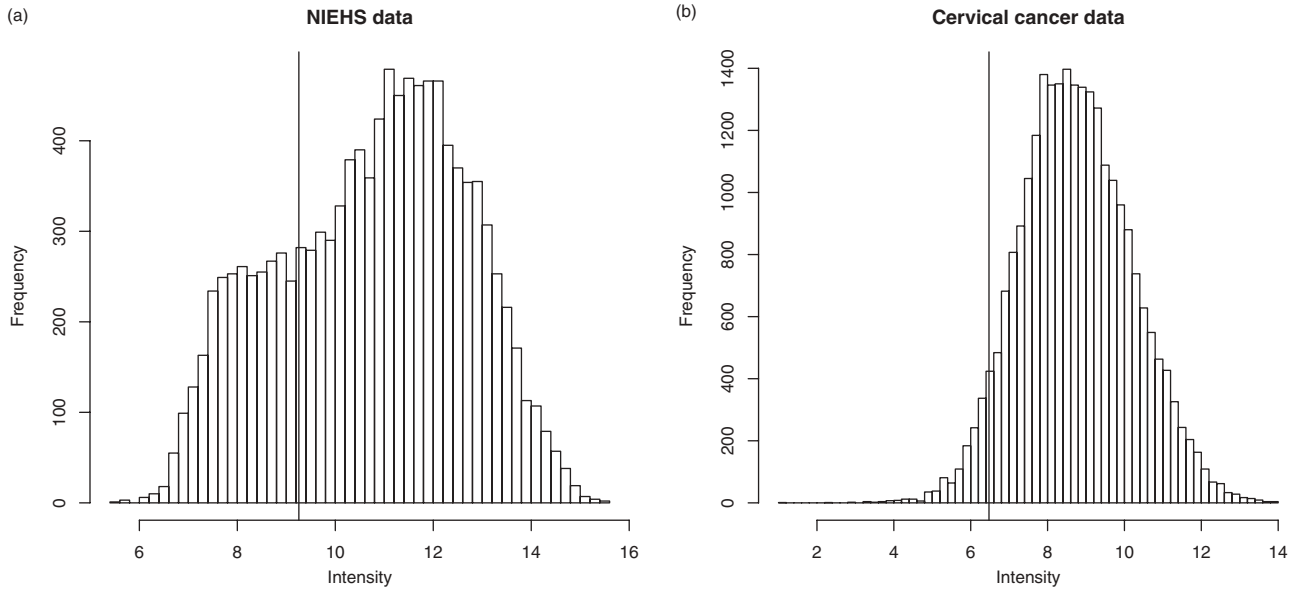

Fig. 2. A histogram of the lowest log2 intensity for each spot for the NIEHS dataset with a vertical line indicating the 25% quantile (**a**) and a histogram of the lowest log2 intensity for each spot for the cervical cancer dataset with a vertical line indicating the 5% quantile (**b**).

testing, using for example the linear model (1) as in Kerr *et al*. (2000). Here we measure the success of imputation by looking to lost and added differentially expressed genes compared with the list of differentially expressed genes from an analysis of the true full dataset. That is we look for genes which would be on the list if we knew the true full dataset but are lost due to imputation errors and genes which enter the list by mistake again as an effect of imputation errors.

When using the linear model (1) to analyze a dataset with two different varieties, we test for each gene g if $VG_{1g} - VG_{2g}$ significantly differs from 0. To facilitate comparison of methods, we fixed the length of the list of differentially expressed genes for both datasets to 100. When the list length is fixed, the numbers of lost and added differentially expressed genes are the same.

In addition we evaluate the effect of intensity-based imputation when analyzing ratio datasets. For that we used the methods of Tusher *et al*. (2001) implemented in SAM. When using SAM for testing on one-class data each gene is assigned a score, the average log ratios for that gene divided by a sum of the standard deviation for that gene and a small positive constant. We also here fixed the length of the lists to 100. This gave an estimated FDR of 1% for the NIEHS dataset.

The lists of length 100 of differentially expressed genes for the true full NIEHS dataset when analyzing using the linear model and using SAM agree for 97 genes.

3 RESULTS

In Figure 3 we compare the percentage of lost differentially expressed genes when analyzing the datasets using the linear model (1). The figure is based on the average of the 50 runs, and results for both LinImp and KNNimpute are shown. The ratios between the average percentage lost genes for KNNimpute and LinImp can be seen at the top of the plots. The averages of the percentages of potentially lost differentially expressed genes are also shown at the top of the plots. That is the differentially expressed genes based on the true full dataset that have one or more missing values in the simulated dataset and thus are lost if genes with one or more missing values were deleted.

Imputing missing values clearly makes a vast improvement on identifying differentially expressed genes with the linear model (1). For the NIEHS dataset when 10% of the data are missing at random, at least 47% of the differentially expressed genes would be missing if instead of imputing, genes with one or more missing values were deleted. By imputing with KNNimpute 80.8% of these genes are recovered and 89.4% by imputing with LinImp. When 10% of the NIEHS data are missing not at random, at least 48.3% of the differentially expressed genes would be missing if genes with one or more missing values were deleted. By imputing with KNNimpute 75.1% of these genes are recovered and 83.4% by imputing with LinImp. For the cervical cancer dataset when 10% of the data are missing at random, at least 97.1% of the differentially expressed genes would be missing if genes with one or more missing values were deleted. By imputing with KNNimpute 89.7% of these genes are recovered and 88.7% by imputing with LinImp. When 5% of the cervical cancer data are missing not at random, at least 62% of the differentially expressed genes would be missing if genes with one or more missing values were deleted. By imputing with KNNimpute 62.9% of these genes are recovered and 74.2% by imputing with LinImp.

For the NIEHS data, LinImp outperforms KNNimpute for all missing percentages, for both missing at random and missing not at random. KNNimpute shows 50–100% more lost differentially expressed genes than LinImp. For the cervical cancer data, the results are quite similar for LinImp and KNNimpute for missing at random, but for missing not at random KNNimpute shows 20–40% more lost differentially expressed genes than LinImp. Of course, LinImp might have an advantage with respect to KNNimpute since analysis is done by the same model used for imputation. Still, if the linear model is to be used for the analysis, the comparison is fair. Because of the possible advantage LinImp has when analyzing by the linear model, we also did an analysis using SAM, which is done on log ratio data. As an example of alternatives to KNNimpute, here we also imputed using LSimpute

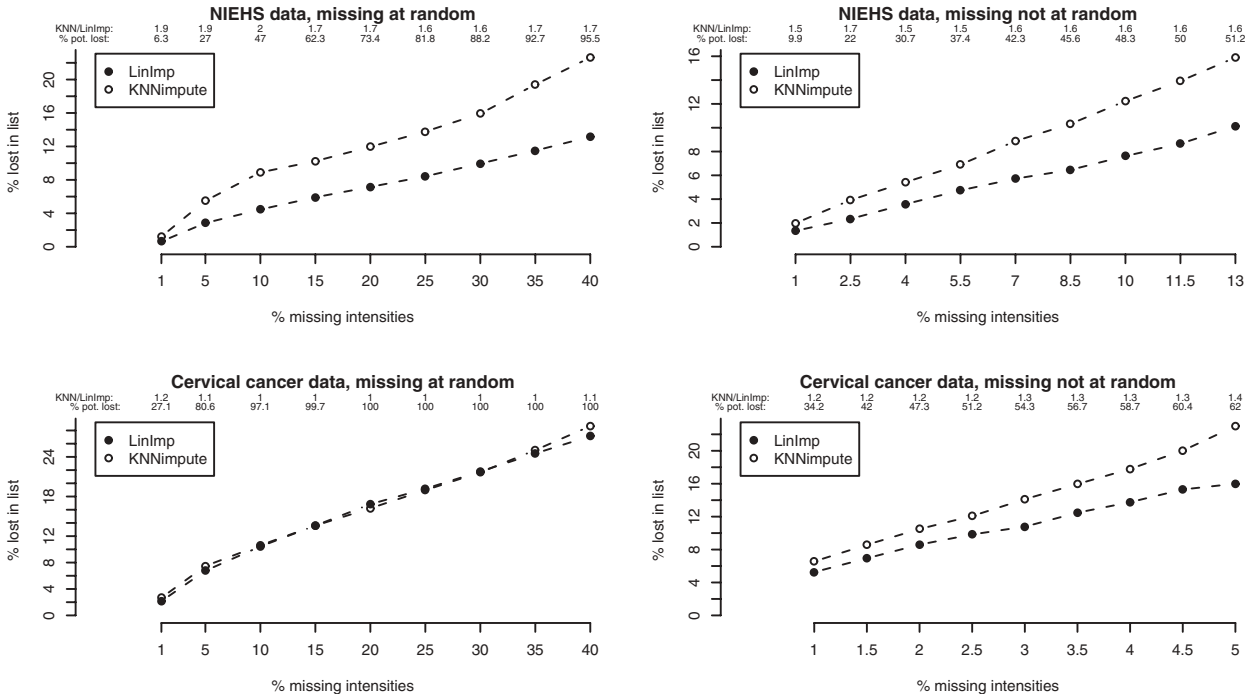

Fig. 3. Percentage lost differentially expressed genes when analyzing the datasets by using the linear model, the averages of the 50 runs. At the top of each plot ratios between the average percentage lost for KNNimpute and LinImp are shown, as well as the average percentage potentially lost differentially expressed genes when imputation is not done.

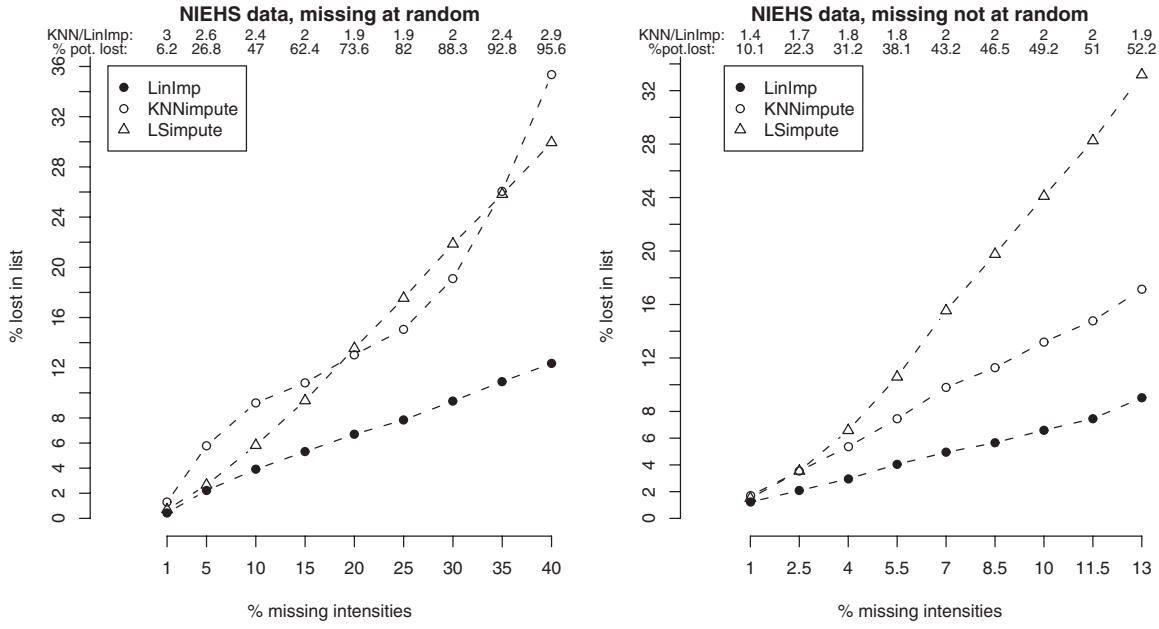

Fig. 4. Percentage lost differentially expressed genes when analyzing the NIEHS data by using SAM, the averages of the 50 runs. At the top of the plots ratios between the average percentage lost for KNNimpute and LinImp are shown, as well as the average percentage potentially lost differentially expressed genes when imputation is not done.

(Bø *et al*., 2004). LSimpute is an imputation method for log ratio data which utilizes the correlation structure. It is based on regression with non-missing log ratios as explanatory variables. Note that LinImp is based on a different type of regression model with the explanatory variables describing the experimental conditions. In Figure 4 we compare the percentage of lost differentially expressed genes when analyzing the NIEHS dataset using SAM. LinImp performs better than KNNimpute for the NIEHS data also when

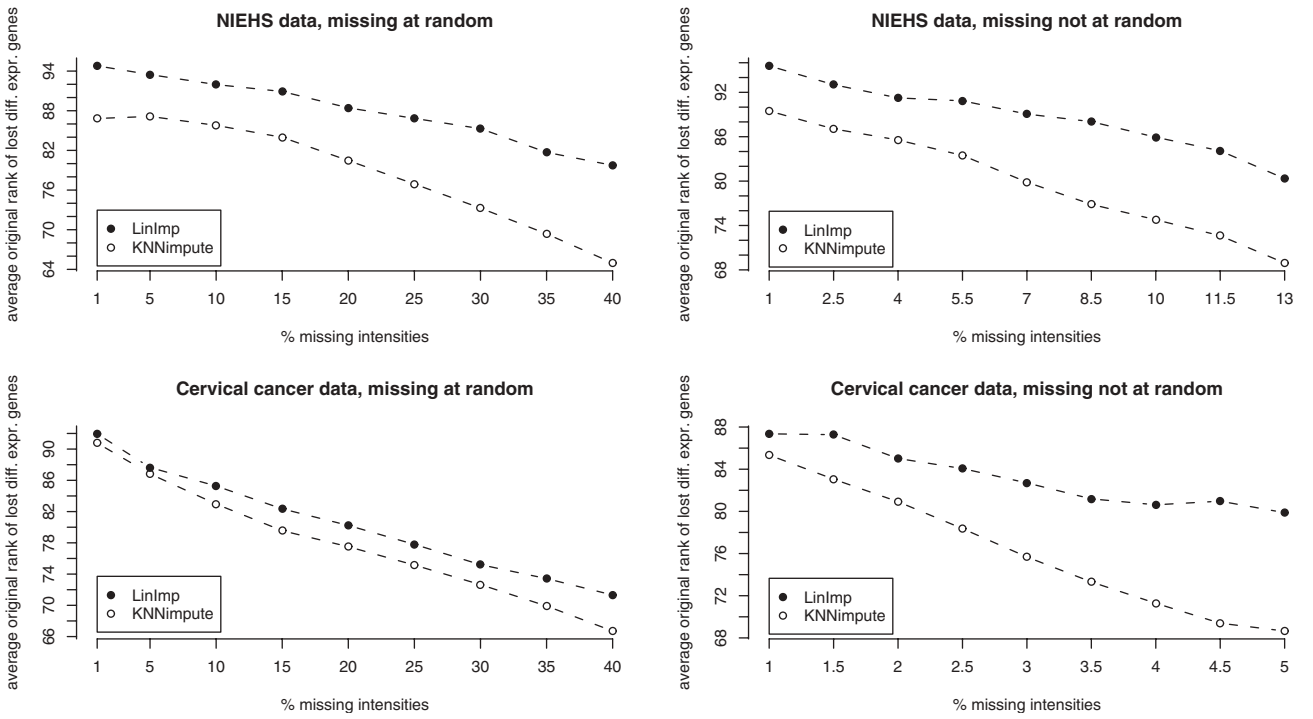

Fig. 5. The average of the rank the genes that are lost have in the list based on the true full dataset (original rank), when analyzing by using the linear model. The plot shows the average of the 50 runs.

analyzing with SAM instead of the linear model. LSimpute shows better results than KNNimpute for low percentages missing at random, but worse for missing not at random.

For the same percentage of missing, the percent lost differentially expressed genes is higher for missing not at random than for missing at random. This is the case for both analysis methods and both imputation methods. For the NIEHS data the results for 10% missing not at random are approximately the same as the results for 20% missing at random, for both imputation methods. For the cervical cancer data the results for 5% missing not at random are approximately the same as the results for 10% missing at random when imputing with LinImp. When imputing with KNNimpute the results for 5% missing not at random are approximately the same as the results for 30% missing at random. The reason for the difference between missing at random and not at random is that in simulating missing not at random genes that have low values have a higher probability of having missing values than other genes. When a gene that is differentially expressed when comparing two varieties is very low expressed for one of the two varieties, all the intensities of that particular gene for that variety, that is half of all the intensities for that gene, are likely to be very low and thus missing at the same time. This results in a lot of imputed values for that gene and makes it more vulnerable in the analysis. It also indicates that imputing low values is difficult for both imputation methods, even though LinImp does a better job than KNNimpute for missing not at random for both analysis methods.

How serious is the loss? Where are the genes that are lost located in the list of differentially expressed genes based on the true full dataset? In Figure 5 we have plotted the position of the lost genes in the list based on the true full dataset when analyzing using the linear model. Specifically we plot the average position the lost genes would have had in the list of differentially expressed genes if there were no missing values. The figure shows the average of the 50 runs for each percent missing. Rank 1 means top of the list and most significant and rank 100 means bottom of the list and least significant, thus the lower the number the more serious the loss is. As expected the curves decrease with increasing percentage. The fact that the curves of LinImp are always above those of KNNimpute shows that LinImp performs better than KNNimpute, which confirms Figure 3. Figure 5 also shows that the loss is more serious for missing not at random than missing at random for the same percentage missing. There is a dependency between Figures 5 and 3, because the more genes lost the higher the average position in the original list, but Figure 5 provides additional information. For example, for 1% missing at random and missing not at random in the NIEHS dataset, there is a clear difference between LinImp and KNNimpute in favor of LinImp, whereas in Figure 3 there is no difference. This means that the genes that are lost when imputing with KNNimpute are more significant in the original list than those that are lost when imputing with LinImp. The loss is less serious using LinImp than KNNimpute.

Since most studies use RMSE values in the evaluation of imputation performance, we have plotted the average of the NRMSE values for the 50 runs in Figure 6. Results for both LinImp and KNNimpute are shown. Also here LinImp outperforms KNNimpute. For the same percentage of missing, the NRMSE values for the cervical cancer data are better for missing at random than missing not at random, which coincides with the results from the lost differentially expressed genes. Still, the conclusions from Figures 3 and 4 are somewhat different from those of Figure 6. For the NIEHS

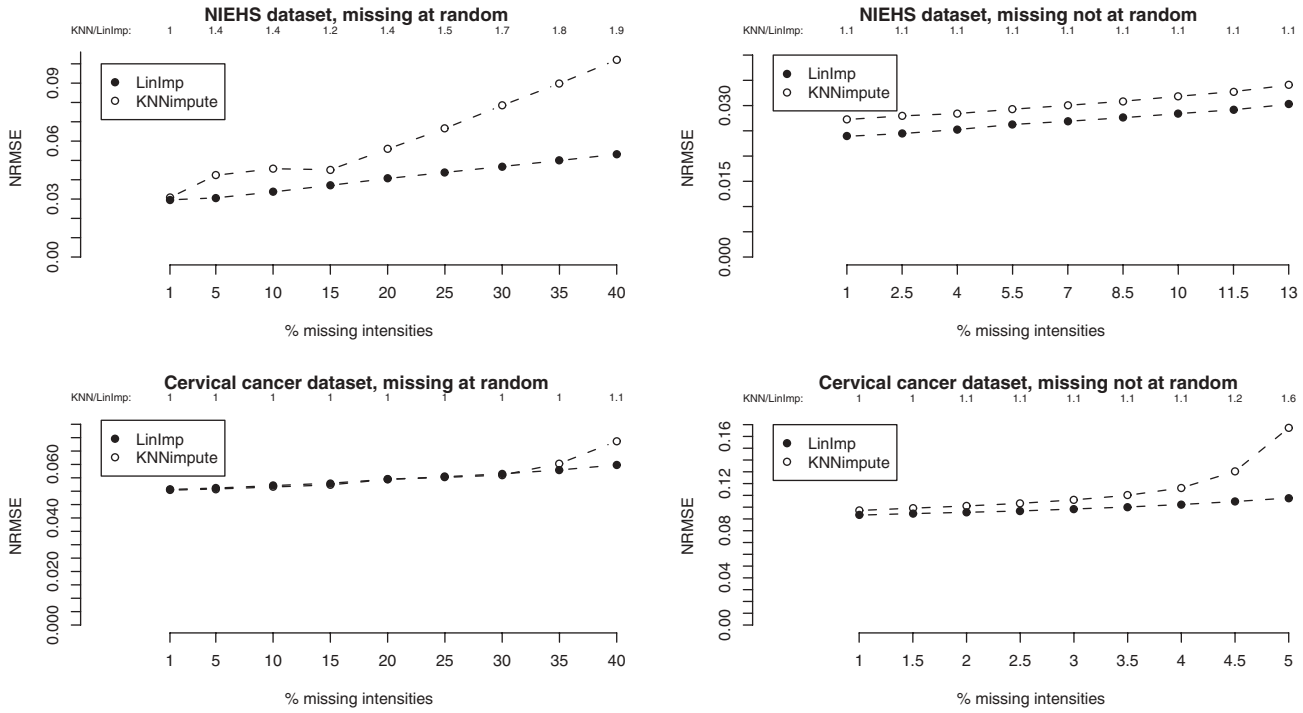

Fig. 6. Normalized RMSE, averages of the 50 runs.

data the NRMSE values are actually a bit better for missing not at random than missing at random. Whereas NRMSE is quite stable for both LinImp and KNNimpute for both datasets for missing not at random and for the cervical cancer data for missing at random, the increase in percentage lost is more dramatic. Also, the NRMSE values are 10% higher for KNNimpute than LinImp for the NIEHS data for missing not at random, whereas Figure 3 using the linear model implies that KNNimpute leads to up to 60% more lost differentially expressed genes than LinImp and Figure 4 using SAM implies that KNNimpute leads to up to twice as many lost differentially expressed genes than LinImp. The new way of evaluating imputation performance thus seems to provide useful information that NRMSE cannot capture.

Almost all the differences between the imputation methods in Figures 3, 4 and 6 are significantly larger than 0, even when correcting for multiple testing. This means that LinImp is significantly better than KNNimpute most of the time. The only exception is 1% missing and some of the other percentages missing at random for the cervical cancer data, which is not surprising in light of the figures. KNNimpute is never significantly better than LinImp. The average of the differences between the results of the imputation methods together with their estimated standard errors can be seen in Tables 1–3 available on our Supplementary information web page.

4 DISCUSSION AND CONCLUSION

Though KNNimpute is the most used imputation method several alternatives have been proposed, all for log ratio datasets. Oba *et al*. (2003) present a Bayesian principal component analysis approach, BPCA, based on an EM-like algorithm. Datasets with 1–20% entries missing at random are simulated from original full datasets and the performance is evaluated by computing the NRMSE2. BPCA shows better NRSME2 values than KNNimpute when the number of samples is large, though possibly a suboptimal version of KNNimpute was used. When the number of samples is <40 KNNimpute performs equally well or better than BPCA. Also, for one dataset the NRMSE2 for the BPCA is tripled from 1 to 20% missing values, while KNNimpute is much more stable. Zhou *et al*. (2003) investigate imputation based on linear and non-linear regression with Bayesian gene selection. The results are better for both versions compared with KNNimpute, though only 1 and 5% of missing data are investigated. Bø *et al*. (2004) show 15–20% smaller RMSE values for LSimpute than for KNNimpute for 10% entries missing at random. Ouyang *et al*. (2004) impute with GMCimpute, modeling data with a Gaussian mixture and using the EM algorithm. The number of mixture components is determined empirically. The simulation includes only very low missing probabilities in the range 0.003–0.04. The performance is evaluated by computing NRMSE3 and the number of mis-clustered genes. GMCimpute shows better results than KNNimpute, but the version of KNNimpute utilized requires the neighbors to be complete. It is possible to improve on this, so that KNNimpute could have performed better. Nguyen *et al*. (2004) compare KNNimpute to imputation via OLS and PLS regression with other genes as explanatory variables. The methods are evaluated by looking to the relative estimation error as a function of the true expression value. KNNimpute performs best near the median of the true expression values, while PLS seems best for the more extreme expression values. Kim *et al*. (2005) introduce a local least squares imputation method, LLSimpute, imputing a missing value for a gene by a linear combination of similar genes. It is called local because it uses only the most similar genes.

The method differs from LSimpute in that they use also the L_2-norm for determining similarity between genes, while LSimpute uses only the Pearson correlation. LLSimpute shows lower NRMSE2 values than KNNimpute and BPCA with 1–20% entries missing. Specifications on KNNimpute do not allow to understand whether KNNimpute has been implemented at best. Feten *et al*. (2005) investigate six imputation methods, four based on regression with other genes as explanatory variables and KNNimpute both with genes and observations as neighbors. The conclusion is that for datasets with strong correlation structure, KNNimpute with genes as neighbors performs best. LinImp outperforms KNNimpute when the linear model captures linear relationships within the log intensities that KNNimpute cannot capture. As Feten *et al*. (2005) concluded KNNimpute works well for highly correlated data. A possible improvement on LinImp for such datasets is to exploit the correlation between genes. When assuming uncorrelated data, as in LinImp, the expectation of the error term conditioned on information from other genes is 0. If the correlation structure would be considered, the multinomial distribution would give a conditional expectation for the error term different from 0. Conditioning should be done on information from other genes that are highly correlated with the gene which value is to be imputed. For computational reasons one cannot condition on information from all the other genes, thus it would not be completely automatic, since the number of correlated genes to consider when imputing for another gene would need to be decided.

In addition to erasing data at random Nguyen *et al*. (2004) also have an experiment where the probability of a gene having a missing value depends on the expression level. The conclusion is that the results are similar to the missing at random case. We have seen that imputing values that are missing not at random has a more serious effect on the final analysis than imputing values that are missing at random. Of course, in reality the missingness in a microarray dataset is a mixture of missing at random and missing not at random. We have introduced a new imputation method, LinImp. In most of our experiments we have found that LinImp performs better than the widely used KNNimpute, in particular when comparing resulting lists of differentially expressed genes. Finally, we conclude that looking to the actual effect imputation has on the final analysis gives valuable information in addition to the traditional RMSE.

Conflict of Interest: none declared.

REFERENCES

de Brevern,A.G. *et al*. (2004) Influence of microarrays experiments missing values on the stability of gene groups by hierarchical clustering. *BMC Bioinformatics*, **5**, 114.

Bø,T.H. *et al*. (2004) LSimpute: accurate estimation of missing values in microarray data with least squares methods. *Nucleic Acids Res*., **32**, e34.

Feten,G. *et al*. (2005) Prediction of missing values in microarray and use of mixed models to evaluate the predictors. *Stat. Appl. Genet. Mol. Biol*., **4**, 10.

Gasch,A.P. *et al*. (2001) Genomic expression responses to DNA-damaging agents and the regulatory role of the yeast ATR homolog Mec1p. *Mol. Biol. Cell*, **12**, 2987–3003.

Kerr,M.K. *et al*. (2000) Analysis of variance for gene expression microarray data. *J. Comput. Biol*., **7**, 819–837.

Kerr,M.K. *et al*. (2002) Statistical analysis of a gene expression microarray experiment with replication. *Stat. Sinica*, **12**, 203–217.

Kim,H. *et al*. (2005) Missing value estimation for DNA microarray gene expression data: local least squares imputation. *Bioinformatics*, **21**, 187–198.

Lee,M.-L.T. (2004) *Analysis of Microarray Gene Expression Data*. Kluwer Academic Publishers, MA.

Nguyen,D.V. *et al*. (2004) Evaluation of missing value estimation for microarray data. *J. Data Sci*., **2**, 347–370.

Oba,S. *et al*. (2003) A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics*, **19**, 2088–2096.

Ouyang,M. *et al*. (2004) Gaussian mixture clustering and imputation of microarray data. *Bioinformatics*, **20**, 917–923.

Pyle,D. (1999) *Data Preparation for Data Mining*. Morgan Kaufmann Publishers, Inc., San Francisco, CA, pp. 275–297.

Sebastiani,P. *et al*. (2003) Statistical challenges in functional genomics. *Stat. Sci*., **18**, 33–70.

Tibshirani,R. *et al*. (2002) Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc. Natl Acad. Sci. USA*, **99**, 6567–6572.

Troyanskaya,O. *et al*. (2001) Missing value estimation methods for cDNA microarrays. *Bioinformatics*, **17**, 520–525.

Tusher,V.G. *et al*. (2001) Significance analysis of microarrays applied to the ionizing radiation response. *Proc. Natl Acad. Sci. USA*, **98**, 5116–5121.

Wit,E. and McClure,J. (2004) *Statistics for Microarrays: Design, Analysis and Inference*. John Wiley and Sons Ltd, West Sussex, England, pp. 65–69.

Zhou,X. *et al*. (2003) Missing-value estimation using linear and non-linear regression with Bayesian gene selection. *Bioinformatics*, **19**, 2302–2307.