MICROARRAY MISSING VALUE IMPUTATION BY ITERATED LOCAL LEAST SQUARES *

ZHIPENG CAI†, MAYSAM HEYDARI †, AND GUOHUI LIN ‡

Bioinformatics Research Group, Department of Computing Science, University of Alberta. Edmonton, Alberta T6G 2E8, Canada.

Emails: zhipeng,maysam,ghlin@cs.ualberta.ca

Microarray gene expression data often contains missing values resulted from various reasons. However, most of the gene expression data analysis algorithms, such as clustering, classification and network design, require complete information, that is, without any missing values. It is therefore very important to accurately impute the missing values before applying the data analysis algorithms. In this paper, an *Iterated Local Least Squares Imputation method* (ILLsimpute) is proposed to estimate the missing values. In ILLsimpute, a similarity threshold is learned using known expression values and at every iteration it is used to obtain a set of coherent genes for every target gene containing missing values. The target gene is then represented as a linear combination of the coherent genes, using the least squares. The algorithm terminates after certain iterations or when the imputation converges. The experimental results on real microarray datasets show that ILLsimpute outperforms three most recent methods on several commonly tested datasets.

Keywords: Microarray gene expression data, Missing value imputation, Local least squares

1. Introduction

DNA microarray experiments are extensively used to monitor the expression of a large amount of genes under various conditions. Associated with mathematical analysis methods, DNA microarray has important applications in biological and clinical studies. The data generated from a set of microarray experiments is usually expressed as a large matrix, with the expression levels of genes in rows, and the experimental conditions ordered in columns.[7] In other words, let $G_{m\times n} = (g_{ij})_{m\times n}$ be the expression matrix of m genes in n experiments. Then g_{ij} records the expression level of the i-th gene in the j-th experiment. One frequent issue that affects microarray data analysis is the existence of missing values, that is, matrix $G_{m\times n}$ could contain many entries with unknown expression levels. A number of reasons could lead to the missing data, including insufficient resolution, image corruption, or even dust and scratches on the slides.[7] On the other hand, most of the microarray data analysis algorithms, such as gene clustering, disease (experiment) classification, and gene network design, require the complete information. In other words, they

*This work is supported by NSERC and CFI.

†Z.C. and M.H. contributed equally to this work.

‡To whom correspondence should be addressed. Fax: (780) 492-1071. Email: ghlin@cs.ualberta.ca.

require matrix $G_{m\times n}$ contains no missing values. It is therefore very important to accurately estimate the missing values in matrix $G_{m\times n}$, if any, before we can apply the data analysis algorithms. One straightforward solution is to repeat the experiments a sufficient amount of times[1,7] and use a combination of them to obtain a complete expression matrix. It is easily seen that such an approach is very costly and inefficient. Moreover, a portion of expression information would be wasted.

There are several proposals on effectively imputing the missing values, without extra experiments, through taking advantage of modern mathematical and computational techniques. To name a few, Troyanskaya *et al.*[7] proposed a *weighted K-nearest neighbor* method (KNNimpute) and a *singular value decomposition* method (SVDimpute) to impute the missing values. In more details, KNNimpute method selects for every target gene K nearest neighboring genes from the entire set of genes (one measure of distance will be detailed in Section 2). It then uses weighted linear combinations of these neighboring genes to predict the missing values in the target gene. In SVDimpute method, from the expression matrix $G_{m\times n}$, a set of mutually orthogonal expression patterns (called *eigengenes*) is obtained that can be linearly combined to approximate the expression levels of all the genes. Subsequently, K most significant eigengenes are selected to estimate the missing values. It has been shown[7] that KNNimpute works well on static microarray data and noisy time series microarray data; SVDimpute performs better on time series microarray data with low noise levels.[7] In year 2003, Oba *et al.*[4] proposed a novel missing value estimation method based on *Bayesian Principal Component Analysis* (BPCA), which estimates a probabilistic model and latent variables within the framework of Bayesian inference. More recently, Kim *et al.*[3] successfully applied *Local Least Squares Imputation* (LLSimpute) method to estimate the missing values. In LLSimpute, a target gene with missing values is modeled as a linear combination of K coherent genes that were found using nearest neighbor method, where K is learned using known expression values.

In this paper, we propose a novel iterated way of using local least squares to more accurately impute the missing values — *Iterated Local Least Square Imputation* (ILLsimpute). Note that in most existing imputation methods, a constant K coherent genes are selected for a target gene. This constant is usually pre-specified, though in LLSimpute it is learned out of the microarray dataset. But they select for every target gene K coherent genes. In our ILLsimpute, we do not put a hard constraint on the numbers of coherent genes picked for target genes, i.e., they may vary. In fact, the known expression values in the microarray dataset are used to learn a *threshold* of similarity to obtain a set of coherent genes for every target gene. Subsequently, the target gene is represented by a linear combination of those coherent genes using the local least squares, and the missing values are estimated as done in LLSimpute method. The process is repeated for a number of iterations or terminates when the imputed values converge. The detailed steps of operations in ILLsimpute will be presented in Section 2. We have compared the performance of ILLsimpute with three other most recent methods, namely, BPCA and LLSimpute, and KNNimpute method on six microarray datasets that we were able to obtain using a number of different percentages of missing values. The detailed experimental results are summarized in Section 3. Finally,

we conclude the paper in Section 4 with some future works and discussions.

2. Iterated Local Least Squares Imputation — ILLsimpute

Again, let $G_{m\times n}$ denote the gene expression matrix for m genes in n experiments. Note that in general we have $m \gg n$. g_{ij} denotes the expression level of the i-th gene in the j-th experiment. The *Iterated Local Least Squares Imputation* (ILLsimpute) method is made up of two parts: In the first part, a similarity *threshold* is estimated using the known expression values in $G_{m\times n}$. In the second part, the *threshold* is used in LLSimpute method for several iterations to obtain the final estimated values for the missing entries in $G_{m\times n}$. In what follows, we introduce how LLSimpute method[3] works and then how we estimate the similarity *threshold* for finding a set of coherent genes for each target gene.

First of all, we introduce a distance measure between two genes, which is adopted throughout the paper in finding coherent genes for a target gene. A *target* gene is one that contains missing values to be estimated. To determine its coherent genes, for every other gene, the missing values are filled with the average of the known expression levels of the gene, called *row average*. Then, ignoring those entries in the gene that correspond to missing value columns in the target gene, as well as those missing value columns in the target gene, we have two complete vectors of (known or row average) expression levels, whose Euclidean distance is taken as the distance between the candidate gene and the target gene. For example, if the target gene is $(U, 1.5, U, 2.0, -0.5, U, 3.3)$, where U stands for unknown, then for gene $(1.5, 1.4, U, U, -0.5, -3.9, 3.5)$ in which the unknowns are estimated to be $\frac{1}{5}(1.5 + 1.4 - 0.5 - 3.9 + 3.5) = 0.4$, the distance to the target gene is the Euclidean distance between $(1.5, 2.0, -0.5, 3.3)$ and $(1.4, 0.4, -0.5, 3.5)$, which is $\sqrt{2.61} \approx 1.61$.

2.1. Local Least Squares Imputation — LLSimpute

Using LLSimpute method to estimate missing values in a target gene, one first chooses K nearest neighboring genes using the distance measure defined in the above (K to be determined, Section 2.3). These genes are regarded as coherent genes to the target gene. The missing values in these coherent genes are filled with their respective row averages.

To explain how local least squares imputation works, we assume without loss of generality that gene 1 is the target gene and it has missing values at the first $n-n'$ positions. That is, $g_{11}, g_{12}, \ldots, g_{1,n-n'}$ are unknown. Suppose the K nearest neighboring genes of gene 1 are genes $s_1, s_2, \ldots, s_K$. Denote the submatrix of $G_{m\times n}$ containing rows $1, s_1, s_2, \ldots, s_K$ as $G'_{(K+1)\times n}$. We rewrite $G'_{(K+1)\times n}$ as:

$$\begin{pmatrix} g_1 \\ g_{s_1} \\ \vdots \\ g_{s_K} \end{pmatrix} = \begin{pmatrix} g_{1\times(n-n')} & w^T \\ B_{K\times(n-n')} & A_{K\times n'} \end{pmatrix} = \begin{pmatrix} g_{11} & \ldots & g_{1,n-n'} & w_1 & \ldots & w_{n'} \\ B_{11} & \ldots & B_{1,n-n'} & A_{1,1} & \ldots & A_{1,n'} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ B_{K1} & \ldots & B_{K,n-n'} & A_{K,1} & \ldots & A_{K,n'} \end{pmatrix}.$$

We then proceed to compute a K-dimensional coefficient vector x such that the square

$|A^T x - w|^2 = (A^T x - w)^T (A^T x - w)$ is minimized, that is,

$$\min_x |A^T x - w|^2.$$

Let x^* denote the vector such that the square is minimized, that is,

$$w \simeq A^T x^* = x_1^* a_1 + x_2^* a_2 + \ldots + x_K^* a_K.$$

Therefore, we may take

$$g^*_{1\times(n-n')} = B^T x^* = x_1^* B_1 + x_2^* B_2 + \ldots + x_K^* B_K$$

as an estimation for the missing values $g_{1\times(n-n')}$.

2.2. Measures of Performance

The performance of an imputation method, or the *imputation accuracy*, is generally measured by the *normalized root mean squared error* (NRMSE). Let S be the set of expression matrix entries containing missing values. Since these missing values were simulated (see Section 3.1 for more details), every entry $x_i \in S$ has its true expression value a_i. The imputed value for this entry is a_i^*. The difference $|a_i^* - a_i|$ is the imputation error associated with this entry x_i. Let μ denote the mean of all squares of errors, i.e. $\mu = \frac{1}{|S|}\sum_{x_i \in S}(a_i^* - a_i)^2$, and σ denote the standard deviation of all the true expression values for these entries, i.e. $\sigma = \sqrt{\frac{1}{|S|}\sum_{x_i \in S}(a_i - \bar{a})^2}$, where $\bar{a} = \frac{1}{|S|}\sum_{x_i \in S} a_i$ is the mean of these true expression values. Then, NRMSE is defined to be

$$\text{NRMSE} = \frac{\sqrt{\mu}}{\sigma}.$$

Clearly, the lower NRMSE, the better performance the method has.

2.3. Nearest Neighboring Gene Determination

In LLSimpute method, there is a stage to determine the value for parameter K, before actually doing the imputation. For every target gene, it first replaces every missing value in the other genes by its row average, as is done in the distance calculation. Then, for every gene, a certain number of known expression levels are erased to create the so-called *known missing values*. For every value of K ranging from 1 to the total number of genes in the dataset, it runs LLSimpute once to estimate these *known missing values* and calculates the imputation accuracy, measured by NRMSE. The value that achieves the best imputation accuracy is chosen for K. It's worth mentioning that when there are at least 400 complete genes, only the complete genes are considered as candidate neighboring genes.

Note that once K is determined, LLSimpute method finds exactly K coherent genes for every target gene, regardless however quality difference is for different target genes. We have observed that some target genes seem to have closer coherent genes while for the others the coherent genes are not necessarily similar. Therefore, it is more reasonable to assume that different target genes have different numbers of coherent genes. We have decided not to constrain the number of coherent genes, but to set up a distance *threshold* to cut off dissimilar genes. That is, only those genes within distance *threshold* to the target

gene are selected as coherent genes. We set *threshold* to $mean \times ratio$, where $mean$ is the average distance of all other genes to the target gene and $ratio$ is a constant to be determined.

We have tried several ways to determine $ratio$. In one way, we followed the determination of K in LLSimpute method. That is, we filled in the original missing values by their respective row averages, and then erased a certain number of known expression levels to create the *known missing values*. We then tried different values for $ratio$, ranging from 0.5 to 1.5 with an increment of 0.1, and picked the one that achieves the best imputation accuracy through running LLSimpute method once to be the value for $ratio$. In another way, we tried a greedy fashion in which $ratio$ was set initially to 1.0, and then it was increased or decreased by 0.1 depending on the direction leading to better imputation accuracy. The process was terminated when there was no better imputation accuracy could be achieved and the final $ratio$ could be considered as a local optimum. Note that in the second way, we allowed $ratio$ to go beyond 0.5 and 1.5.

In the third and the fourth ways of $ratio$ determination, we first calculated the percentage of missing values in the dataset, i.e. the *missing rate*, then removed genes containing missing values from the dataset to obtain a complete dataset, and created a same percentage of *known missing values*. This new dataset, though smaller, was used to determine $ratio$, again by two approaches as in the last paragraph.

Among these four ways of $ratio$ determination, we found out that the first way leads to the best performance. The reported results in Section 3 were obtained using $ratio$ determined in the first way.

2.4. Iterated Local Least Squares Imputation — ILLsimpute

Using the determined $ratio$ (or equivalently, similarity *threshold*), in the first iteration, our method ILLsimpute selects the coherent genes for every target gene and then runs LLSimpute to estimate the missing values. Afterwards, at each iteration, ILLsimpute uses the imputed results from last iteration to *re-select* the coherent genes for every target gene, using the same $ratio$. Note that the difference in the first iteration is that row averages were used to select the coherent genes. ILLsimpute then applies again LLSimpute method once to *re-estimate* each of the missing values. ILLsimpute terminates after a pre-specified number of iterations or when the re-imputed values in the current iteration have no differences to the imputed values in the preceding iteration, that is, the imputed values converge. In our implementation, we found out that convergence usually took hundreds of iterations and according to literature discussions we have decided to set the number of iterations to 5.

3. Experimental Results

We compared the performance of our method ILLsimpute to three other most recent methods, namely, BPCA, LLSimpute, and KNNimpute method.

3.1. Datasets

We have obtained six microarray datasets for our comparison purpose. The first four datasets are from Spellman *et al.*,[5] which were used for identification of cell-cycle regulated

genes in yeast *Saccharomyces cerevisiae*. These datasets were obtained from the file 'CDCDATA.txt' following link `http://genome-www.stanford.edu/cellcycle/data/rawdata/`. There are three parts in the file: Alpha-part, Cdc part and Elu part. There are 6178 genes in the original file. The first dataset *alpha-dataset* and the second *elu-dataset* are the Alpha-part and the Elu-part in the file, respectively, obtained by removing genes with missing values in any part. Both datasets contain 4304 genes, with alpha-dataset in 18 experiments and elu-dataset in 14 experiments, respectively.

Again from the original file, consider only those C-genes (i.e., YAC, YBC, ..., YPC genes) in the last 14 columns. Removing genes with missing values gives us *cyc-a-dataset* that contains 2865 genes in 14 experiments. Another way is to remove genes as long as they contain a missing value in any column in the original file. This gives a much smaller dataset *cyc-b-dataset* that contains 242 genes in 14 experiments.

The fifth dataset was from a study of response to environmental changes in yeast[2] and can be retrieved through link `http://www-genome.stanford.edu/Mec1/data/DNAcomplete_dataset/`. It contains 6167 genes in 52 experiments. We first removed experiments/columns that have more than 2% missing values, and then removed genes still containing missing values, to obtain *env-dataset* that contains 5431 genes in 13 experiments.

The sixth dataset is the cDNA microarray data relevant to human colorectal cancer (CRC) studied in Takemasa *et al.*,[6] called *ta.crc-dataset* and containing 758 genes in 50 samples.

We note that alpha-dataset, elu-dataset, and ta.crc-dataset have been used in the studies of BPCA[4] and LLsimpute.[3]

3.2. Threshold Determination in ILLsimpute

As explained in Section 2.3, we have tried 4 different ways to determine the best value for $ratio$ and we have decided to go with the first way. Figure 1 plots the NRMSE values achieved by ILLsimpute using different $ratio$ values on elu-dataset and cyc-b-dataset both with 10% missing rate. We have tested all $ratio$ values from 0.5 to 1.5 with an increment of 0.1. We remark that the best value for $ratio$ is dataset dependent, for example, it is 0.6 and 0.9 for elu-dataset and cyc-b-dataset, respectively. (The greedy way of $ratio$ determination gave both 1.0 for elu-dataset and cyc-b-dataset, and the subsequent NRMSE was 0.246 and 0.283, respectively.)

3.3. Number of Iterations Determination in ILLsimpute

Though we expect that the imputed missing values converge, we found out that ILLsimpute method took a large number (in hundreds) of iterations when the maximum number of iterations was not specified. We have also observed that sometimes even at convergence point ILLsimpute method didn't necessarily achieve the best NRMSE values. Figure 2 plots the NRMSE values achieved by ILLsimpute method using different maximum numbers of iterations, ranging from 1 to 10, on elu-dataset with 10% missing rate and $ratio$ set to 0.6 and cyc-b-dataset with 10% missing rate and ratio set to 0.9 — the best values for $ratio$. On both cases, the best NRMSE values were achieved in 5 iterations. Therefore, according

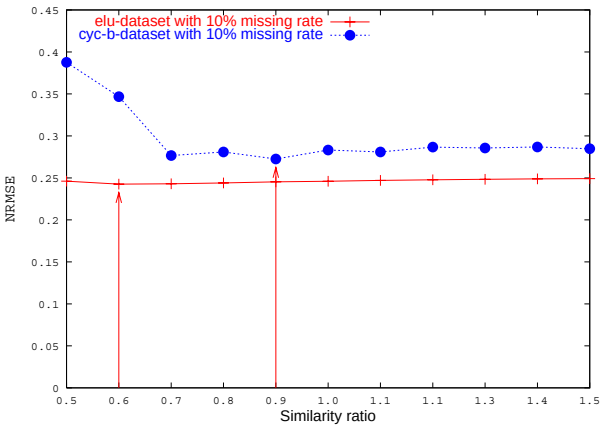


Figure 1. NRMSE values for ILLsimpute on elu-dataset and cyc-b-dataset both with 10% missing rate, w.r.t. different $ratio$ values.

to these results and some discussions in the literature we have chosen 5 iterations to be used in the other experiments.

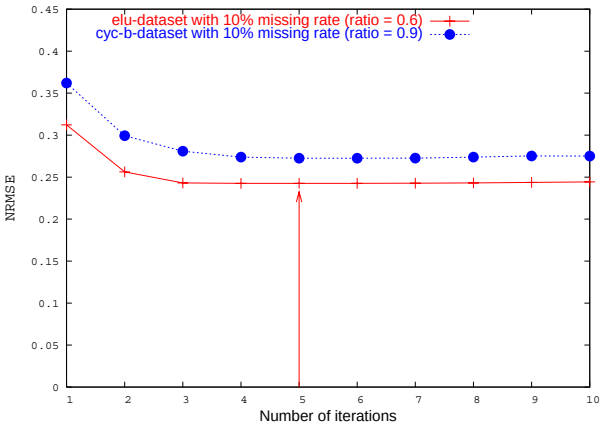


Figure 2. NRMSE values for ILLsimpute on elu-dataset and cyc-b-dataset with 10% missing rate, after different numbers of iterations.

3.4. Imputation Accuracy Comparison

The six datasets we have at hand were obtained through throwing away genes containing missing values from the original files. That is, they no longer contain any missing values. In the experiments, we randomly removed some percentage, i.e. missing rate, of expression levels to create missing values, then applied four imputation methods, KNNimpute, ILLsimpute, BPCA, and LLSimpute, to estimate them. The performances of the methods were measured by NRMSE values. Recall that for each dataset we needed to estimate the value for $ratio$ in ILLsimpute (estimate the value for K in LLSimpute as well), but the number of iterations was fixed at 5. Table 1 summarizes the best values for $ratio$ (and the average number of coherent genes for all target genes in all 5 iterations) in ILLsimpute and the best values for K in LLSimpute on cyc-b-dataset with different missing rates. We have tried several different values for K in KNNimpute, and found out that $K = 10$ gave the best accuracy. The reported results for KNNimpute method were obtained using $K = 10$. It is interesting to note that in ILLsimpute method the average numbers of coherent genes for all target genes in each iteration only differ at most 1 (results not reported), though for individual targets their numbers of coherent genes vary a lot. Also is interesting is that these average numbers differ quite a lot from those K values in LLSimpute method, which could contribute to the improved imputation accuracies of ILLsimpute.

Table 1. The best values for $ratio$ in ILLsimpute and the resultant average numbers $\overline{K}$ of coherent genes for all target genes, the best values for K in LLSimpute, and NRMSE values for all four methods, on cyc-b-dataset with different missing rates. Accuracies in bold are the best ones among all four (cf. Figure 3(d)).

	1%	2%	3%	4%	5%	10%	15%	20%
$ratio$	0.9	0.9	1.2	0.9	1.3	0.8	1.4	1.3
$\overline{K}$	139	139	181	139	190	119	197	190
ILLsimpute NRMSE	0.118	**0.098**	**0.123**	**0.135**	**0.139**	**0.281**	**0.318**	**0.356**
BPCA NRMSE	**0.086**	0.111	0.225	0.339	0.368	0.409	0.398	0.400
KNNimpute NRMSE	0.552	0.534	0.516	0.511	0.519	0.531	0.539	0.541
LLSimpute NRMSE	0.165	0.184	0.225	0.238	0.247	0.353	0.373	0.393
K	140	140	140	140	210	210	210	210

From the plots of NRMSE values (Figure 3) achieved by all four methods on six datasets, we can see that KNNimpute method always performs the worst. For all the other three methods, they perform equally well on env-dataset and ta.crc-dataset. In fact, from Figures 3(e) and 3(f), it is hard to tell which one of them performs better than the other two except KNNimpute. All three methods again perform equally well on alpha-, elu-, and cyc-a-datasets when the missing rate is small, i.e. less than 5% (cf. Figures 3(a), 3(b), and 3(c)). However, their performances differ when the missing rate is large. Typically, our method ILLsimpute performs very close to BPCA, though still a little better, and both of them outperform LLSimpute. On cyc-b-dataset, except 1% missing rate, our method

IILsimpute outperforms BPCA, LLSimpute, and KNNimpute, and the difference is larger at the practical cases where the missing rate is 5–10%.

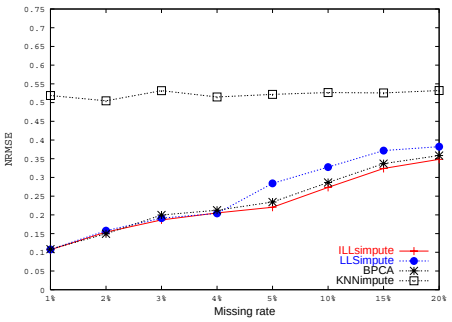
From all these results, we might be able to claim that our method ILLsimpute performs better than both BPCA and LLSimpute, the two most recent imputation methods, or at least as well as they perform.

4. Conclusions

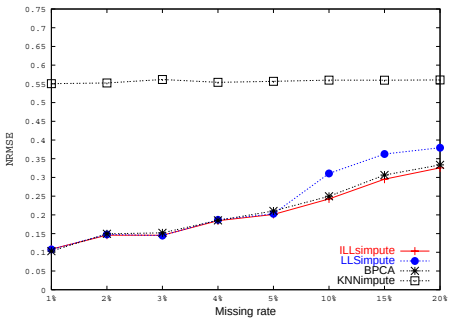
We have proposed a novel iterated version of Local Least Squares Imputation (ILLsimpute) method to estimate the missing values in microarray data. In ILLsimpute, the number of nearest neighbors for every target gene is automatically determined, rather than pre-specified in most existing imputation methods. The experimental results on six real microarray datasets show that ILLsimpute outperforms three most recent imputation methods BPCA, LLSimpute, and KNNimpute, or at least equally well, on all datasets with simulated missing values.

References

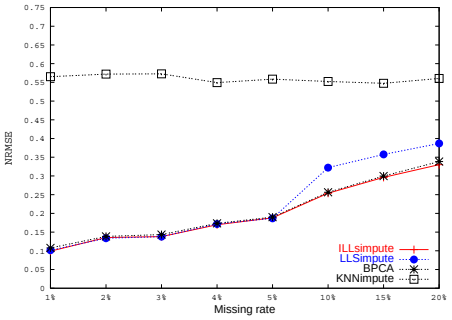
1. A. J. Butte, J. Ye, G. Niederfellner, K. Rett, H. Hring, M. F. White, and K. P. White. Determining significant fold differences in gene expression analysis. *Pacific Symposium on Biocomputing*, 6:6–17, 2001.
2. A. P. Gasch, M. Huang, S. Metzner, D. Botstein, S. J. Elledge, and P. O. Brown. Genomic expression responses to DNA-damaging agents and the regulatory role of the yeast ATR homolog Meclp. *Molecular Biology of the Cell*, 12:2987–3003, 2001.
3. H. Kim, G. H. Golub, and H. Park. Missing value estimation for DNA microarray gene expression data: Local least squares imputation. *Bioinformatics*, 20:1–12, 2004.
4. S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubara, and S. Ishii. A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics*, 19:2088–2096, 2003.
5. 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. *Molecular Biology of the Cell*, 9:3273–3297, 1998.
6. I. Takemasa, H. Higuchi, H. Yamamoto, M. Sekimoto, N. Tomita, S. Nakamori, R. Matoba, M. Monden, and K. Matsubara. Construction of preferential cDNA microarray specialized for human colorectal carcinoma: molecular sketch of colorectal cancer. *Biochemical and Biophysical Research Communications*, 285:1244–1249, 2001.
7. O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17:520–525, 2001.



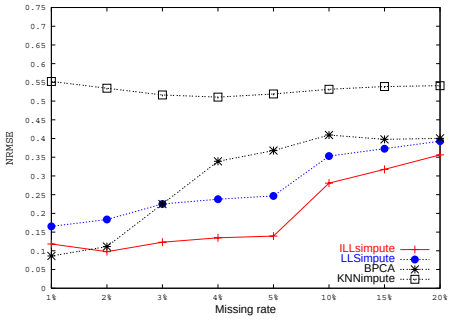
(a) alpha-dataset.



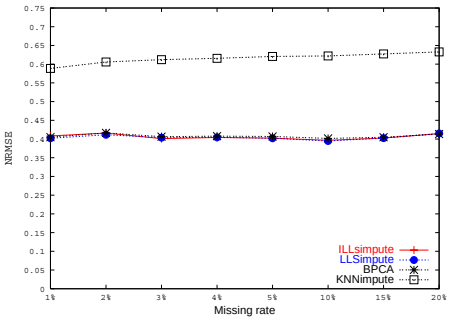
(b) elu-dataset.



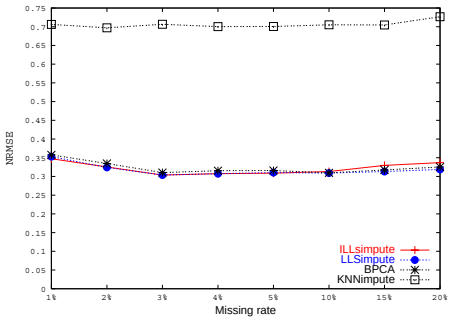
(c) cyc-a-dataset.



(d) cyc-b-dataset.



(e) env-dataset.



(f) ta.crc-dataset.

Figure 3. NRMSE comparisons for ILLsimpute, BPCA, LLSimpute, and KNNimpute on six datasets with various percent of missing values.