Microarray Missing Value Imputation: A Regularized Local Learning Method

Aiguo Wang, Ye Chen, Ning An, Jing Yang, Lian Li, and Lili Jiang

Abstract—Microarray experiments on gene expression inevitably generate missing values, which impedes further downstream biological analysis. Therefore, it is key to estimate the missing values accurately. Most of the existing imputation methods tend to suffer from the over-fitting problem. In this study, we propose two regularized local learning methods for microarray missing value imputation. Motivated by the grouping effect of L_2 regularization, after selecting the target gene, we train an L_2 Regularized Local Least Squares imputation model (RLLSimpute_L2) on the target gene and its neighbors to estimate the missing values of the target gene. Furthermore, RLLSimpute_L2 imputes the missing values in an ascending order based on the associated missing rate with each target gene. This contributes to fully utilizing the previously estimated values. Besides L_2, we further explore L_1 regularization and propose an L_1 Regularized Local Least Squares imputation model (RLLSimpute_L1). To evaluate their effectiveness, we conducted extensive experimental studies on six benchmark datasets covering both time series and non-time series cases. Nine state-of-the-art imputation methods are compared with RLLSimpute_L2 and RLLSimpute_L1 in terms of three performance metrics. The comparative experimental results indicate that RLLSimpute_L2 outperforms its competitors by achieving smaller imputation errors and better structure preservation of differentially expressed genes.

Index Terms—Microarray data, missing value imputation, regularized model, local learning, similarity measurement

1 Introduction

The rapid development of DNA microarray technology impressively facilitates the simultaneous measurement of expression profiles of thousands of genes under different experimental conditions [1], [2], and the obtained microarray data provide an alternative to the identification of disease genes and the classification of tumor subtypes at the molecular level [3], [4], [5]. Accordingly, researchers have used various machine learning models and statistical analysis techniques to analyze microarray data for the discovery of meaningful biological knowledge [6], [7]. However, the complexity of microarray technology makes it inevitable to generate missing values with varying degrees in gene expression profiles [8]. Many human and non-human factors in microarray technology, such as the irregular use of microarray chips, insufficient resolution and contamination of microarray surface, would lead to gene expression profiles with missing values [9]. There are studies showing that most of publicly available microarray datasets contain missing values to varying degrees as high as 50 precent or even 95 precent [10]. On the other hand, most of existing gene selection, clustering and functional annotation algorithms have no internal way for missing values and require complete data as inputs ideally [11]. Directly removing these instances or genes with missing entries definitely causes loss of information, especially when the removed genes play a dominant role in a biological process [12]. Therefore, it is crucial to precisely estimate the missing values of microarray data [13].

Obviously, repeating microarray experiments is a simple and effective method to deal with this situation. However, due to the high experimental costs and the uncertainty of obtaining the expected complete data, this multiple repetition-based method is usually not a priority [14]. To obtain a complete dataset, we can simply replace missing values with zeros, average of the observable values in the same gene (row mean) or sample (column mean) [15]. Though these methods have the advantages of efficiency and easy implementation, they fail to utilize the latent data structure information such as gene co-expression and relevance, thus generally achieve a lower imputation accuracy.

During the past few years, researchers have proposed a wealth of effective imputation methods for microarray data, which can be broadly categorized into four groups: biology knowledge-based, global learning-based, local learning-based, and hybrid-based methods [16], [17], [18]. Biology knowledge-based methods utilize validated biology knowledge as prior information for the target gene imputation [19]. A major limitation of these methods is that they heavily rely on domain specific knowledge and fail to tackle the new under-explored cases that are with less biological knowledge available. Global learning-based methods assume that a covariance structure exists in the dataset and they utilize such information to estimate the missing values. Singular vector decomposition imputation (SVDimpute) and Bayesian

- *A. Wang, Y. Chen, N. An, J. Yang, and L. Li are with the School of Computer and Information, Hefei University of Technology, Hefei, Anhui 230000, China. E-mail: wangaiguo2546@163com, ye1991214@126.com, ning.g.an@acm.org, jsjyj0801@163.com, llian@hfut.edu.cn.*
- *L. Jiang is with the Department of Computing Science, Umeå University, Umeå, Västerbotten 90187, Sweden. E-mail: lili.jiang@cs.umu.se.*

Manuscript received 8 Feb. 2017; revised 7 Jan. 2018; accepted 24 Feb. 2018. Date of publication 27 Feb. 2018; date of current version 31 May 2019. (Corresponding author: Ning An).
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCBB.2018.2810205

principal component analysis (BPCAimpute) methods are two representatives [16], [20]. Generally, global learning-based methods are suitable for microarray datasets with large sizes, but they fail to capture the local similar structures in the data. In contrast to global learning-based methods, local learning-based methods seek to find the similar local structure and then impute the missing values with those genes that are similar to the target gene. For example, the *k*-nearest-neighbor imputation (KNNimpute) is among the first proposed method to estimate the missing values of a target gene with its k nearest neighbors [17]. Sequential *k*-nearest-neighbor imputation (SKNNimpute) and iterative *k*-nearest-neighbor imputation (IKNNimpute) are two improved versions of KNNimpute and they also estimate the missing values by weighting the values of the selected neighbors [21], [22]. As nearest-neighbor-based methods fail to consider the relevance between the neighbors, there are studies that propose to use linear regression models to build the relationships between the target gene and its neighbors and then impute the missing values with its neighbors and associated regression coefficients. In comparison with nearest-neighbor based methods, these methods, including least squares imputation (LSimpute) and local least squares imputation (LLSimpute), tend to obtain better imputation accuracy [23], [24]. Hybrid-based methods aim to combine multiple imputation methods for better performance [5]. Commonly used schemes include integrating different imputation methods under an ensemble or semi-supervised learning framework [25], [26], linearly or non-linearly combining multiple imputation methods [15], [27].

Regarding missing value imputation, as most microarray datasets contain genes with similar expression profiles and have strong local structures, local learning-based methods generally outperform global learning-based methods. Nevertheless, most of them tend to suffer from the over-fitting problem. That is, a model fits the training set well but behaves poorly on the independent unseen test set. To mitigate this problem, in this study, we propose to penalize the linear regression model with an L_2 regularization for missing value imputation of microarray data. Significantly, besides L_2 regularization, we can incorporate other regularization terms, such as L_1 regularization and $L_{1/2}$ regularization. In comparison with L_2 regularization that has the grouping effect, L_1 regularization is a sparse model that encourages the sum of the absolute values of the parameters to be small, and $L_{1/2}$ regularization ignores the correlation between variables. If there is a group of highly relevant variables, L_1 regularization and $L_{1/2}$ regularization simply pick one variable to represent the group. This can reduce overfitting but at the cost of a loss in predictive power. In contrast, L_2 regularization will keep all the highly relevant variables towards a better predictive performance. Also, our preliminary experimental results indicate that L_2 regularized regression models achieve better imputation performance, so we choose L_2 regularization in the study. Specifically, we train an L_2 Regularized Local Least Squares model (RLLSimpute_L2) between the target gene and its neighbors for estimating the missing values of the target gene. Because L_2 regularization exhibits the grouping effect, RLLSimpute_L2 can make highly correlated variables in or out a model together. This enables us to handle the overfitting problem as well as keep these highly predictive variables in the regression model. Furthermore, RLLSimpute_L2 sequentially imputes the missing values in ascending order of the missing rates associated with the target genes for utilizing previously estimated values. Therefore, the proposed method is expected to achieve a satisfactory performance for microarray data imputation. Besides, we further explore the use of L_1 regularization in mitigating the problem of over-fitting. We then propose an L_1 Regularized Local Least Squares imputation model (i.e., RLLSimpute_L1) and compare it with RLLSimpute_L2 in microarray missing value imputation.

The remainder of this paper is organized as follows. Section 2 reviews related work on microarray data imputation by giving three representative methods. Section 3 illustrates the proposed regularized local learning-based imputation methods, including RLLSimpute_L2 and RLLSimpute_L1. In section 4, we introduce three metrics for performance evaluation. Section 5 presents the experimental setup and experimental results, shows how the proposed methods avoid the risk of over-fitting, and analyzes the time complexity of the proposed methods. Section 6 concludes this study briefly and discusses future work.

2 Related Work

There is a wealth of imputation methods for researchers to use for estimating the missing values. And they can be broadly grouped into four categories: biology knowledge-based, global learning-based, local learning-based, and hybrid-based methods. In practice, global learning-based methods and local learning methods are commonly used and are the basic building block of many hybrid-based methods [28], [26]. To have a general idea of global learning-based methods and local learning-based methods and to better understand their relationships with our proposed methods RLLSimpute_L2 and RLLSimpute_L1, in this section, we detail one representative global learning-based method (i.e., BPCAimpute) and two local learning-based methods (i.e., KNNimpute and LSimpute).

2.1 BPCAimpute

Bayesian principal component analysis imputation (BPCAimpute) method belongs to the global learning-based methods and is performed by three steps (as follows) in the processes of missing value estimation: 1) Principal Component (PC) regression, 2) Bayesian estimation, and 3) an Expectation-Maximization (EM)-like repetitive algorithm [20]. In detail, the following example is taken to explain how BPCAimpute works. BPCAimpute regards a d-dimensional gene expression vector y as a linear combination of principal axis vectors $\boldsymbol{w}_l\,(1 \ll l \ll k, \text{and}\, k < d)$,

$$y = \sum_{l=1}^{k} \mathbf{x}_l \mathbf{w}_l + \varepsilon, \tag{1}$$

where k is the number of principle components, $\boldsymbol{x}_l$ is called the factor score, ε denotes the residual error. The l-th principal axis vector $\mathbf{w}_l = \sqrt{\lambda_l} u_l$, where λ_l and u_l denote the $l-\text{th}$ eigenvalue and the corresponding eigenvector of the covariance matrix S for the data set Y. Y is a collection of gene expression vectors. The principal axis vectors $\mathrm{W} = (\mathrm{W}^{\mathrm{obs}}, \mathrm{W}^{\mathrm{miss}})$, where $\mathrm{W}^{\mathrm{obs}}$ denotes the observed part in the data

and W^{miss} denotes the missing part. If the value of k is given, the factor scores $\mathbf{x} = (x_1, x_2, ...x_k)$ for the expression vector $\mathbf{y}$ can be obtained by minimizing the residual error (2) over the observed data set $\mathbf{y}^{obs}$,

$$err = \left\| \mathbf{y}^{obs} - W^{obs}\mathbf{x} \right\|^2. \tag{2}$$

Then, the missing values in the gene vector $\mathbf{y}$ can be estimated using (3),

$$\mathbf{y}^{miss} = W^{miss}\mathbf{x}, \tag{3}$$

The parameter W is unknown beforehand in the above procedure. BPCA use a probabilistic model, which is called probabilistic principle component analysis (PPCA). Meanwhile, the model is built based on the assumption that the residual error ε and the factor scores $\mathbf{x}$ obey normal distributions, as shown in (4), (5),

$$p(\mathbf{x}) = N_k(\mathbf{x}|0, I_k), \tag{4}$$

$$p(\varepsilon) = N_d(\varepsilon|0, (1/\tau)I_d), \tag{5}$$

where I_k is a $(k \times k)$ identity matrix and τ is a scalar inverse variance of ε. $N_k(\boldsymbol{x}|u, \sigma)$ denotes a k-dimensional normal distribution for $\boldsymbol{x}$, whose mean and covariance are u and σ, respectively. BPCAimpute assumes $Y = \{Y^{obs}, Y^{miss}\}$, where Y^{obs} and Y^{miss} denote observed part and the missing part, respectively. The variational Bayes algorithm is used to estimate the posterior distribution of the parameter $\theta = \{W, u, \tau\}$ and Y^{miss} simultaneously. The default value of k is $(d-1)$.

2.2 Methods Based on KNNimpute

In KNNimpute method, k nearest-neighbor genes are taken from the whole matrix of the test data set except genes that has missing values at the same position with the gene to be imputed [17]. Euclidean distance is usually used as the metric to estimate the similarity of neighboring genes. To compare the similarity of this metric, each gene should have the same dimension and missing positions of values inside. Missing values are estimated with the weighted average of the corresponding column of the k nearest genes.

The weight w_i of ith gene is calculated using

$$w_i = \frac{1/d_i}{\sum_{i=1}^{k} 1/d_i}, \tag{6}$$

where k is the number of selected neighbor genes and d_i is the distance between ith gene and the target gene.

Slightly different from KNNimpute, for sequential KNNimpute (SKNNimpute) method, the target gene after imputed can be reused in another round of imputation for other target genes. And the strategy to select similar genes is more rigorous than KNNimpute, that is, only complete genes have chances to be selected as similar genes for the target gene. In addition, the missing values were estimated sequentially, starting from the gene having the smallest missing rate to the gene with the largest missing rate [21].

Iterative KNNimpute (IKNNimpute), an enhanced version of KNNimpute [22], first replaces all missing values in microarray data by row (gene) average, obtains a complete matrix X^0 of microarray data. Then IKNNimpute selects k nearest similar genes in X for the target gene to estimate the missing values. Repeat this step until there is no gene with missing values. Finally, we can get an updated version of X, denoted as X^1. The sum of squared differences between the estimated positions of the X and X^1 can be measured using (7),

$$G = [\mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_n] = \begin{bmatrix} \mathbf{s}_1^T \\ \mathbf{s}_2^T \\ \vdots \\ \mathbf{s}_m^T \end{bmatrix} = \begin{bmatrix} g_{1,1} & \alpha_{1,2} & \alpha_{1,3} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & g_{2,3} & \cdots & g_{2,n} \\ g_{3,1} & g_{3,2} & g_{3,3} & \cdots & \alpha_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{m,1} & g_{m,2} & \alpha_{m,3} & \cdots & g_{m,n} \end{bmatrix}$$

Fig. 1. Logical storage structure of a microarray dataset with rows denoting samples and column representing genes. $\alpha_{i,j}$ indicates that there is a missing value at the $i-th$ row and $j-th$ column.

$$\delta^i = \left\| X^i - X^{i+1} \right\|. \tag{7}$$

If $\delta < \tau$ (τ is a threshold), the imputation is over. Otherwise, start a new round imputation for X^1 to obtain the X^2 until the convergence criterion is reached.

2.3 Methods Based on LLSimpute

In Local least squares imputation (LLSimpute), the basic idea for choosing similar genes is the same as that of KNNimpute [24], and the difference is that LLSimpute involves a multiple linear regression model to replace the weighted average algorithm as equation (6). The model is presented as below,

$$\min_{\mathbf{x}} \left\| A^T\mathbf{x} - \mathbf{s} \right\|_2, \tag{8}$$

where A is the matrix composed of similar genes that deletes the samples with missing values at the same indices as the target gene, $\mathbf{s}$ denotes the target gene after deleting the position of missing values, and $\mathbf{x}$ is the regression coefficient vector. After minimizing (8) and obtaining $\mathbf{x}$, we can estimate the missing values of the target gene using

$$\alpha = \mathbf{b}^T\mathbf{x} = \mathbf{b}^T\left(A^T\right)^+\mathbf{w}, \tag{9}$$

where $\mathbf{b}$ is the matrix composed of samples having the same positions as the target gene, and α denotes the estimated values for the target gene.

Similar to the case of KNNimpute, the differences between SLLSimpute and LLSimpute are corresponding to the ones between KNNimpute and SKNNimpute [29].

3 The Proposed Method

In this section, we present the proposed imputation method for gene expression profiles. In the analysis of microarray data, we usually use a matrix to represent microarray data, as shown in Fig. 1, where each row denotes a sample and each column corresponds to a gene. Specifically, we use $G \in R^{m \times n}$ to represent a microarray dataset that contains m samples and n genes ($m << n$). We use $s_1, s_2, ..., s_m$ to denote the m samples, and $\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_m (\mathbf{s}_i \in R^{n \times 1}, 1 \le i \le m)$ are corresponding vectors. We use $g_1, g_2, ..., g_n$ to represent the n genes and use $\mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_n$ ($\mathbf{g}_i \in R^{1 \times m}$, $1 \le i \le n$) to indicate their vector forms. The entry $g_{i,j}$ represents the expression value of the jth gene in the $i-th$ sample. Particularly, we use $\alpha_{i,j}$ to

indicate that there is a missing value at the ith row and jth column, such as $\alpha_{1,2}$ and $\alpha_{m,3}$.

According to the above discussions, we first present the proposed imputation method RLLSimpute_L2. Specifically, RLLSimpute_L2 estimates the missing values with the following three steps until there exists no gene with missing values. In the next sections, we illustrate its three key steps: identifying a target gene with missing values, selecting neighbors of the target gene, and training a regression model between the target gene and its neighbors for imputation.

3.1 The Selection of Target Gene

Because the previously estimated values can be used to impute the missing values of other target genes, RLLSimpute_L2 adopts such a strategy: sequentially imputing the missing values in ascending order of the missing rate associated with each target gene. That is, RLLSimpute_L2 selects the gene with minimal missing rate as target gene and estimates its missing values. Then we can use the target gene with missing values estimated when handling other genes with missing values. For a microarray dataset G, suppose that there are n_1 genes with missing values, we use $G_1 \in \mathrm{R}^{m\times n_1}$ to store the genes with missing values, and use (10) to calculate the missing rate r_i of the target gene $\mathbf{g}_i$ in G_1,

$$r_i = \frac{l_i}{m}, \tag{10}$$

where l_i represents the number of missing entries in $\mathbf{g}_\mathrm{i}$ and m is the total number of samples. Then, the gene $\mathbf{g}_\mathrm{i}$ with minimal missing rate is chosen as target gene $\mathbf{g}_\mathrm{t}$.

3.2 The Selection of Similar Gene

For local learning-based methods, the second step is to identify genes that are similar to the target gene. Obviously, we can use various distance metrics, such as Pearson correlation coefficient and Euclidean distance, to measure the similarity between two genes. The following objective function (11) is used to measure the distance d_v between the target gene $\mathbf{g}_t$ and its neighbor $\mathbf{g}_v$.

$$d_v = f(\mathbf{g}_t, \mathbf{g}_v), \tag{11}$$

In this study, RLLSimpute_L2 uses the Euclidean distance for similarity measurement as shown in (12),

$$d_v = \left\|(\mathbf{g}_t^{\mathbf{obs}} - \mathbf{g}_v^{\mathbf{obs}})\right\|_2, \tag{12}$$

where **obs** represents the indices of samples with observable values for $\mathbf{g}_t$. A smaller distance between $\mathbf{g}_v$ and $\mathbf{g}_t$ indicates that $\mathbf{g}_v$ is more similar to $\mathbf{g}_t$ than these genes with a larger distance to $\mathbf{g}_t$.

3.3 Missing Values Estimation

After calculating the distances between the target gene $\mathbf{g}_t$ and each of its neighbors, we can select k nearest neighbors for $\mathbf{g}_t$ and use those neighbors to estimate the missing values of $\mathbf{g}_t$. We can use various linear or non-linear regression techniques to model the relationship between the target gene and its neighbors, and herein we adopt the least square model because of its easy implementation and initial success. Suppose that there are k similar genes available, we can train a linear regression model with a regularization term using (13),

$$\arg\min_{\beta}\left\{\left(\mathbf{g}_t^{\mathbf{obs}} - \sum_{v=1}^{k}\beta_v\mathbf{g}_v^{\mathbf{obs}}\right)^2 + \lambda R(\beta)\right\}, \tag{13}$$

where $\beta = [\beta_1, \beta_2, \cdots, \beta_k]$, β_v is the regression coefficient of $\mathbf{g}_v$ $(1 \le v \le k)$, and $R(\beta)$ is a regularization term that is used to penalize large coefficients/weights. The parameter λ controls a tradeoff between fitting the data well and having small weights.

Furthermore, for the regularization term,

1) if $R(\beta) = 0$, then the model is a standard regression model that easily suffers from over-fitting in the presence of many irrelevant variables.
2) if $R(\beta) = \|\beta\|_1 = \sum_{i=1}^{k}|\beta_i|$, then it is a L_1 regularized regression model and causes many coefficients to equal zero, so that β is sparse.
3) if $R(\beta) = \|\beta\|_2^2 = \sum_{i=1}^{k}\beta_i^2$, then it is a L_2 regularized regression model that encourages the sum of the squares of coefficients to be small.

In contrast to L_1 regularization that selects one variable to represent the corresponding group, L_2 regularization can make highly correlated variables in or out a model together. Hence, L_2 regularized regression model generally achieves a better tradeoff between handling overfitting and keeping the highly predictive variables. Consequently, in this study, RLLSimpute_L2 trains a linear regression model by solving (14),

$$\operatorname{argmin}_{\beta}\left\{\left(\mathbf{g}_t^{\mathbf{obs}} - \sum_{v=1}^{k}\beta_v\mathbf{g}_v^{\mathbf{obs}}\right)^2 + \lambda\sum\nolimits_{i=1}^{k}\beta_i^2\right\}. \tag{14}$$

Further, the solution to (14) is given in (15),

$$\beta^* = [\mathbf{A}\mathbf{A}^\mathbf{T} - \lambda\mathbf{I}]^{+}\mathbf{A}\mathbf{g}_t^{\mathbf{obs}}, \tag{15}$$

where $\mathbf{A} = [\mathbf{g}_1^{\mathbf{obs}}, \mathbf{g}_2^{\mathbf{obs}}, \cdots, \mathbf{g}_k^{\mathbf{obs}}]$, $\mathbf{I}$ is an identity matrix, and $(\mathbf{M})^{+}$ is the pseudo-inverse of matrix $\mathbf{M}$. RLLSimpute_L2 then estimates the missing values of $\mathbf{g}_t$ using (16),

$$\mathbf{g}_t^{\mathbf{miss}} = (\beta_1^*, \beta_2^*, ..., \beta_k^*) * [\mathbf{g}_1^{\mathbf{miss}}, \mathbf{g}_2^{\mathbf{miss}}, \cdots, \mathbf{g}_k^{\mathbf{miss}}]^\mathbf{T}, \tag{16}$$

where **miss** indicates the indices of samples with missing values for $\mathbf{g}_t$.

3.4 L_1 Regularized Local Learning Method

Besides L_2 regularization, L_1 regularization is also one of the commonly used techniques for avoiding over-fitting. Hence, we propose the L_1 Regularized Local Least Squares imputation model (i.e., RLLSimpute_L1) and explore its effectiveness. RLLSimpute_L1 takes a similar step to RLLSimpute_L2 except that RLLSimpute_L1 solves the optimization problem (17) rather than (14) to obtain the regression coefficients. We can use the least angle regression (LARS) algorithm to solve the problem and to further estimate the missing values.

$$\operatorname{argmin}_{\beta}\left\{(\mathbf{g}_t^{\mathbf{obs}} - \sum_{v=1}^{k}\beta_v\mathbf{g}_v^{\mathbf{obs}})^2 + \lambda\sum\nolimits_{i=1}^{k}|\beta_i|\right\}. \tag{17}$$

4 Evaluation Metrics

To evaluate the effectiveness of RLLSimpute_L2 and RLLSimpute_L1 in missing value imputation, we use three evaluation metrics: root mean squared error [30], Pearson correlation coefficient [31], and biomarker list concordance index [32]. The former two are statistical analysis related metrics, while the last one determines the impact of an imputation method on subsequent biological analysis. These three metrics will present a relatively comprehensive comparison between RLLSimpute_L2 and RLLSimpute_L1 and its competitors.

4.1 Root Mean Squared Error

Root mean squared error (*RMSE*) is used to measure the overall deviation of estimated values from their corresponding true values [30]. We calculate *RMSE* using (18),

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{m}\sum_{j=1}^{n}\left[\hat{G}(i,j) - G_{ori}(i,j)\right]^2}, \quad (18)$$

where N equals the total number of missing entries in G,$\hat{G}$ is an estimator of G, and G_{ori} is the original dataset that is used to generate G. *RMSE* takes a value larger than or equal to 0, and a smaller *RMSE* indicates better performance of the corresponding imputation method.

4.2 Pearson Correlation Coefficient

To investigate to which extent an imputation method maintains the original dataset structure, we use Pearson correlation coefficient, as given in (19), to evaluate the performance of an imputation method [31],

$$correlation\,coefficient = \frac{\mathrm{cov}(\hat{\mathbf{s}}^{\mathbf{T}}, \mathbf{s}_{ori}^{\mathbf{T}})}{std(\hat{\mathbf{s}}^{\mathbf{T}})std(\mathbf{s}_{ori}^{\mathbf{T}})}, \quad (19)$$

where $\mathbf{s}_{ori}^{\mathbf{T}}$ is a sample of the original dataset G_{ori}, $\hat{\mathbf{s}}^{\mathbf{T}}$ is the estimated sample of $\mathbf{s}_{ori}^{\mathbf{T}}$ in $\hat{G}$, $\mathrm{cov}(\hat{\mathbf{s}}^{\mathbf{T}}, \mathbf{s}_{ori}^{\mathbf{T}})$ is the covariance between $\mathbf{s}_{ori}^{\mathbf{T}}$ and $\hat{\mathbf{s}}^{\mathbf{T}}$, and $std(\mathbf{s}_{ori}^{\mathbf{T}})$ is the standard deviation of $\mathbf{s}_{ori}^{\mathbf{T}}$. The larger the Pearson correlation coefficient is, the better the original data structure is maintained. Pearson correlation coefficient takes a value of 1 if the estimated values are linearly relevant to their original values.

4.3 Biomarker List Concordance Index

In the study of disease gene discovery and tumor subtype classification, an important task is to identify the differentially expressed genes [32]. Biomarker list concordance index (*BLCI*) is used for measuring the preservation of differentially expression genes when we select differentially expressed genes on an imputed microarray dataset [33]. Given G_{sig} that is a collection of differentially expressed genes selected from G_{ori} and H_{sig} that is a collection of differentially expressed genes selected from $\hat{G}$, $BLCI$ can be calculated using (20),

$$BLCI(G_{sig}, H_{sig}) = \frac{|G_{sig} \cap H_{sig}|}{|G_{sig}|} + \frac{|G_{sig}^c \cap H_{sig}^c|}{|G_{sig}^c|} - 1, \quad (20)$$

where $|S|$ is the size of a set S, G_{sig}^c is the complementary set of G_{sig}, and so is H_{sig}^c to H_{sig}. *BLCI* uses significance analysis of microarrays (SAM) to identify differentially expression genes with a false discovery rate of 5 precent [34]. A larger *BLCI* indicates that the corresponding imputation method can better preserve the structure of differentially expressed genes. *BLCI* takes a maximal value of 1 if G_{sig} equals H_{sig}.

TABLE 1
Experimental Dataset Description

Dataset	Original dataset (genes*samples)	Complete dataset (genes*samples)	Missing rate (%)	Ref.
GDS38	7680*16	5282*16	6.10	[36]
GDS39	7680*14	6942*14	3.21	[16]
GDS2967	6159*33	3587*33	9.24	[37]
GDS1761	9706*64	8849*64	0.15	[38]
GDS3835	27648*48	5070*48	72.25	[39]
GDS4831	24526*22	10523*22	23.75	[40]

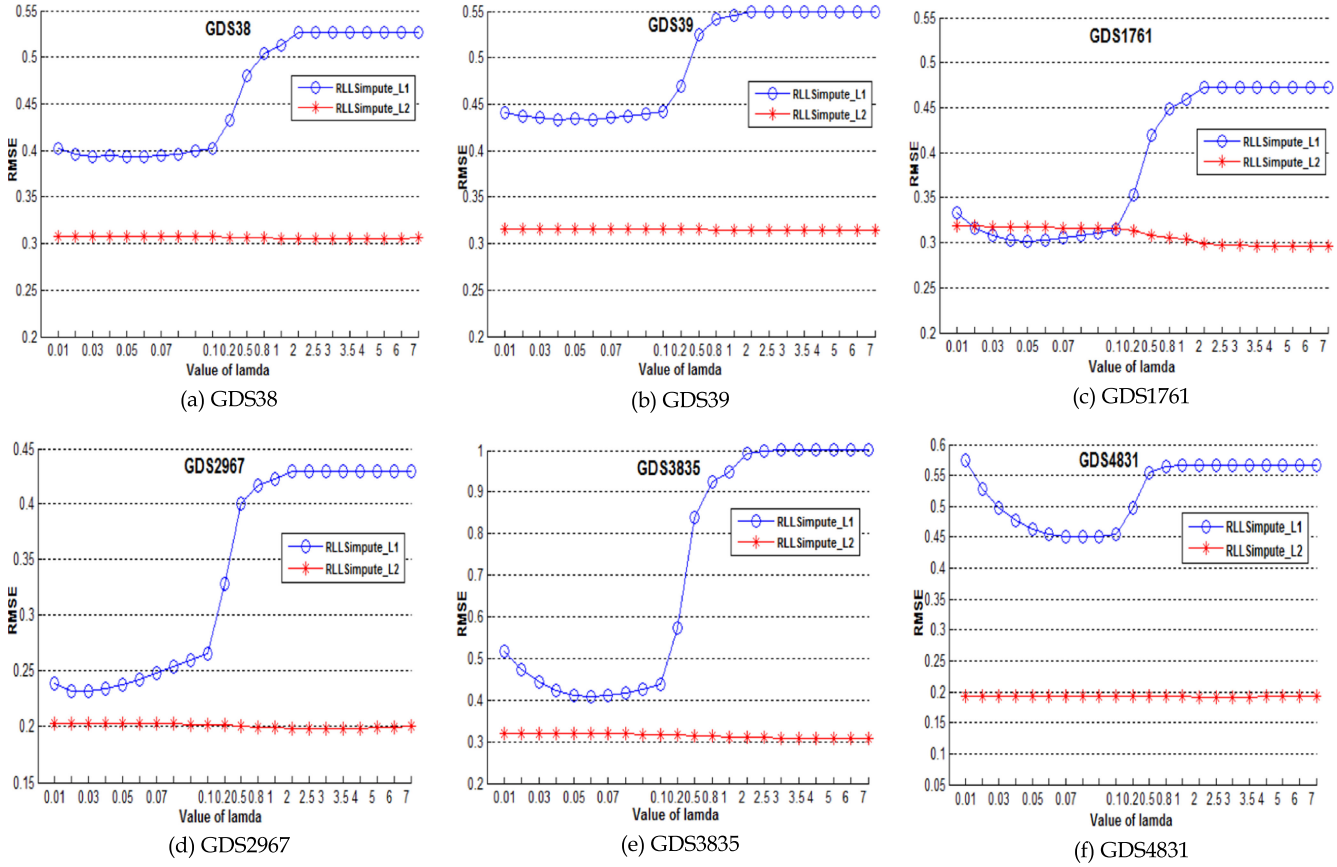
5 Experimental Results and Analysis

5.1 Experimental Data and Comparative Methods

To validate the effectiveness of the proposed method in microarray missing value imputation, we conducted extensive experiments on six publicly available microarray datasets that cover both time series and non-time series cases. Table 1 presents a brief summary of the six datasets. The first three datasets are time-series data and the rest are non-time series data. The second column shows the total number of genes of the original dataset, the third column gives the number of complete genes, and the last column shows the missing rate of each dataset. We can see that all six microarray datasets contain missing values at varying degrees. For example, GDS38 has a missing rate of 6.10 precent, and 2398 out of 7680 genes contain at least one missing entry; the missing rate of GDS1761 is about 0.15 precent; the missing rate of GDS3835 reaches up to 72.25 precent with 22578 out of 27648 genes having at least one missing entry. All experimental datasets are available online[1].

In this study, following the way of previous works [17], [23], we obtain experimental datasets by randomly introducing missing values to the complete dataset with missing rates of 1 precent, 5 precent, 10 precent, 15 precent, and 20 precent, respectively. Specifically, each original microarray dataset is pre-processed by removing samples and genes that contain missing entries to obtain a complete matrix. Then certain percentage of entries of the complete matrix are randomly selected and marked as missing values. Every random running will create an experimental dataset with the corresponding missing rate. For each experimental dataset, one imputation method is then used to estimate the introduced missing values, and the imputed values are compared to the corresponding true values for performance evaluation. To demonstrate the effectiveness of RLLSimpute_L2 and RLLSimpute_L1, we include other nine well-performing imputation methods, including BPCAimpute [20], KNNimpute [17], SKNNimpute [21], IKNNimpute [22], LLSimpute [24], SLLSimpute [29], [35], Biclustering Imputation (BSimpute) [41], Multiple Imputation by Chained Equations (MICE) [42], and ShrinkageLLS [43], as a comparison. BPCAimpute is global learning-based

1. https://www.ncbi.nlm.nih.gov/



(a) GDS38 (b) GDS39 (c) GDS1761

(d) GDS2967 (e) GDS3835 (f) GDS4831

Fig. 2. *RMSE* of RLLSimpute_L2 and RLLSimpute_L1 with different λ-values on the six datasets.

method, while RLLSimpute_L2, RLLSimpute_L1 and the rest are local learning-based methods. Furthermore, we categorize KNNimpute, SKNNimpute, and IKNNimpute as nearest neighbor-based methods and group LLSimpute, SLLSimpute, and ShrinkageLLS into least squares-based methods. BSimpute is an imputation method based on biclustering by selecting a subset of similar genes and a subset of similar samples [41]. In this study, as the authors suggested, MICE employs the classification and regression tree as the conditional model to identify the complex and nonlinear relations among variables [42]. RLLSimpue_L2 and RLLSimpute_L1 were implemented in Matlab and the code will be available on our website soon[2]. For local learning-based methods, we are required to determine the optimal number of neighbors to be used. Besides, there is an extra parameter λ for RLLSimpute_L2 and RLLSimpute_L1 to determine its optimal value. And the optimal value for the parameter is chosen as the one achieving the smallest root mean squared error. In this following section, we conduct comprehensive experiments to show the effectiveness of L_2 regularization in estimating microarray missing values.

5.2 Experimental Results and Analysis

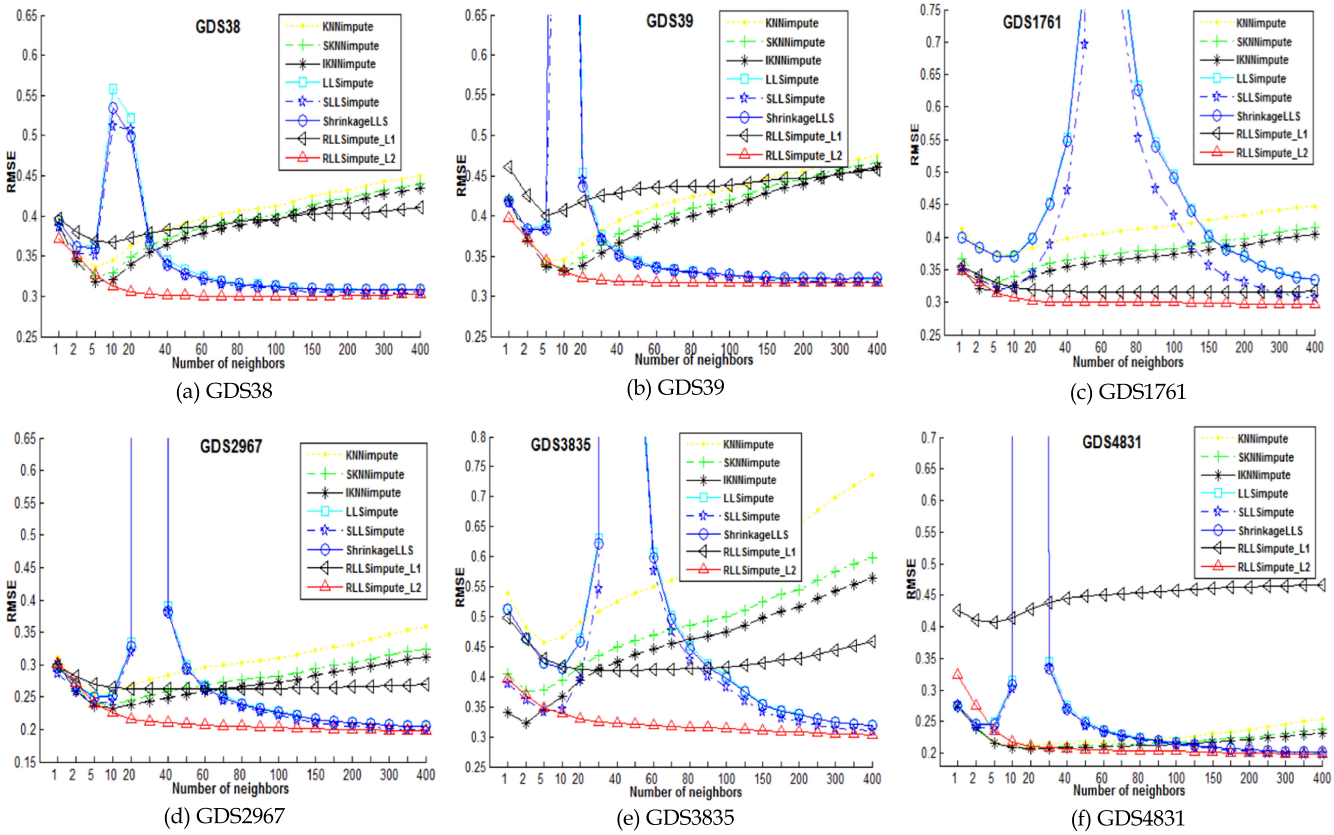
5.2.1 Determining the Value of λ

The parameter λ of RLLSimpute_L2 and RLLSimpute_L1 controls a tradeoff between fitting the data well and having small weights. To determine the approximately optimal value of λ, we conduct experiments thirty times on each dataset with a representative 5 precent missing rate. 5 precent is an empirical value from previous approaches including KNNimpute, SKNNimpute, LLSimpute, and SVDimpute. To effectively explore the value of λ, according to our preliminary work, the candidate values of λ used in this study include 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.2, 0.5, 0.8, 1, 2, 2.5, 3, 3.5, 4, 5, 6, and 7. Fig. 2 presents the averaged root mean squared errors of RLLSimpute_L2 and RLLSimpute_L1 on the six microarray datasets. The X-axis represents the optional values of λ, and the Y-axis shows the corresponding root mean squared errors (*RMSE*). From Fig. 2, we see that there is a general trend for all datasets: the *RMSE*s first decrease with increase of λ, and increase as λ increases. We then can determine the approximately optimal value of λ for each dataset. From Fig. 2, we also observe that RLLSimpute_L2 consistently obtains smaller *RMSE*s than RLLSimpute_L1, which demonstrates the superiority of L_2 regularization over L_1 regularization. Herein, L_2 regularization is a priority over L_1 regularization in estimating missing values.

5.2.2 Determining the Value of k

The number of nearest neighbors is an important parameter for local learning-based methods, including KNNimpute, SKNNimpute, IKNNimpute, LLSimpute, SLLSimpute, ShrinkageLLS, RLLSimpute_L2, and RLLSimpute_L1. To determine the approximately optimal number of neighbors for each method, we conduct experiments to investigate the relationship between the number of neighbors used and the

2. http://gerontech.hfut.edu.cn



(a) GDS38 (b) GDS39 (c) GDS1761

(d) GDS2967 (e) GDS3835 (f) GDS4831

Fig. 3. *RMSE* vs. the number of neighbors on the six datasets.

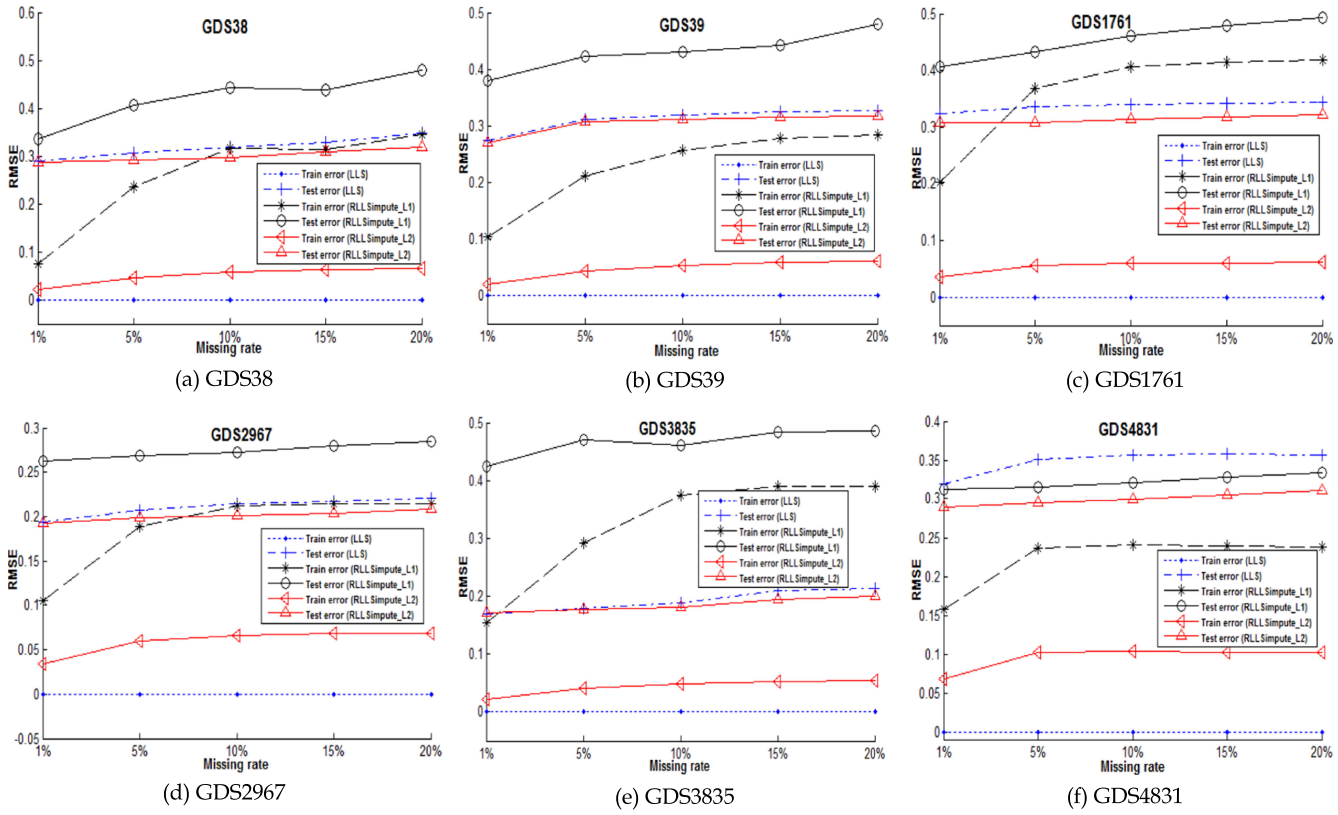
root mean squared errors at a representative missing rate of 5 precent. We conducted experiments thirty times with the number of neighbors ranging from 1 to 400. Fig. 3 presents the averaged root mean squared errors of the eight methods. The X-axis represents the number of candidate neighbors, k, and the Y-axis corresponds to the root mean squared errors. From Fig. 3, we can observe that for LLSimpute, SLLSimpute, and ShrinkageLLS, the *RMSE*s rise quickly when k approaches the number of samples of a dataset and then decreases with the increase of k. For KNNimpute, SKNNimpute, and IKNNimpute, the *RMSE*s first decrease with the use of more neighbors and then increase with the increase of k. The main reason is that as k increases, the contribution of weak relevant or irrelevant genes plays a dominant role, leading to accuracy decease. In contrast to the other seven methods, RLLSimpute_L2 tends to obtain smaller *RMSE*s with the increase of k and is relatively insensitive to the exact value of k. We observe that RLLSimpute_L1 behaves poorly, which is consistent to our previous analysis. In addition, we see that KNNimpute, SKNNimpute, and IKNNimpute achieve smaller root mean squared error when k ranges from 5 to 13. However, because nearest-neighbor-based methods fail to consider the relevance between the neighbors, nearest neighbor-based methods become worse with the increase of k.

5.2.3 Risk of Over-Fitting

Over-fitting refers to the situation that a model fits the training set well but works poorly on the unseen test set. That is, the model has a small training error but achieves high errors on the test set. In this study, to show how regularization techniques help achieve a tradeoff between model complexity and accuracy, we conduct experiments with three representative methods, including LLSimpute, RLLSimpute_L2 and RLLSimpute_L1, under different missing rates. Specifically, different from RLLSimpute_L2 and RLLSimpute_L1, there is no regularization term in LLSimpute. We record and plot their training errors and test errors. Fig. 4 presents the results. The X-axis represents the missing rates and the Y-axis corresponds to *RMSE*. From Fig. 4, we observe that LLSimpute works perfectly on the training set but achieves high test errors. This indicate that over-fitting occurs in LLSimpute. In contrast, RLLSimpute_L2 and RLLSimpute_L1 can mitigate the problem by reducing the model complexity for high predictive performance. We also see the superiority of RLLSimpute_L2 over RLLSimpute_L1.

5.2.4 Root Mean Squared Error

In this study, for each missing rate, we conduct experiments thirty times for each imputation method and each missing rate and obtained the average root mean squared error. Fig. 5 shows the experimental results of the eleven imputation methods. The X-axis denotes the missing rates and the Y-axis represents the *RMSE*. A smaller *RMSE* reflects lower deviation of imputation from corresponding true values. From Fig. 5, we can observe that *RMSE* increases with the increase of missing rates for all imputation methods. This is reasonable because a larger missing rate comes with the loss of more information. We can also observe that the *RMSE*s of LLSimpute, SLLSimpute, ShrinkageLLS, and RLLSimpute_L2 are close to each other at the missing rate of 1 precent. However, with the increase of missing rates,



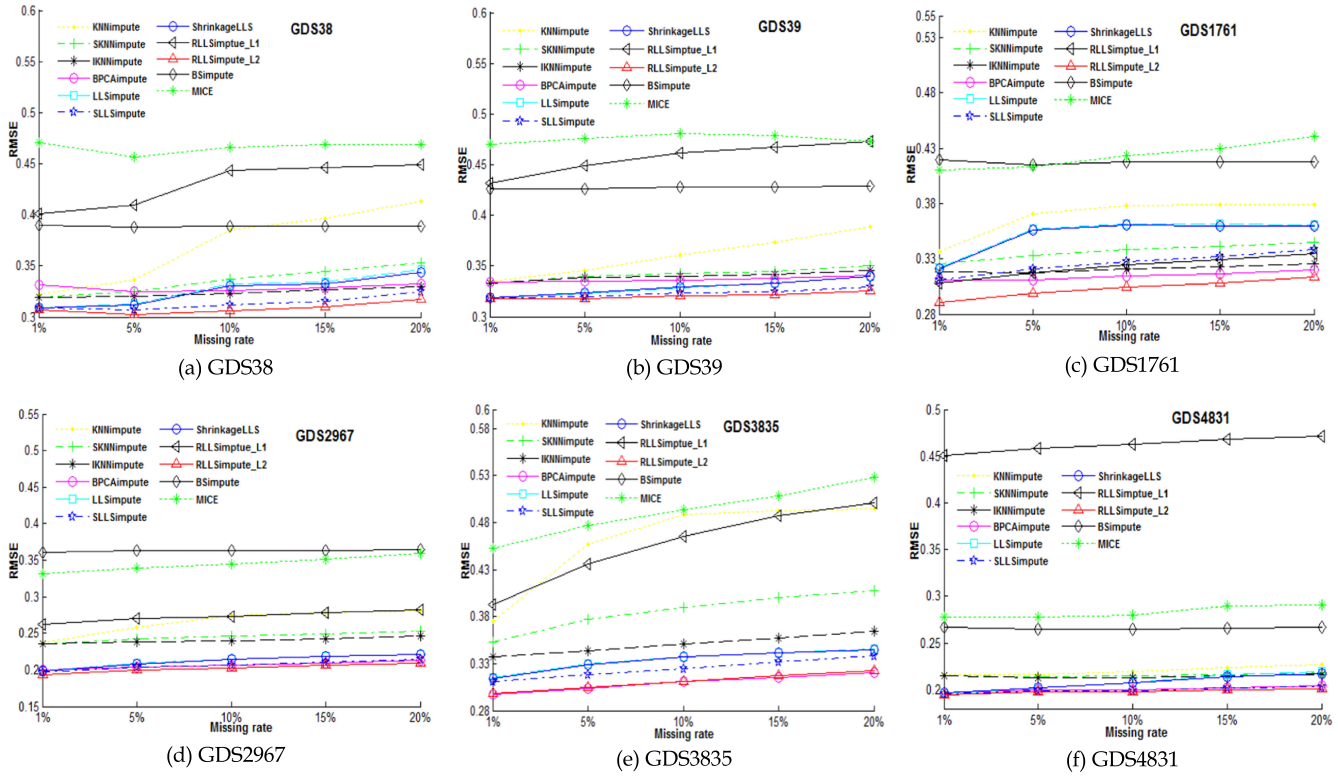
(a) GDS38 (b) GDS39 (c) GDS1761

(d) GDS2967 (e) GDS3835 (f) GDS4831

Fig. 4. Comparison of training and test errors on the six datasets.

RLLSimpute_L2 consistently outperforms its competitors. This is probably because LLSimpute, SLLSimpute, and ShrinkageLLS suffer from over-fitting when dealing with a larger number of similar genes.

In contrast to RLLSimpute_L1, RLLSimpute_L_2 can achieve a better tradeoff between handling overfitting and keeping the highly predictive variables. We also observe that least squares-based methods tend to obtain better imputation



(a) GDS38 (b) GDS39 (c) GDS1761

(d) GDS2967 (e) GDS3835 (f) GDS4831

Fig. 5. *RMSE* vs. different missing rates on the six datasets.

TABLE 2
Results of Significance Test on GDS38

Missing rate (%)	KNN impute	SKNN impute	IKNN impute	BPCA impute	LLS impute	SLLS impute	Shrinkage impute	RLLSimpute_L1	BS impute	MICE
1	0.0890	0.2413	0.2413	0.0890	0.6232	0.7913	0.7913	0.0008	0.0002	0.0002
5	0.0022	0.0073	0.0173	0.0091	0.1212	0.2730	0.1620	0.0002	0.0002	0.0002
10	0.0002	0.0006	0.0046	0.0036	0.0010	0.1859	0.0010	0.0002	0.0002	0.0002
15	0.0002	0.0004	0.0022	0.0028	0.0013	0.0376	0.0013	0.0002	0.0002	0.0002
20	0.0002	0.0003	0.0036	0.0022	0.0008	0.0890	0.0013	0.0002	0.0002	0.0002

TABLE 3
Results of Significance Test on GDS39

Missing rate (%)	KNN impute	SKNN impute	IKNN impute	BPCA impute	LLS impute	SLLS impute	Shrinkage impute	RLLSimpute_L1	BS impute	MICE
1	0.0452	0.0539	0.0376	0.1405	0.8501	1.0000	0.9097	0.0010	0.0002	0.0002
5	0.0010	0.0036	0.0036	0.0211	0.4727	0.5708	0.3075	0.0002	0.0002	0.0002
10	0.0002	0.0008	0.0017	0.0046	0.1620	0.4727	0.1212	0.0002	0.0002	0.0002
15	0.0002	0.0002	0.0002	0.0004	0.0058	0.4274	0.0058	0.0002	0.0002	0.0002
20	0.0002	0.0002	0.0002	0.0002	0.0008	0.1859	0.0013	0.0002	0.0002	0.0002

TABLE 4
Results of Significance Test on GDS1761

Missing rate (%)	KNN impute	SKNN impute	IKNN impute	BPCA impute	LLS impute	SLLS impute	Shrinkage impute	RLLSimpute_L1	BS impute	MICE
1	0.0002	0.0002	0.0003	0.0008	0.0002	0.0008	0.0002	0.0017	0.0002	0.0002
5	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
10	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
15	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
20	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002

TABLE 5
Results of Significance Test on GDS2967

Missing rate (%)	KNN impute	SKNN impute	IKNN impute	BPCA impute	LLS impute	SLLS impute	Shrinkage impute	RLLSimpute_L1	BS impute	MICE
1	0.0002	0.0002	0.0002	0.1041	0.2123	0.2730	0.2123	0.0002	0.0002	0.0002
5	0.0002	0.0002	0.0002	0.0539	0.0008	0.0539	0.0017	0.0002	0.0002	0.0002
10	0.0002	0.0002	0.0002	0.0173	0.0002	0.0073	0.0002	0.0002	0.0002	0.0002
15	0.0002	0.0002	0.0002	0.0008	0.0002	0.0006	0.0002	0.0002	0.0002	0.0002
20	0.0002	0.0002	0.0002	0.0036	0.0002	0.0004	0.0002	0.0002	0.0002	0.0002

accuracy than nearest neighbor-based methods. This indicates the superiority of regression models over nearest neighbor techniques. In Comparison with RLLSimpute_L2, BPCAimpute has a larger *RMSE* except on GDS3835 where BPCAimpute and RLLSimpute_L2 obtain similar imputation performance. This is probably because GDS3835 has a covariance structure in which situation global learning-based methods generally better estimated the missing values. Besides, because more missing entries leads to the damage of global data structure, RLLSimpute_L2 works better than BPCAimpute at a high missing rate. We can observe that MICE performs poorly on all datasets. The main reason is that MICE uses classification and regression tree as the conditional model and it desires a large number of samples for better performances, which is not the case of microarray data.

We further use the two-sided Wilcoxon rank sum test with a significance interval of 95 precent to determine whether there is any difference between RLLSimpute_L2 and its competitors in terms of *RMSE*. In our experiments, the difference is significant if the p-value is less than 0.05. Tables 2, 3, 4, 5, 6, and 7 present the experimental results on the six datasets, respectively. A value in tables less than 0.05 represents that RLLSimpute_L2 obtains smaller *RMSE* than its competitor, and a value greater than 0.05 means that there is no statistical difference between RLLSimpute_L2 and the competing method.

According to the results in Tables 2, 3, 4, 5, 6, and 7, we can observe that RLLSimpute_L2 almost demonstrates the best performance compared with the referenced methods, which further proves the effectiveness of L_2 regularization algorithm used in RLLSimpute_L2. Specifically, RLLSimpute_L2 consistently displays a smaller *RMSE* than any other method on GDS1761 under any missing rate. It is necessary to note that GDS1761 has 64 samples, this

TABLE 6
Results of Significance Test on GDS3835

Missing rate (%)	KNN impute	SKNN impute	IKNN impute	BPCA impute	LLS impute	SLLS impute	Shrinkage impute	RLLSimpute_L1	BS impute	MICE
1	0.0002	0.0002	0.0002	0.9097	0.0211	0.0539	0.0257	0.0002	0.0002	0.0002
5	0.0002	0.0002	0.0002	0.4727	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
10	0.0002	0.0002	0.0002	0.6776	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
15	0.0002	0.0002	0.0002	0.0640	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
20	0.0002	0.0002	0.0002	0.0640	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002

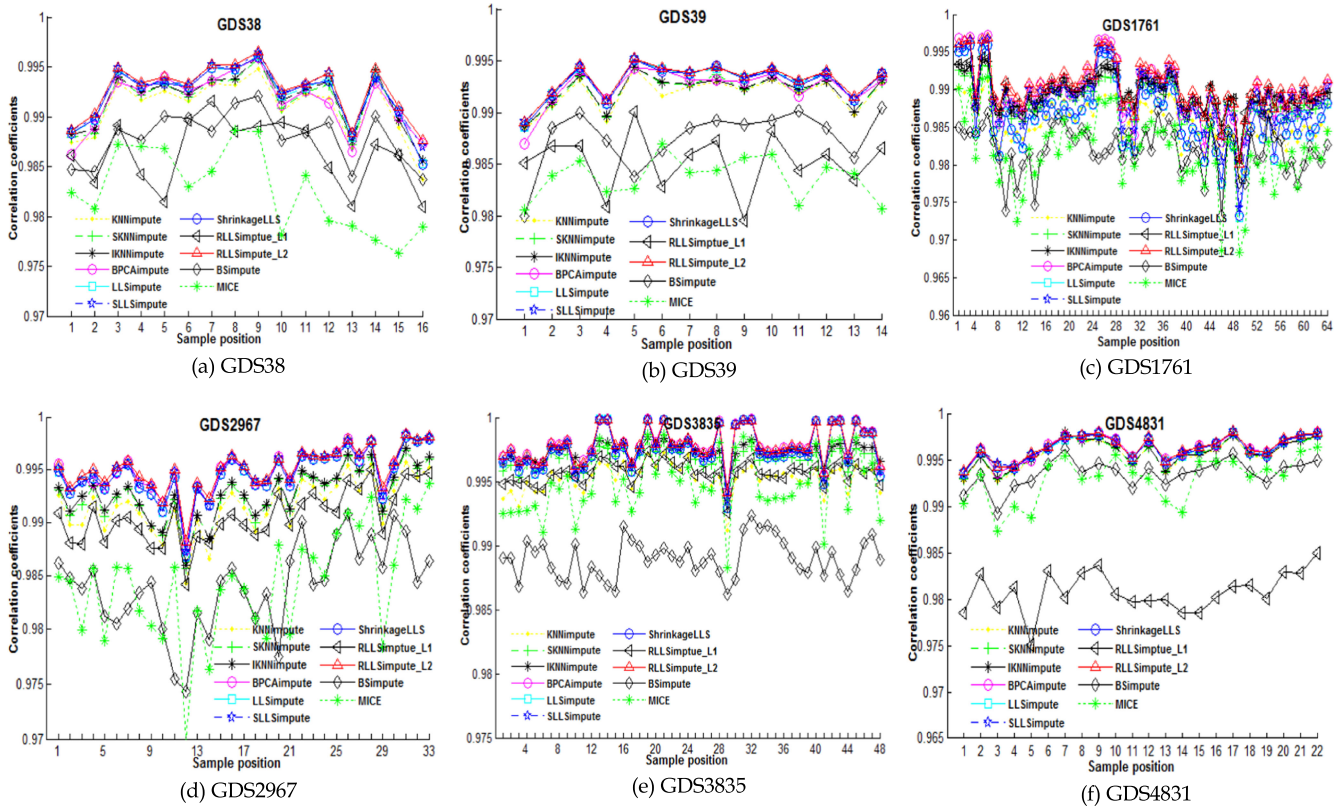
TABLE 7
Results of Significance Test on GDS4831

Missing rate (%)	KNN impute	SKNN impute	IKNN impute	BPCA impute	LLS impute	SLLS impute	Shrinkage impute	RLLSimpute_L1	BS impute	MICE
1	0.1405	0.1405	0.1405	0.7913	0.9097	0.7337	0.6776	0.0002	0.0008	0.0058
5	0.0113	0.0452	0.0640	0.4274	0.1859	0.4274	0.1859	0.0002	0.0002	0.0002
10	0.0073	0.0257	0.0539	0.5708	0.0890	0.6232	0.1212	0.0002	0.0002	0.0002
15	0.0003	0.0073	0.0113	0.4961	0.0091	0.5708	0.0140	0.0002	0.0002	0.0002
20	0.0002	0.0013	0.0058	0.2729	0.0022	0.3075	0.0028	0.0002	0.0002	0.0002

number is larger than any other dataset's number of samples. So, it seems that RLLSimpute_L2 prefers to show a better result when the number of samples in dataset is large enough. Furthermore, on GDS3835, BPCAimpute shows results close to that of RLLSimpute_L2 with a missing rate larger than 10 precent, but RLLSimpute_L2 outperforms it when the missing rate is as low as 5 precent. In addition, KNNimpute, SKNNimpute and IKNNimpute always perform worse than RLLSimpute_L2, which proves that these methods simply using the weighted averaging strategy are not sufficient to precisely estimate missing values in microarray data.

5.2.5 *Pearson Correlation Coefficient*

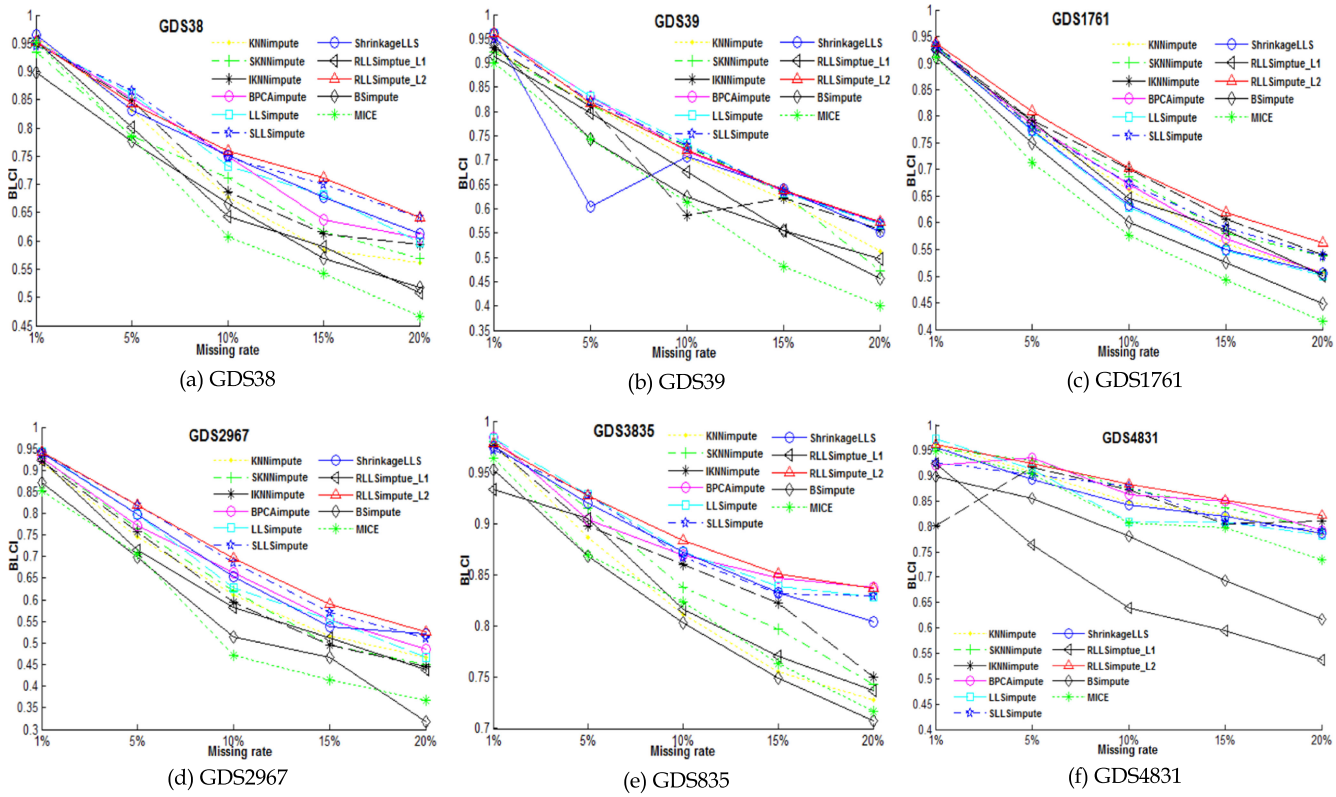
To show the effectiveness of different imputation methods in preserving the original data structure, we conduct experiments with a representative 5 precent missing rate for the eleven methods. Fig. 6 presents corresponding experimental results for different methods. The X-axis denotes the index of a sample and Y-axis represents the Pearson correlation



(a) GDS38 (b) GDS39 (c) GDS1761

(d) GDS2967 (e) GDS3835 (f) GDS4831

Fig. 6. Pearson correlation coefficients of different imputation methods on the six datasets.



(a) GDS38 (b) GDS39 (c) GDS1761

(d) GDS2967 (e) GDS835 (f) GDS4831

Fig. 7. *BLCI* vs. different missing rates on the six datasets.

coefficient. For Fig. 6, we can observe that RLLSimpute_L2 consistently achieves better performance than the other imputation methods. This means that RLLSimpute_L2 can better preserve the structure of original dataset.

In addition, from Figs. 5 and 6, we can see that methods with a larger *RMSE* may better preserve data structure. For example, LLSimpute has a smaller *RMSE* than BPCAimpute on GDS38, while BPCAimpute achieves a higher Pearson correlation coefficient. This is because *RMSE* reflects the overall degree of deviation of imputation, while Pearson correlation coefficient works at the sample level.

5.2.6 Biomarker List Concordance Index

Biomarker list concordance index (*BLCI*) is used for measuring the preservation of differentially expression genes that are selected from the original microarray data and the estimated microarray data [4]. Fig. 7 presents the experimental results of *BLCI* for different imputation methods. The X-axis represents the missing rates, and the Y-axis denotes the *BLCI*. From Fig. 7, we see a general trend on all experimental datasets for all methods: a larger missing rate leads to a lower *BLCI*, and *BLCI* decreases quickly with the increase of missing rates. For example, at the missing rate of 1 precent, most imputation methods obtain a *BLCI* as high as 0.95 and even reach up to 0.97; but at the missing rate of 20 precent, *BLCI* decrease to 0.52. This indicates that a larger missing rate brings a greater challenge to the identification of differentially expressed genes. We can also observe that RLLSimpute_L2 and BPCAimpute generally obtain a higher *BLCI* than other nine imputation methods. In comparison with BPCAimpute, RLLSimpute_L2 obtains a slightly lower *BLCI* at a low missing rate, and performs better when working on a dataset with high missing rates. This is mainly because that a low missing rate comes with less information loss, in which situation that BPCAimpute can utilize the covariance structure. Hence, the deterioration of BPCAimpute in imputation performance is not surprising for the case of a high missing rate. Similar to the case of *RMSE*, we can observe that least squares-based methods perform better than nearest neighbor-based method.

Overall, according to the above extensive experimental results and analysis, we conclude that compared with one global learning-based method (BPCAimpute), three nearest neighbor-based methods (KNNimpute, SKNNimpute, and IKNNimpute), three least squares-based methods (LLSimpute, SLLSimpute, and ShrinkageLLS), and RLLSimpute_L1, RLLSimpute_L2 achieves smaller root mean squared errors in the statistical view and better preserves the structure of original dataset in biological view. In addition, compared with RLLSimpute_L2, BPCAimpute is sensitive to the type of microarray data being analyzed. When dominant local similarity structures exist among genes, BPCAimpute is generally less accurate than RLLSimpute_L2. Even though BPCAimpute shows slightly better performance than RLLSimpute_L2 on the dataset having covariance structure, RLLSimpute_L2 tends to obtain better imputation results than BPCAimpute when a high percentage of entries are missing.

5.2.7 Time Complexity Analysis

For a microarray dataset with m genes and n samples $(m \gg n)$, suppose that the number of used neighbors is k, then the time complexity of KNNimpute is $O(m^2 n)$ and the time complexity of SKNNimpute is $O(m \log m + mn \log m)$. Suppose the number of iterations of IKNNimpute is i, then its time complexity is $O(mn + imn \log m)$. Least squares based methods take $O(k^3)$ to calculate the inverse of a $k * k$

matrix, then the time complexity of LLSimpute, ShrinkageLLS, and SLLSimpute is $O(m(mn+k^3))$, $O(m(mn+k^3)+m)$, and $O(m\log m+m(mn+k^3))$, respectively. For RLLSimpute_L2 and RLLSimpute_L1, the time complexity of lasso or ridge regression is $O(k^3+k^2n)$. So, the time complexity of RLLSimpute_L2 and RLLSimpute_L1 are both $O(m(k^3+k^2n+mn)+m\log m)$, where $O(m\log m)$ is the time complexity for sorting genes. BPCAimpute is built on probabilistic principal component analysis and its time complexity is $O(m(mn^2+mn(n-1)))=O(m^2n^2)$. For BSimpute, its time complexity is $O((un+m)mq)$, where u represents the number of iterations in selecting similar genes and q represents the number of missing entries. For MICE using a classification and regression tree as the building block, its time complexity is $O(inm^2\log m)$, where i is the number of iterations. Herein, we can conclude the followings. (1) MICE has the worst time complexity and BSimpute has the best time complexity. (2) BPCAimpute has higher time complexity than that of RLLSimpute_L2 and RLLSimpute_L1. (3) RLLSimpute_L2 and RLLSimpute_L1 cost more time than least squares-based methods (LLSimpute, SLLSimpute, and ShrinkageLLS), but they have the same order of magnitude. (4) Nearest neighbor-based methods are more time-efficient than least squares-based methods.

6 Conclusion

Microarray missing value imputation is a challenging, yet meaningful research field in the analysis of gene expression profiles. Although there is a wealth of imputation methods available, most local learning-based methods tend to suffer from the over-fitting problem. In this study, we proposed a regularized local learning-based method for missing value imputation. Specifically, we trained a L_2 Regularized Local Least Squares imputation model (RLLSimpute_L2) between the target gene and its neighbors for estimating the missing values of the target genes. To utilize previously estimated values, RLLSimpute_L2 imputes the missing values in ascending order according to the missing rates associated with each target gene. Besides, we explored the use of L_1 regularization and proposed the L_1 Regularized Local Least Squares imputation model (RLLSimpute_L1). Finally, we conducted experiments on six microarray datasets and compared RLLSimpute_L2 and RLLSimpute_L1 with nine state-of-the-art imputation methods in terms of three evaluation metrics. Experimental results demonstrate the effectiveness of regularization techniques in mitigating the risk of over-fitting and the superiority of RLLSimpute_L2 over its competing ones in missing value imputation.

For the future work, because the distribution of missing values is an important factor for evaluating an imputation method, investigating the relationship between the performance of an imputation method and a specific distribution of missing values remains a topic for future research. Second, the proposed imputation algorithm could be applied to other fields that also suffer from the problem of missing values such as proteomics and RNA-seq datasets [44]. Typically, RNA-seq data, obtained with massively parallel sequencing methods in transcript analysis, are the time series RNA sequencing data and may have missing read counts. Such data have similar expression profiles and exhibit local similarity structure within a cell population, indicating that we can utilize correlations among genes or cells as a basis for missing value estimation. On the other hand, RNA-seq data measure the gene expression using read counts rather than intensities of microarray data. That is, we need to consider the countability property of RNA-seq data for missing value imputation. Therefore, how to apply the proposed method to RNA-seq data and to further test its performance remains another topic.

Acknowledgments

This work was supported in part by the China Postdoctoral Science Foundation (No. 2016M592046), the National Natural Science Foundation of China (No. 71661167004), the Anhui Key Research and Development Plan (No. 2017-222), the Fundamental Research Funds for the Central Universities (No. JZ2016HGBH1053), the Science and Technology Innovation Project of Foshan City (No. 2015IT100095), and the "111 Project" of the Ministry of Education and State Administration of Foreign Experts Affairs (No. B14025).

References

[1] D. Lockhart and E. Winzeler, "Genomics, gene expression and DNA arrays," *Nature*, vol. 405, no. 6788, pp. 827–836, Jun. 2000.
[2] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, and C. Bloomfield, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Sci.*, vol. 286, no. 5439, pp. 531–537, Oct. 1999.
[3] J. Mirus, Y. Zhang, C. Li, A. Lokshin, R. Prentice, S. Hingorani, and P. Lampe, "Cross-species antibody microarray interrogation identifies a 3-protein panel of plasma biomarkers for early diagnosis of pancreas cancer," *Clin. Cancer Res.*, vol. 21, no. 7, pp. 1764–1771, Apr. 2015.
[4] Y. Xiang, Y. Fei, X. Chen, and C. Dong, "Bioinformatics analysis to identify the differentially expressed genes of glaucoma," *Mol. Med. Rep.*, vol. 12, no. 4, pp. 4829–4836, Oct. 2015.
[5] C. He, C. Zhao, G. Li, W. Zhu, W. Yang, and M. Yang, "A hybrid iterative approach for microarray missing value estimation," in *Proc. 2016 IEEE Int. Conf. Bioinf. Biomed.*, 2016, pp. 1350–1352.
[6] M. Lenz, F. Müller, M. Zenke, and A. Schuppert, "Principal components analysis and the reported low intrinsic dimensionality of gene expression microarray data," *Sci. Rep.*, vol. 6, Jun. 2016, Art. no. 25696.
[7] A. Wang, N. An, Y. Yang, G. Chen, L. Li, and G. Alterovitz, "Wrapper-based gene selection with Markov blanket," *Comput. Biol. Med.*, vol. 81, pp. 11–23, Feb. 2017.
[8] L. Dyrskjot, T. Thykjaer, M. Kruhoffer, J. Jensen, N. Marcussen, S. Hamilton-Dutoit, H. Wolf, and T. Ørntoft, "Identifying distinct classes of bladder carcinoma using microarrays," *Nat. Genetics*, vol. 33, no. 1, pp. 90–96, Jan. 2003.
[9] F. Meng, C. Cai, and H. Yan, "A bicluster-based bayesian principal component analysis method for microarray missing value estimation," *IEEE J. Biomed. Health Inf.*, vol. 18, no. 3, pp. 863–871, May 2014.
[10] M. Souto, P. Jaskowiak, and I. Costa, "Impact of missing data imputation methods on gene expression clustering and classification," *BMC Bioinf.*, vol. 16, no. 1, Feb. 2015, Art. no. 64.
[11] R. Priya and R. Sivaraj, "Pre-processing of microarray gene expression data for classification using adaptive feature selection and imputation of non-ignorable missing values," *Int. J. Data Min. Bioin.*, vol. 16, no. 3, pp. 183–204, 2016.
[12] S. Taylor, L. Sandra, L. Ruhaak, K. Karen, R. Weiss, and K. Kim, "Effects of imputation on correlation: implications for analysis of mass spectrometry data from multiple biological matrices," *Brief. Bioinf.*, vol. 18, no. 2, pp. 312–320, Mar. 2017.
[13] A. Liew, N.-F. Law, and H. Yan, "Missing value imputation for gene expression data: computational techniques to recover missing data from available information," *Brief. Bioinf.*, vol. 12, no. 5, pp. 498–513, Sep. 2011.

[14] A. Butte, J. Ye, G. Niederfellner, K. Rett, H. Häring, M. White, and I. Kohane, "Determining significant fold differences in gene expression analysis," in *Proc. Pacific Symp. Biocomput.*, pp. 6–17, Feb. 2001.
[15] R. Jörnsten, H. Wang, W. Welsh, and M. Ouyang, "DNA microarray data imputation and significance analysis of differential expression," *Bioinf.*, vol. 21, no. 22, pp. 4155–4161, Nov. 2005.
[16] O. Alter, P. Brown, and D. Botstein, "Singular value decomposition for genome-wide expression data processing and modeling," *Proc. Nat. Academy Sci. USA*, vol. 97, no. 18, pp. 10101–10106, Aug. 2000.
[17] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. Altman, "Missing value estimation methods for DNA microarrays," *Bioinf.*, vol. 17, no. 6, pp. 520–525, Jan. 2001.
[18] K. Moorthy, M. Mohamad, and S. Deris, "A review on missing value imputation algorithms for microarray gene expression data," *Curr. Bioinf.*, vol. 9, no. 1, pp. 18–22, Feb. 2014.
[19] Y. Yang, Z. Xu, and D. Song, "Missing value imputaiton for microRNA expression data by using a GO-based similarity measure," *BMC Bioinf.*, vol. 17, no. suppl 1, Jan. 2016, Art. no. 10.
[20] S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubara, and S. Ishii, "A Bayesian missing value estimation method for gene expression profile data," *Bioinf.*, vol. 19, no. 16, pp. 2088–2096, Nov. 2003.
[21] K. Kim, B. Kim, and G. Yi, "Reuse of imputed data in microarray analysis increases imputation efficiency," *BMC Bioinf.*, vol. 5, no. 1, Oct. 2004, Art. no. 160.
[22] L. Brás and J. Menezes, "Improving cluster-based missing value estimation of DNA microarray data," *Biomol. Eng.*, vol. 24, no. 2, pp. 273–282, Jun. 2007.
[23] T. Bø, B. Dysvik, and I. Jonassen, "LSimpute: accurate estimation of missing values in microarray data with least squares methods," *Nucleic Acids Res.*, vol. 32, no. 3, Feb. 2004, Art. no. e34.
[24] H. Kim, G. Golub, and H. Park "Missing value estimation for DNA microarray gene expression data: local least squares imputation," *Bioinf.*, vol. 21, no 2, pp. 187–198, Jan. 2005.
[25] C. He, H. Li, C. Zhao, G. Li, and W. Zhang, "Triple imputation for microarray missing value estimation," in *Proc. Int. Conf. Bioinf. Biomed.*, 2015, pp. 208–213.
[26] H. Li, C. Zhao, F. Shao, G. Li, and X. Wang, "A hybrid imputation approach for microarray missing value estimation," *BMC Genomics*, vol. 16, no. 9, pp. 1–11, Aug. 2015.
[27] P. Keerin, W. Kurutach, and T. Boongoen, "A cluster-directed framework for neighbour based imputation of missing value in microarray data," *Int. J. Data Min. Bioin.*, vol. 15, no. 2, pp. 165–193, 2016.
[28] F. Shi, D. Zhang, J. Chen, and H. Karimi, "Missing value estimation for microarray data by Bayesian principal component analysis and iterative local least squares," *Math. Probl. Eng.*, vol. 16, pp. 301–312, Mar. 2013.
[29] X. Zhang, X. Song, H. Wang, and H. Zhang, "Sequential local least squares imputation estimating missing value of microarray data," *Comput. Biol. Med.*, vol. 38, no. 10, pp. 1112–1120, Oct. 2008.
[30] T. Chai and R. Draxler, "Root mean square error (rmse) or mean absolute error (mae)?" *Geosci. Model Dev. Discuss.*, vol. 7, pp. 1525–1534, Feb. 2014.
[31] C. Truntzer, C. Mercier, J. Estève, C. Gautier, and P. Roy, "Importance of data structure in comparing two dimension reduction methods for classification of microarray gene expression data," *BMC Bioinf.*, vol. 8, no. 1, pp. 1–12, Mar. 2007.
[32] I. Scheel, M. Aldrin, I. Glad, R. Sørum, H. Lyng, and A. Frigessi, "The influence of missing value imputation on detection of differentially expressed genes from microarray data," *Bioinf.*, vol. 21, no. 23, pp. 4272–4289, Dec. 2005.
[33] S. Oh, D. Kang, G. Brock, and G. Tseng, "Biological impact of missing-value imputation on downstream analyses of gene expression profiles," *Bioinf.*, vol. 27, no. 1, pp. 78–86, Jan. 2011.
[34] V. Tusher, R. Tibshirani, and G. Chu, "Significance analysis of microarrays applied to the ionizing radiation response," *Proc. Nat. Academy Sci. USA*, vol. 98, no. 9, pp. 5116–5121, Apr. 2001.
[35] Y. Chen, A. Wang, H. Ding, X. Que, Y. Li, N. An, and L. Jiang, "A global learning with local preservation method for microarray data imputation," *Comput. Biol. Med.*, vol. 77, pp. 76–89, Oct. 2016.
[36] P. Spellman, G. Sherlock, M. Zhang, V. Iyer, K. Anders, M. Eisen, P. Brown, D. Botstein, and B. Futcher, "Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization," *Mol. Biol. Cell*, vol. 9, no. 12, pp. 3273–3297, Dec. 1998.
[37] O. Sarig, "Bar1-deficient mating type a cells response to alpha mating factor: Time course and dose response," 2007. [Online]. Available: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE8982
[38] D. Ross, U. Scherf, M. Eisen, C. Perou, C. Rees, P. Spellman, V. Iyer, S. Jeffrey, M. Van de Rijn, M. Waltham, and A. Pergamenschikov, "Systematic variation in gene expression patterns in human cancer cell lines," *Nature Genetics*, vol. 24, no 3. pp. 227–235, Mar. 2000.
[39] C. Artieri and R. Singh, "Molecular evidence for increased regulatory conservation during metamorphosis, and against deleterious cascading effects of hybrid breakdown in Drosophila," *BMC Biol.*, vol. 8, no. 1, Mar. 2010, Art. no. 26.
[40] Y. Lee, J. Andersen, H. Song, A. Judge, D. Seo, T. Ishikawa, J. Marquardt, M. Kitade, M. Durkin, C. Raggi, and H. Woo, "Definition of ubiquitination modulator COP1 as a novel therapeutic target in human hepatocellular carcinoma," *Cancer Res.*, vol. 70, no. 21, pp. 8264–8269, Nov. 2010.
[41] S. Chattopadhyay, C. Das, and S. Bose, "A novel biclustering based missing value prediction method for microarray gene expression data," in *Proc. Int. Conf. Man Mach. Interfac.*, 2015, pp. 1–6.
[42] L. Burgette and J. Reiter, "Multiple imputation for missing data via sequential regression trees," *Am. J. Epidemiol.*, vol. 172, no. 9, pp. 1070–1076, 2010.
[43] H. Wang, C. Chiu, Y. Wu, and W. Wu, "Shrinkage regression-based methods for microarray missing value imputation," *BMC Syst. Biol.*, vol. 7, no. Suppl 6, 2013, Art. no. S11.
[44] M. Grabherr, B. Haas, M. Yassour, J. Levin, D. Thompson, I. Amit, and A. Regev, "Full-length transcriptome assembly from RNA-Seq data without a reference genome," *Nat. Biotechnol.*, vol. 29, pp. 644–652, 2011.

Aiguo Wang received the bachelor's degree from the School of Computer and Information, Hefei University of Technology, China, in 2010, and the PhD degree in computer applied technology from the Hefei University of Technology, in 2015. He is currently a post doctor in the School of Computer and Information, Hefei University of Technology. His research interests include bioinformatics, data mining, activity recognition, and assisted living systems.

Ye Chen He received the bachelor's degree in computer science and technology, from the School of Computer, Henan Polytechnic University, China, in 2014. He is now a graduate student with the School of Computer and Information, Hefei University of Technology. His research interests include bioinformatics and data mining.

Ning An received the bachelor's degree from Lanzhou University, China, in 1993, and the PhD degree in computer science and engineering from the Pennsylvania State University, in 2002. He worked at Oracle, Inc. as technical staff for 10 years. He is currently a "Yellow Mountain" professor with the School of Computer and Information, Hefei University of Technology. His research interests include gerontechnology, data mining, and mobile health technologies.

Jing Yang received the BS and PhD degrees from the Hefei University of Technology, in 2004, and 2013. She is currently an associate professor with the School of Computer and Information, Hefei University of Technology, China. Her current research interests include bioinformatics, artificial intelligence, data mining, and Bayesian network.

Lian Li He received the bachelor's degree from the School of Mathematics, Northwest Normal University, in 1976, and the master's degree from Lanzhou University, in 1982. He is now a professor of Hefei University of Technology. His research interests include computational theory, computational mathematics, grid computing, and social computing.

Lili Jiang received the bachelor's degree from Lanzhou University, in 2005, and the doctoral degree from the School of Information Science and Engineering, Lanzhou University, in 2012. She is now an assistant professor with Umeå University. Her research interests include data federation, crowdsourcing, entity search, natural language processing, and information retrieval.