EMMA: An EM-based Imputation Technique for Handling Missing Sample-Values in Microarray Expression Profiles.

Amitava Karmaker[1]*, Edward A. Salinas[2], Stephen Kwek[3]

[1]University of Wisconsin-Stout, Menomonie, Wisconsin 54751, USA
[2]Johns Hopkins University, Baltimore, Maryland 21218, USA
[3]Microsoft Corporation, Redmond, Washington 98052, USA

Abstract *- Data with missing sample-values are quite common in many microarray expression profiles. The outcome of the analysis of these microarray data mostly depends on the quality of underlying data. In fact, without complete data, most computational approaches fail to deliver the expected performance. So, filling out missing values in the microarray, if any, is a prerequisite for successful data analysis. In this paper, we propose an Expectation-Maximization (EM) inspired approach that handles a substantial amount of missing values with the objective of improving imputation accuracy. Here, each missing sample-value is iteratively filled out using an updater (predictor) constructed from the known values and predicted values from the previous iteration. We demonstrate that our approach significantly outperforms some standard methods in terms of treating missing values, and shows robustness in increasing levels of missing rates.*

Keywords: Microarray, Biological Data Mining, Missing Sample Value Estimation, EM Algorithm.

1 Introduction

Since the last decade, microarray technology has been applied as one of the widely used tools for gene expression profiling across various experimental conditions [1]. It generates enormous amounts of data that can be visualized and analyzed by computational tools. As a precondition for effective data analysis, microarray profiling data need to be preprocessed to ensure superior data quality. Due to intrinsic experimental settings and erroneous hybridization processes, very often microarray data contain missing values (probes), which deteriorate the subsequent analysis significantly. Studies found that on an average a microarray dataset contains ~5% missing values, and ~60% of the genes typically have at least one feature (sample) value missing [2]. Nevertheless, several data analysis algorithms, namely principal component analysis (PCA) [3], support vector machines (SVM) [4-6], singular value decomposition (SVD) [7], artificial neural network (ANN) [8] etc., require fairly complete datasets to perform stably. In addition, unsupervised clustering (e.g. hierarchical clustering[9]) suffers from missing values while constructing clusters using distance measures. Moreover, because of higher expenses, sometimes replications of experiments are not often feasible. In order to ensure better analysis, incomplete microarray data are required to be preprocessed and reasonably complete.

In this paper, we propose an iterative technique to handle missing values in microarray data. Our method is inspired by the EM (Expectation Maximization) algorithm, which is widely used for missing value imputation in data preprocessing. In our algorithm, we try to implement an updater which will eventually estimate the most appropriate values replacing the imputed values in the preceding iterations. Unlike other methods, our technique can estimate the unknown values in the dataset and fill out the entries in the single dataset without incorporating "reference datasets". Empirically, we tested our method on six publicly available *Saccharomyces cerevisiae* (Yeast) microarray datasets and evaluated the performance measures. Our findings outperform other existing techniques considerably and tend to be quite robust in higher missing rates.

2 Related works

As we know, microarray is a large matrix of expression levels of genes (rows) under differential experimental conditions/derived from various samples (columns). The general hypothesis behind estimating missing values for microarray is to capture the inherent association among the underlying rows and columns, and infer new values for the missing ones taking this relationship into account as a whole. To preserve better correlations among the data values, sometimes gene expressions with missing values are discarded from further considerations. But it might not be an option if the most of the gene expressions have some of their values missing. Another simple way to deal with missing values is to impute average gene expression over the row [10]. Besides these, Troyanskaya et al. [11] proposed a

*Corresponding author

estimation method based on singular value decomposition (SVDimpute) [11]. Another Bayesian principal component analysis (BPCA) [12] based imputation algorithm was presented by Oba et al. [12], which assumes higher covariance among the gene expressions to estimate unknown values. These global imputation approaches are suitable for datasets with considerably large number of samples (~30) having strong global associations among them (e.g. temporal/time series datasets). On the other hand, there are quite a number of techniques for local missing value estimation. For example, k-nearest neighbor based KNNimpute [11], least square (LSImpute) [13], local least square (LLS) [14], etc. can handle relatively smaller datasets. To start with, these methods select neighborhood genes by Euclidean distance measures or Pearson's Correlations as required. The next step involves predicting the missing vales based on selected genes' expression pattern. Still, these methods are error prone due to noise and insufficient samples. One of the shortcomings of all these methods is to integrate multiple datasets from diverse sources and consolidate those for analysis. Combination of datasets without proper relevance may critically degenerate the quality of neighborhood gene analysis, as pointed out for KNNimpute [12]. To this end, Tuikkala et al. [15] devised a gene ontology based technique GOImpute, which separates functionally related genes for further imputation. This method outperforms KNNimpute, but its performance is dependent on the availability of enough genes and accuracy of their annotations. Finally, we found another order statistics based approach called integrative Missing Value Estimation (iMISS) [16], which improves LSS algorithm and subsequently beats the GOImpute in terms of imputation accuracy.

Table 1: Description of the test datasets (Datasets denoted with (*) are time series)

Dataset	No of genes/ instances	No. of samples/ attribute	Description
Diauxic*	5289	7	Metabolic transition from fermentation to respiration
Adaptive	3685	4	Evolutionary adaptability
Phosphate	5257	8	Polyphosphate metabolism
Alpha-factor*	4053	18	Yeast Cell cycle-regulation
Elutriation*	5192	14	
CDC15*	4833	13	

3 Methods and materials

3.1 Description of datasets used

To test our algorithms, we collected microarray profile data of *Saccharomyces cerevisiae* (Yeast) from the Princeton *Saccharomyces* Genome Database SGD Lite (http://sgdlite.princeton.edu/), a publicly accessible yeast microarray data repository. Our datasets are composed of both time series and non-time series data. The first dataset (Diauxic) we selected is a time series, spotted cDNA microarray gene expression profiles dealing with metabolic shift from fermentation to respiration in yeast [17]. The second dataset (Adaptive) is on the study of adaptability of yeast and their differential gene expressions under diverse stress conditions[18]. Another dataset (Phosphate) reports the resulting gene expressions for phosphate accumulation and polyphosphate metabolism [19]. The rest of the datasets (Alpha-factor, Elutriation, CDC15) were created from Spellman time series cell-cycle datasets [20] based on the methods used for yeast cultures. These three datasets are all temporal and comprise higher sample dimensions. The characteristics of the datasets used are furnished in Table 1.

3.2 The Expectation-Maximization (EM) Algorithm

A popular way of dealing with missing values is to use the Expectation-Maximization (EM) algorithm introduced by Dempster, Laird and Rubin [21]. Here, the data source is assumed to be from a certain (mixture of) parametric model(s). EM algorithm tends to perform very well in parameter estimation. EM iteratively performs the following two steps.

Estimation (E) step: Estimate the parameters in the probabilistic model for the data source by using the known attribute-values and estimates of the missing attribute values obtained in the previous iteration of the M-step.

Maximization (M) step: Fill in the missing values to maximize the likelihood function that is refined in the E-step.

There are two drawbacks in using EM algorithm to fill up missing values. Firstly, it assumes that the data source comes from some parametric model (or a mixture of parametric models) with a finite mixture of Gaussian (k-Gaussians) being the most commonly used. Due to this assumption, most EM applications are applicable to numerical attributes only. Secondly, while EM can be proved to converge (with the appropriate parametric model assumption), the convergence process tends to be extremely slow. In particular, EM algorithm is useful when maximum likelihood estimation of a complete data model is relatively easy. Ouyang et al. [22] showed the use of microarray data for Gaussian mixture clustering and imputation. This research originated when we tried to investigate whether imputation accuracy can be improved by using EM algorithm in filling up

missing numerical attribute-values, which is literally appropriate for microarray data.

EMMA **(η_{known}, $\eta_{missing}$)**

//Here $\eta_{known = 0.0}$, $\eta_{missing = 1.0}$, H_i =Linear Regressor

Initialize:

Fill the missing values using its mean (for continuous values).

Update:

Repeat the following two steps until convergence (*k* iterations).

E-step:

for each attribute x_i **do**

Construct an updater H_i for x_i.

M-step:

for each attribute x_i **do**

if x_i's value was missing then

$\eta \leftarrow \eta_{missing}$

else

$\eta \leftarrow \eta_{known}$

$x_i \leftarrow \eta\, H_i(x) + (1 - \eta)\, x_i$

Output:

The final updaters for filling in the missing values.

Figure 1: Pseudo code of our algorithm, *EMMA*.

3.3 Our iterative technique for handling missing values

Instead of using common parametric models, we assume that the value of each attribute is somehow dependent on the values of the other attributes, which can be captured to a certain extent by simple linear regressor. In fact, this assumption is quite rational for analyzing microarray data derived from particular stand-alone (not distributed) experiment, be it temporal or not.

Inspired by EM approach, we propose an iterative algorithm, which is EMMA (EM on MicroArray) by the name. In the E-step, we build a linear regressor, which we call updater Hi, for each attribute xi using the other attributes as input. In the M-step, we update the predicted value of those attributes based on these models constructed in the E-step as shown in Figure 1. The refined values are then used in the subsequent iterations to construct the updaters. Initially, if the sample value is missing, we use the mean values for first imputation. Because of the property of convergence at local maximum (saddle point) of EM algorithm, we need to start up with somewhat known (filled out) values. That is why we initiate with mean value imputation for the missing ones.

We continue this process iteratively until a certain number of iterations is reached or the attributes cease to change much. The rate of refinement of certain sample value is moderated by the parameter η (eta). Our experiment sets η to 1.0 (specified by $\eta_{missing}$) for attributes (samples) with missing values, as they can be replaced with completely new values. On the other hand, η was valued at 0.0 (specified by η_{known}) for attributes (samples) without missing values to restrict drastic changes of values over iterations. These values of η are not fully optimized in order to prevent overfitting. Besides, we obtained outperforming results using these non-optimized parameters, and also values of η may be fine-tuned for better yields.

3.4 Experimental settings

To construct test datasets, we removed the gene with missing values from these datasets, so that we can calculate the accuracy of missing value imputations more precisely. The experiments use source code from the machine learning software WEKA[23]. Missing values are artificially added to the data sets to simulate randomized missing values. To introduce *m*% missing values per attribute x_i in a data set of size *n*, we randomly selected *mn* instances and replaced its x_i-value with an "unknown" (In WEKA, missing values are denoted as '?') label. Missing values were added in the original data sets from which both training data sets and test data sets were generated. In each set of experiment, we used increasing levels of 'missingness' - missing rate: *m* = 1%, 5%, 10%, 20%, 25%, and 50%. We find that often at $m \geq 10\%$, the majority of the instances (genes) have some missing values, while at $m \geq 25\%$, all instances (genes) have some missing values. Moreover, as ours is an iterative approach, we recorded the performance metrics at increasing number of iterations, *T* = 1, 2, 5, 10, 15, 20, 25, 50 respectively.

3.5 Performance Evaluation

In order to validate the efficacy of our imputation method, we used Root Mean Squared Error (RMSE)[24, 25] (See Equation 1) performance metric, which estimates the relative closeness of the predicted and actual values. To minimize the variances in the RMSE measures, we created as many as ten datasets at same missing level (*m*%), and finally took average of all ten performance figures. There are other formulations of RMSE than that in Equation 1. The reason why we choose this expression because in the ideal case (null imputation algorithm), this RMSE measure will stand out as zero (null). This ensures that we can use this metric to compare same datasets using various algorithms. The closer the predicted (estimated) values to the actual values, the lesser the RMSE values are, resulting in the values of RMSE ~ 0.0 for almost correct prediction.

$$RMSE = \sqrt{\frac{mean\{\vec{Y}_{predicted} - \vec{Y}_{actual}\}^2}{mean\{\vec{Y}_{actual}\}^2}} \quad (1)$$

Table 2: RMSE measures of six datasets at different iterations for different missing rates (*m%*)

Missing rate (*m*)	**1%**	**5%**	**10%**	**20%**	**25%**	**50%**
# of Iterations (*T*)	**Diauxic**					
1	0.05617	0.12377	0.18004	0.26307	0.30318	0.45581
2	0.05598	0.12312	0.17718	0.25392	0.29049	0.42738
5	0.05591	0.12305	0.17669	0.25264	0.28854	0.43352
10	0.05590	0.12307	0.17679	0.25288	0.28897	0.45119
20	0.05590	0.12307	0.17681	0.25296	0.28914	0.45561
50	0.05590	0.12307	0.17681	0.25296	0.28915	0.45695
	Adaptive					
1	0.06932	0.15188	0.21863	0.31998	0.35122	0.50188
2	0.06924	0.15117	0.21685	0.31247	0.34323	0.48526
5	0.06922	0.15112	0.21699	0.31202	0.34435	0.49516
10	0.06921	0.15112	0.21700	0.31203	0.34454	0.50235
20	0.06921	0.15112	0.21700	0.31203	0.34454	0.50522
50	0.06921	0.15112	0.21700	0.31203	0.34454	0.50554
	Phosphate					
1	0.07581	0.16862	0.23738	0.33168	0.37086	0.51028
2	0.07585	0.16858	0.23661	0.32842	0.36520	0.49984
5	0.07588	0.16864	0.23704	0.32938	0.36700	0.52675
10	0.07588	0.16864	0.23713	0.32951	0.36745	0.55547
20	0.07588	0.16864	0.23714	0.32952	0.36748	0.56368
50	0.07588	0.16864	0.23714	0.32952	0.36748	0.56678
	CDC15					
1	0.06420	0.15783	0.22104	0.31270	0.35536	0.51542
2	0.06374	0.15700	0.21827	0.30564	0.34639	0.49566
5	0.06370	0.15689	0.21771	0.30640	0.34725	0.52152
10	0.06370	0.15689	0.21767	0.30644	0.34780	0.55042
20	0.06370	0.15689	0.21767	0.30638	0.34780	0.55335
50	0.06370	0.15689	0.21767	0.30637	0.34779	0.56756
	Alpha-Factor					
1	0.04435	0.11903	0.19110	0.29814	0.34277	0.50708
2	0.04814	0.11370	0.18466	0.29244	0.33753	0.50076
5	0.04720	0.11530	0.18688	0.29501	0.34183	0.50746
10	0.04750	0.11553	0.18666	0.29532	0.33986	0.51684
20	0.04718	0.11592	0.18554	0.29375	0.33845	0.52423
50	0.04699	0.11596	0.18471	0.29380	0.33879	0.53709
	Elutriation					
1	0.05573	0.11865	0.17985	0.27823	0.31824	0.48531
2	0.05529	0.11899	0.17841	0.27091	0.30842	0.46726
5	0.05528	0.12042	0.17588	0.27105	0.31368	0.47391
10	0.05528	0.11941	0.17756	0.27294	0.31711	0.47882
20	0.05528	0.11949	0.17623	0.27251	0.31790	0.47941
50	0.05528	0.11923	0.17573	0.27300	0.31804	0.48761

4 Results and Discussion

As we mentioned before, we took the performance measures (in RMSE) at different iteration counts for increasing missing value levels. The findings are reported in the Table 2. Because EMMA is based on an iterative approach, we observed the change of accuracy with the number of iterations the underlying updater Hi is called upon for imputation. For almost all cases, it seems that RMSE is higher at the very first turn, and lessens within a few iterations (T = ~2-5). These values remain fairly steady throughout higher iterations (T >10). This happens due to the property of the regressor we used. Basically, linear regressor here tends to fit the data points within first few runs, and adjusted regressor does not rectify too much in the following turns. Moreover, the RMSE values across different missing rates remain

relatively robust, and do not swing erratically with the proportion of missing values induced in the datasets. For example, the RMSE counts for Diauxic dataset (T = 10) are 0.05590, 0.15112, 0.12307, 0.17679, 0.25288, 0.28897, 0.45119 at the missing rate of 1%, 5%, 10%, 20%, 25%, 50% respectively. As expected, as the missing rate increases, the performance of our technique deteriorates (see Table 2), but the degree of imputation accuracy does not fall as much as the magnitude of missing rates. This suggests that our algorithm can handle higher number of missing values efficiently.

To evaluate the performance of our method comparing to other extant methods, we obtained the RMSE numbers of those methods on the same datasets. As reported by Hu et al.[16], for Diauxic, Adaptive and Phosphate dataset, the RMSE measures (10% missing rate) of KNNimpute [11] and LLS based method [14] are ~0.6-0.8; while their integrative approach improved those numbers by decreasing almost ~0.05. On the other hand, the error numbers for our algorithm at the same settings are in the range of ~0.17-0.24 (T = 10), which is an improvement over the aforementioned methods by a significant margin. Besides, we maintain a competitive edge against these methods at higher missing rates (See Figure 2~4).

For a dataset with m rows (genes), n columns (samples) and k iterations, the computational complexity of our algorithm is approximately O(mn2k). So the running time scales up with the dimensions of the data matrix for the microarray. Instead of using linear regressor, we can also use any other hypothesis that can handle numeric class values (e.g. decision stump, multi-layer perceptrons, etc.).

All the datasets we used here were originated from same experiments using common microarray platform (spotted cDNA microarray). One of the advantages of our method is we do not need to integrate expression profiles from different experimental settings (cDNA vs. Affymetrix). Still, the performance of our algorithm depends on the variance of the datasets. Also, to infer linear regression, the dataset has to contain enough attributes (columns) to fit on the hypothesis.

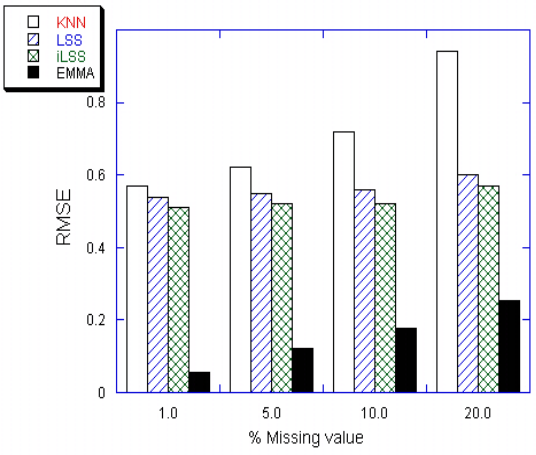

Figure 2: Comparison of performances for KNN, LSS, iLSS and EMMA for Diauxic dataset

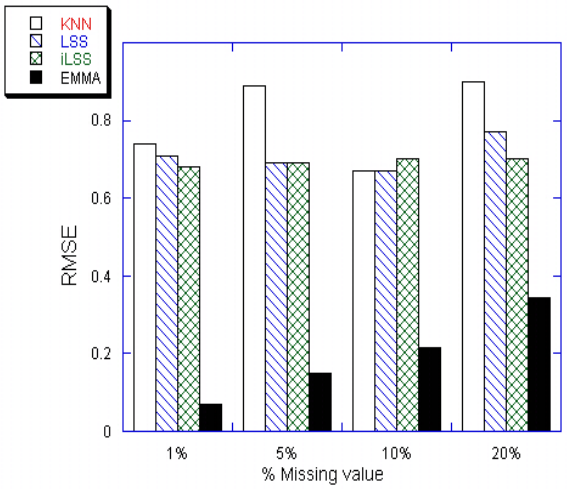

Figure 3: Comparison of performances for KNN, LSS, iLSS and EMMA for Adaptive dataset

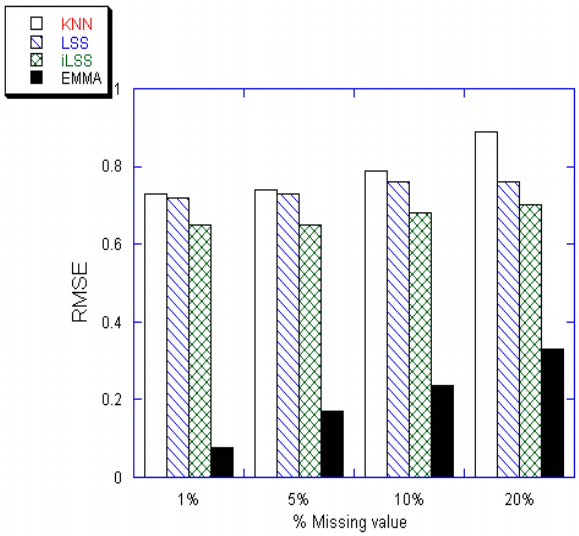

Figure 4: Comparison of performances for KNN, LSS, iLSS and EMMA for Phosphate dataset

5 Conclusions

To summarize, we present a novel method for treating missing sample values of genes in microarray data. Our technique is based on popular EM algorithm, and it far surpasses other existing state-of-the-art techniques in terms of imputation accuracy. In fact, being iterative in nature, our algorithm successfully grasps the innate relationships among the samples in subsequent runs, and stabilizes after the imputed and known values converge at some point. We

validated the strength of our algorithm by applying it to estimate missing values in both temporal and non-temporal benchmark datasets. In future, we expect to extend this technique to handle noise in microarray data in the preprocessing step.

6 References

[1] D. B. Allison, *et al.*, "Microarray data analysis: from disarray to consolidation and consensus," *Nat Rev Genet,* vol. 7, pp. 55-65, Jan 2006.

[2] A. G. de Brevern, *et al.*, "Influence of microarrays experiments missing values on the stability of gene groups by hierarchical clustering," *BMC Bioinformatics,* vol. 5, p. 114, Aug 23 2004.

[3] S. Raychaudhuri, *et al.*, "Principal components analysis to summarize microarray experiments: application to sporulation time series," *Pac Symp Biocomput,* pp. 455-66, 2000.

[4] V. Vapnik, "The Nature of Statistical Learning Theory," vol. Springer-Verlag, New York, 1995.

[5] M. P. Brown, *et al.*, "Knowledge-based analysis of microarray gene expression data by using support vector machines," *Proc Natl Acad Sci U S A,* vol. 97, pp. 262-7, Jan 4 2000.

[6] G. Valentini, "Gene expression data analysis of human lymphoma using support vector machines and output coding ensembles," *Artif Intell Med,* vol. 26, pp. 281-304, Nov 2002.

[7] O. Alter, *et al.*, "Singular value decomposition for genome-wide expression data processing and modeling," *Proc Natl Acad Sci U S A,* vol. 97, pp. 10101-6, Aug 29 2000.

[8] J. Huang, *et al.*, "Clustering gene expression pattern and extracting relationship in gene network based on artificial neural networks," *J Biosci Bioeng,* vol. 96, pp. 421-8, 2003.

[9] M. B. Eisen, *et al.*, "Cluster analysis and display of genome-wide expression patterns," *Proc Natl Acad Sci U S A,* vol. 95, pp. 14863-8, Dec 8 1998.

[10] A. A. Alizadeh, *et al.*, "Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling," *Nature,* vol. 403, pp. 503-11, Feb 3 2000.

[11] O. Troyanskaya, *et al.*, "Missing value estimation methods for DNA microarrays," *Bioinformatics,* vol. 17, pp. 520-5, Jun 2001.

[12] S. Oba, *et al.*, "A Bayesian missing value estimation method for gene expression profile data," *Bioinformatics,* vol. 19, pp. 2088-96, Nov 1 2003.

[13] T. H. Bo, *et al.*, "LSimpute: accurate estimation of missing values in microarray data with least squares methods," *Nucleic Acids Res,* vol. 32, p. e34, 2004.

[14] H. Kim, *et al.*, "Missing value estimation for DNA microarray gene expression data: local least squares imputation," *Bioinformatics,* vol. 21, pp. 187-98, Jan 15 2005.

[15] J. Tuikkala, *et al.*, "Improving missing value estimation in microarray data with gene ontology," *Bioinformatics,* vol. 22, pp. 566-72, Mar 1 2006.

[16] J. Hu, *et al.*, "Integrative missing value estimation for microarray data," *BMC Bioinformatics,* vol. 7, p. 449, 2006.

[17] J. L. DeRisi, *et al.*, "Exploring the metabolic and genetic control of gene expression on a genomic scale," *Science,* vol. 278, pp. 680-6, Oct 24 1997.

[18] T. L. Ferea, *et al.*, "Systematic changes in gene expression patterns following adaptive evolution in yeast," *Proc Natl Acad Sci U S A,* vol. 96, pp. 9721-6, Aug 17 1999.

[19] N. Ogawa, *et al.*, "New components of a system for phosphate accumulation and polyphosphate metabolism in Saccharomyces cerevisiae revealed by genomic expression analysis," *Mol Biol Cell,* vol. 11, pp. 4309-21, Dec 2000.

[20] P. T. Spellman, *et al.*, "Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization," *Mol Biol Cell,* vol. 9, pp. 3273-97, Dec 1998.

[21] A. P. Dempster, *et al.*, "Maximum-likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society,* vol. B39, pp. 1-38, 1977.

[22] M. Ouyang, *et al.*, "Gaussian mixture clustering and imputation of microarray data," *Bioinformatics,* vol. 20, pp. 917-23, Apr 12 2004.

[23] E. Frank, *et al.*, "Data mining in bioinformatics using Weka," *Bioinformatics,* vol. 20, pp. 2479-81, Oct 12 2004.

[24] R. Jornsten, *et al.*, "DNA microarray data imputation and significance analysis of differential expression," *Bioinformatics,* vol. 21, pp. 4155-61, Nov 15 2005.

[25] R. Jornsten, *et al.*, "A meta-data based method for DNA microarray imputation," *BMC Bioinformatics,* vol. 8, p. 109, 2007.