Missing Value Estimation in DNA Microarrays Using B-Splines

Sujay Saha
Heritage Institute of Technology, Kolkata, India
Email: sujay.saha@heritageit.edu

Kashi Nath Dey
University of Calcutta, Kolkata, India
Email: kndey55@gmail.com

Riddhiman Dasgupta, Anirban Ghose, and Koustav Mullick
Heritage Institute of Technology, Kolkata, India
Email: {riddhiman.dasgupta, anighose25}@gmail.com, koustav.mullick@yahoo.com

***Abstract*—Gene expression profiles generated by the high-throughput microarray experiments are usually in the form of large matrices with high dimensionality. Unfortunately, microarray experiments can generate data sets with multiple missing values, which significantly affect the performance of subsequent statistical analysis and machine learning algorithms. Numerous imputation algorithms have been proposed to estimate the missing values. However, most of these algorithms fail to take into account the fact that gene expressions are continuous time series, and deal with gene expression profiles in terms of discrete data. In this paper, we present a new approach, FDVSplineImpute, for time series gene expression analysis that permits the estimation of missing observations using B-splines of similar genes from fuzzy difference vectors. We have used smoothing splines to relax the fit of the splines so that they are less prone to over fitting the data. Our algorithm shows significant improvement over the current state-of-the-art methods in use.**

***Index Terms*—missing value estimation, DNA microarray, fuzzy logic, B-Spline, FDVSplineImpute**

I. Introduction

A gene expression microarray is a collection of microscopic DNA spots attached to a solid surface, which is used to study the expression levels of thousands of genes under various conditions simultaneously. Gene expression microarray experiments generate datasets of massive order which are in the form of matrices of gene expression levels under various experimental conditions. Each row of a gene expression matrix is basically a gene of the organism used in the experiment, while each column refers to a particular experimental condition under which the corresponding gene was examined. But biological experiments tend to generate gene expression matrices that contain missing values. These missing values occur due to errors in the experimental process that lead to corruption or absence of expression measurements. Various statistical methods used for gene expression analysis requires the complete gene expression matrix for providing accurate results. Methods such as hierarchical clustering, K-Means clustering are not robust to missing values. Hence, it is necessary to devise proper and accurate methods which impute data values when they are missing.

Time series data are a sequence of data points sampled at regular intervals of time. Gene expression time series data is a special class of microarray data where gene expression levels are sampled at regular intervals of time. Data sets measuring temporal behavior of thousands of genes offer rich opportunities for computational biologists [1]. A time-series gene expression data set is very sparse in nature as it contains a handful of data points. So a very accurate prediction method must be used for estimation.

II. Splines

A spline curve is a sequence of curve segments that are connected together to form a single continuous curve. They are basically piecewise polynomials with boundary, continuity and smoothness constraints. The use of piecewise low-degree polynomials result in smooth curves, thereby avoiding the problems of over fitting which would occur if only one high degree polynomial had been used for estimation. One can write a cubic polynomial in terms of a set of four normalized basis functions. A very popular basis is the B-spline basis. For the application of fitting curves to gene expression time-series data, it is quite convenient with the B-spline basis to obtain approximating or smoothing splines rather than interpolating splines. Smoothing splines use fewer basis coefficients than there are observed data points, which is helpful in avoiding over fitting. In this regard, the coefficients C_i can be interpreted geometrically as control

Manuscript received December 13, 2012; revised February 5, 2013

doi: 10.12720/jomb.2.2.88-92

points. It can be shown that the curve lies entirely within the convex hull of this controlling polygon. Further, each vertex exerts only a local influence on the curve, and by varying the vector of control points and another vector of knot points (discussed below), one can easily change continuity and other properties of the curve.

The normalized B-spline basis can be calculated using the Cox-deBoor recursion formula [1]:

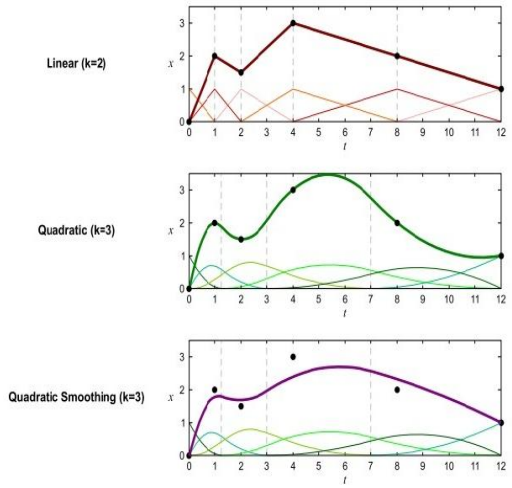

Figure 1. Figures of B-splines of various orders and types.

$$b_{i,1}(t) = \begin{cases} 1, & x_i \le t < x_{i+1} \\ 0 & otherwise \end{cases} \quad (1)$$

$$b_{i,k}(t) = \frac{(t - x_i) b_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t) b_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}} \quad (2)$$

Here k is the order of the basis polynomial (i.e. for a cubic polynomial k=4).

The values x_i are called *knots*, where $i = 1 \ldots n + k$. A *uniform* knot vector is one in which the entries are evenly spaced, i.e., $\mathbf{x} = (0, 1, 2, 3, 4, 5, 6, 7)^T$. If a uniform knot vector is used, the resulting B-spline is called periodic. For a periodic cubic B-spline ($k = 4$), the equation specifying the curve can be written as:

$$y(t) = \sum_{i=1}^{n} C_i b_{i,4}(t) \; for \; x_4 \le t \le x_{n+1} \quad (3)$$

In order to obtain a continuous time series, we use cubic B-splines to represent gene expression curves. By knowing the value of the splines at a set of control points in the time-series, one can generate the entire set of polynomials from the basis functions. Once the spline polynomials are generated we can re-sample the curve to estimate expression values at any time-points. We fit splines to representative similar genes of the target gene based on certain heuristics.

III. Literature Review

Gene expression microarray experiments can generate data sets with multiple missing expression values. Unfortunately, many algorithms for gene expression analysis require a complete matrix of gene array values as input. For example, methods such as hierarchical clustering and K-means clustering are not robust to missing data, and may lose effectiveness even with a few missing values. Methods for imputing missing data are needed, therefore, to minimize the effect of incomplete data sets on analyses, and to increase the range of data sets to which these algorithms can be applied. Rest of this section briefly describes some of those widely used, existing methods for estimation of the missing values from DNA microarray.

The earliest method, named as Row averaging or filling with zeroes, used to fill in the gaps for the missing values in gene data set with zeroes or with the row average

Troyanskaya et al. [2] proposed KNNImpute method to select genes with expression profiles similar to the gene of interest to impute missing values. After experimenting with a number of metrics to calculate the gene similarity, such as Pearson correlation, Euclidian distance, variance minimization, it was found that Euclidian distance was a sufficiently accurate norm.

The SVD Impute method, proposed by Troyanskaya et al. [2] uses Singular Value Decomposition of matrices to estimate the missing values of a DNA micro array. This method works by decomposing the Gene data matrix into a set of mutually orthogonal expression patterns that can be linearly combined to approximate the expression of all genes in the data set. These patterns, which in this case are identical to the principle components of the gene expression matrix, are further referred to as eigengenes. [3], [4]

Golub et al. [5] proposes another method named as LLSImpute that represents a target gene with missing values as a linear combination of similar genes. The similar genes are chosen by k-nearest neighbors or k coherent genes that have large absolute values of correlation coefficients followed by least square regression and estimation.

BPCAImpute method, proposed by Oba et al. [6] uses a Bayesian estimation algorithm to predict missing values. BPCA suggests using the number of samples minus 1 as the number of principal axes. Since BPCA uses an EM-like repetitive algorithm to estimate missing values, it needs intensive computations to impute missing values.

Ziv Bar-Joseph et al. [1] presents algorithms for time-series gene expression analysis that permit the principled estimation of unobserved time-points, clustering, and dataset alignment. Each expression profile is modeled as a cubic spline (piecewise polynomial) that is estimated from the observed data and every time point influences the overall smooth expression curve. The alignment algorithm uses the same spline representation of expression profiles to continuously time warp series.

FDVImpute method, proposed by S. Chakraborty, S. Saha, K. N. Dey [7], incorporates some fuzziness to estimate the missing value of a DNA microarray. The first step selects nearest (most similar) genes of the target gene (whose some component is missing) using Fuzzy Difference Vector algorithm. Then the missing cell is

estimated by using least square fit on the selected genes in the second step.

IV. Proposed Work

In all existing methods for estimation of missing values in gene expression microarrays. In all existing methods for estimation of missing values in gene expression microarrays, similar genes are selected on the basis of distance or correlation based metrics. These methods are, however, quite rigid, and do not consider at all the time series nature of the gene expression profiles. Thus we aim to incorporate a bit of fuzziness in selecting the similar genes so that the inherent properties of the gene expression time series are taken into account. We then use statistical spline estimation to represent time-series gene expression profiles as continuous curves. Although our method uses spline curves to represent gene expression profiles, it is not reasonable to fit each gene with an individual spline due to the large dimensionality of microarray datasets. Instead, we use a fuzzy difference vector to constrain genes which are similar to the same class, and then fit splines to each such class to estimate missing values much more accurately.

Our proposed algorithm, FDVSplineImpute, essentially consists of two distinct steps. In the first step, we select the most similar genes to the target gene using the fuzzy difference vector (FDV) algorithm with a suitable membership threshold. In the second step, a representative gene is chosen from the set of k similar genes, and a smoothing spline is fitted to the representative gene to reconstruct the target gene and estimate the missing value.

Let us consider a gene expression microarray dataset *Data* of m genes $Data_1, Data_2 \ldots, Data_m$ each of which has n observations. We consider that the row with the missing value has been shifted to the top, and the column with the missing value has been removed. We also consider *Time* to be the set of observation time points. So, all following computations are with respect to the first row as the target row, and with $n-1$ columns in the matrix. Now, the difference vector V_i of the i^{th} gene y_i is calculated as follows:

$$Difference\,Table_{i,k} = y_i(k) - y_i(k+1), \quad 1 \le k \le n-1 \qquad (4)$$

We take $Membership_i$ to contain the number of match between difference vectors $Difference\,Table_i$ and $Difference\,Table_j$ of the respective genes. A match in the k^{th} component of the vectors $Difference\,Table_i$ and $Difference\,Table_j$ is determined by whether the sign of $Difference\,Table_{i,k}$ and $Difference\,Table_{j,k}$ is same or not. In this way we calculate the total number of matches between the i^{th} and i^{th} genes, that m_{ij} contain. Since it is a fuzzy approach, we define a membership function $MembershipAverage_i$ for the i^{th} gene as follows:

$$MembershipAverage_i = \frac{m_i}{(n-1)} \qquad (5)$$

If the $MembershipAverage_i$ is greater than the predefined threshold θ then the corresponding i^{Th} gene is selected as a similar gene to the target gene. It is important to note that θ is a heuristic metric here. The set of similar genes is then taken, and a column wise mean gene, *Representative* is formed from this set of similar genes. With this *Representative* as the set of data points, and the set *Time* of observation time points of the microarray dataset as the knot vector, we can construct a smoothing B-spline using the deBoor recursion, with a tolerance of τ**.** The tolerance here specifies the maximum allowed deviation of the spline from the given data points. A tolerance of 0 will give us an interpolating spline, whereas increasing the tolerance will gradually give us a smoothing spline, then eventually a least squares fit, and finally a straight line fit.

The complete algorithm pseudo code for the proposed work is as follows:

```
FDVSplineImpute(Data, Row, Column,Time, θ, τ)
{
Create ProcessedData from Data
Shift target row, Data_Row, to the top in ProcessedData
Remove target column, Data_Column, from ProcessedData
For i = 1 to m do
    For j = 1 to Time.length-1 do
        DifferenceTable_i,j = ProcessedData_i,j-
    ProcessedData_i,j+1
    End
End

For i = 2 to m do
    Membership_i = 0
    For j = 1 to Time.length-1
        If ( DifferenceTable_i,j * DifferenceTable_i,j > 0 )
          Membership_i ++
      End If
  End
End

k=0
For i = 2 to m do
    MembershipAverage_i = Membership_i /(Time.length-1)
    If MembershipAverage_i >= Θ Then
        k = k+1
        For j = 1 to Time.length-1 do
          SimilarGenes_k,j =ProcessedData_i,j
        End
    End If
End

For i = 1 to Time.length-1 do
    Representative_i = 0
    For j = 1 to k do
        Representative_i = Representative_i + SimilarGenes_i,j
    End
    Representative_i = Representative_i / (Time.length-1)
End

GeneSpline = SmoothingSpline (Representative, Time, τ)
MissingValueEstimate = GeneSpline_Row,Column
```

}

V. Experimental Results

We have applied the proposed FDVSplineImpute method on the yeast cell cycle time series dataset from Spellman et al. The dataset consisted of a total of 6178 genes and 82 experiments performed on each gene. The dataset was pre-processed by decomposing the complete gene expression matrix into four specific time series gene expression matrices – alpha, cdc15, cdc28, elu and then removing the rows which contained missing values. We shall not use cdc28 in our analysis as it contains very few rows after pre-processing. The following table lists the characteristics of the dataset used for estimation.

TABLE I. Summary of Gene Expresion Time Series Analysed

Dataset	Start	End	Sampling	Complete Genes	Percentage of Genes used
alpha	0m	119m	Every 7m	4489	72.66 %
cdc15	10m	290m	Every 20m for 1 hr., 10 m for 3hr, 20 m for final hr.	4381	70.91 %
elu	0m	390m	Every 30m	5766	93.33%

We considered different values of θ as the threshold value for the fuzzy membership function based on the difference vectors. We considered two random genes in the alpha and elu datasets, and computed the number of similar genes found using values of θ ranging from 0.1 to 0.9. The results obtained showed us that the optimal value for θ is around 0.5, and all further testing has been done with this value.

After pre-processing the complete dataset, we chose random gene expression levels from each experiment and applied our proposed method. The following table shows a comparative study between two methods: the current state of the art, FDV-LLS and our proposed method FDV-Spline. We compute the error as e=|Estimated Value – Original Value|. We then compute the root mean square error and the normalized root mean square error for each time series experiment.

TABLE II. Results of Varying Θ for Alpha and Elu Datasets

		Alpha	Elu
Alpha Gene=2199 Elu Gene=2830	θ	Similar Genes	Similar Genes
	.1	4483	5660
	.3	4011	4938
	.5	2205	3395
	.7	361	905
	.9	10	133

TABLE III. Comparative Study of Gene Expression Level Estimation between and Fdvllsimpute and Fdvsplineimpute

Experiment	Row	Column	Missing Value	FDV-LLS Result				FDV-Spline Result			
				Estimate	Error	RMSE	N-RMSE	Estimate	Error	RMSE	N-RMSE
ALPHA	1	9	0.190	0.200	0.010	0.096	0.626	0.195	0.005	0.063	0.411
	33	15	- 0.240	- 0.060	0.180			-0.110	0.130		
	72	15	-0.040	-0.017	0.023			-0.060	0.020		
	163	8	0.150	0.167	0.017			0.164	0.014		
	511	9	0.040	0.155	0.115			0.035	0.050		
CDC15	1	7	-0.280	-0.284	0.004	0.258	0.890	-0.281	0.001	0.127	0.438
	301	13	-0.180	0.016	0.196			-0.200	0.020		
	302	17	0.220	0.640	0.420			0.360	0.140		
	537	10	-0.260	-0.027	0.233			-0.160	0.100		
	2345	20	-0.690	-0.942	0.252			-0.465	0.225		
ELU	2	5	0.030	0.008	0.022	0.044	1.329	0.040	0.010	.019	0.576
	150	8	0.000	-0.015	0.015			0.001	0.001		
	296	6	0.050	0.071	0.021			0.066	0.016		
	1381	12	0.100	0.170	0.070			0.063	0.037		
	5432	10	0.030	0.090	0.060			0.020	0.010		

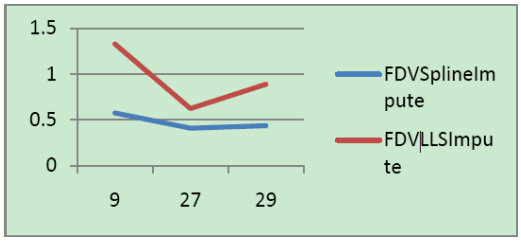

Figure 2. Plot of NRMSE of FDVSplineImpute and FDVLLSImpute against increasing percentage of missing values

VI. Conclusion

Our results show that FDVSplineImpute performs significantly better than the current state-of-the-art, FDVLLSImpute, by at least 30%. Moreover, since FDVLLSImpute is more accurate than the widely used kNNimpute and SVDImpute, we can say that FDVSplineImpute is more accurate than most widely used imputation algorithms. It is also robust to increasing percentages of missing values and is guaranteed to work well with both high and low dimensionalities. This is because only a few sets of points are sufficient to generate splines, and with a high number of points, smoothing splines ensure that there is no over fitting.

Spline representations of continuous time series gene expression data can be used for clustering, alignment of temporal data and dynamic time warping.

A fuzzy threshold to a heuristic membership function based on a difference vector has been used to determine similar genes, which have in turn been fitted by suitable smoothing splines. Our model has shown that factoring in the inherent time series nature of gene expression profiles leads to improved accuracy in estimation of missing values, and using nearest neighbor techniques in conjunction with spline representations leads to even higher accuracy and reduced over fitting.

References

[1] Ziv Bar-joseph, G. K. Gerber, D. K. Gifford, T. S. Jaakkola, and Itamar Simon, "Continuous Representations of Time-Series Gene Expression Data," *Journal of Computational Biology*, 2003.

[2] O. Troyanskaya, M. Cantor, G. Sherlock, *et al.*, "Missing value estimation methods for DNA microarray," *Bioinformatics*, vol. 17, pp. 520–525, 2001.

[3] O. Alter, P. O. Brown, and D. Botstein "Singular value decomposition for genome-wide expression data processing and Modelling," in *Proc. Natl. Acad. Sci*, USA, 2000, vol. 97, pp. 10101–10106.

[4] Golub and V. Loan, "Matrix Computations," 3rd edn. Johns Hopkins University Press, Baltimore, CA, 1996.

[5] H. Kim, G. H. Golub, and H. Park, "Missing value estimation for DNA microarray gene expression data: Local least squares imputation," *Bioinformatics*, vol. 21, pp. 187 – 198, 2005.

[6] S. Oba, M. Sato, I. Takemasa, *et al.*, "A Bayesian missing value estimation method for gene expression profile data," *Bioinformatics*, vol. 19, no. 16, pp. 2088-2096, Nov 1, 2003.

[7] S. Chakraborty, S. Saha, and K. Dey, "Missing value estimation in DNA microarray–A fuzzy approach," *IJAINN*, vol. 2, no. 1, 2012.

Prof. Sujay Saha received his M.Sc. & M.Tech degree in Computer Science from University of Calcutta in 2003 & 2005 respectively. He is currently an Assistant Professor at Computer Science Department of Heritage Institute of Technology, Kolkata, India, which he joined in 2005. His research interests include Soft Computing, Bioinformatics, etc.

Prof. Kashi Nath Dey received his M.Sc. degree in Applied Mathematics from University of Calcutta in 1978 and M.Tech in Computer Science from University of Calcutta in 1980. Presently he is an Associate Professor at Computer Science Department of University of Calcutta. His research interests include Soft Computing, Bioinformatics and Image Processing. Prof. Dey is an author of around 16 research publications.

Riddhiman Dasgupta is currently a final year student, pursuing his B.Tech in Computer Science and Engineering from Heritage Institute of Technology. His research interests include Bioinformatics, Machine Learning, and Computer Vision.

Anirban Ghose is currently a final year student, pursuing his B.Tech in Computer Science and Engineering from Heritage Institute of Technology. His research interests include Bioinformatics, Machine Learning and Computer Vision.

Koustav Mullick is currently a final year student, pursuing his B.Tech in Computer Science and Engineering from Heritage Institute of Technology. His research interests include Bioinformatics, Information Retrieval, and Computer Vision.