Multi-Algorithmic Approaches to Gene Expression Binarization

Jaime Seguel
Department of Electrical and Computer Engineering, University of Puerto Rico at Mayagüez, Mayagüez, Puerto Rico

Keywords: Gene Expression, Expression Threshold, Algorithm, Binarization, Aggregation.

Abstract: A basic problem in the construction of network representations of gene interactions is deciding whether a gene is or is not expressed at a time instant. This problem, referred here as the gene expression decision problem, has been approached with statistical and numerical algorithms. Numerical methods are based on different intuitions on what signals a gene expression threshold and as a consequence, they often return different answers. Consequently, the choice of a particular gene expression decision algorithm influences the gene interaction model. This article proposes an aggregation methodology for numerical gene expression decision algorithms that is based on voting. The result is thus, the expression decision made by the majority of the algorithms, provided that that decision is consistent with an underlying logical law referred as the doctrine. The proposed method is compared with some non-voting aggregation algorithms.

1 INTRODUCTION

Most of the physical and biochemical traits of a living being can be traced back to its genetic make-up through mRNA counts. An mRNA concentration is the result of intricate cascades of stochastic cellular processes that start with the transcription of information stored in the genes. For this reason, mRNA counts are referred as *gene expressions*. Although posttranscriptional events may alter the correlation between mRNA and their related proteins (Greenbaum et al., 2003) gene expression data still provide valuable insights on the transcriptional process in the cell. Transcription networks models such as Boolean and Probabilistic Boolean networks, are usually derived from, and validated with time series of gene expressions (Bornholdt, 2008); (Kim et al., 2013); (Shmulevich et al., 2010).

Gene expression data is normally obtained with DNA microarrays (Tarca et al., 2006), quantitative polymerase chain reactions (qPCR) (Derveaux et al., 2010), or next generation sequencing experiments (Matsumura et al., 2005); (Yamamoto et al., 2001). DNA microarray methods are based on hybridization of dyed mRNA samples to probes, and the measurements of the intensities of a fluorescent signal. The intensities are, in turn, correlated with the amount of mRNA in the sample through a complex protocol that involves a number of ad-hoc decisions on data analysis methods, background noise eliminations, and other error pruning considerations. Just as microarrays, qPCR methods are based on hybridizations and intensity measurements of fluorescent signals. But unlike microarrays, qPCR detection is made in real-time, with each cycle of amplification. Quantitative PCR is, in general, faster and more sensitive than microarrays, and requires lower amounts of material. Both, microarray and qPCR methods quantify only a selected number of transcripts. Next generation sequencing is capable of quantifying all the mRNA in a cell sample. The expression levels returned by these methods are basically free of correlation errors and background noise elimination, as they do not involve the transformation of signal intensities into estimations on the number of transcripts.

Gene expression changes with time and biological context. Thus, capturing meaningful information requires a sequence of experiments whose results are reported in a *gene expression array* (GEA). A GEA with N experiments on a set of M genes is a $M \times N$ array $G = [G(k, j)]$. Each row corresponds to a gene, and each column to a different experiment or *condition*. The *k-th* row in G is called *expression profile* of gene k.

The *gene expression decision problem* (GEDP) is stated as follows: *"For each entry G(k, j) in a GEA, decide whether gene k is or is not expressed at condition j"*.

It is worth remarking that GEDP is much harder to solve than the problem of detecting over

Seguel J..
Multi-Algorithmic Approaches to Gene Expression Binarization.
DOI: 10.5220/0005203701090115
In *Proceedings of the International Conference on Bioinformatics Models, Methods and Algorithms* (BIOINFORMATICS-2015), pages 109-115
ISBN: 978-989-758-070-3

expressed genes that commonly arises in the search for disease biomarkers. GEDP answers are presented, in turn, as a $M \times N$ array $B = [B(k, j)]$, where $B(k, j) = 1$ if the *k-th* gene is expressed at condition *j* and $B(k, j) = 0$, otherwise. Because of the currently limited knowledge of the cell inner mechanisms and the stochastic nature of the events that lead to gene expressions, matrix B is more a hypothesis on the states and transitions of the gene expressions than a deterministic fact. Nonetheless, these hypotheses are often formulated with the help of deterministic data analysis algorithms that mine each expression profile in *G* for signals of an expression threshold *t*. Once a threshold *t* is determined for the expression profile of gene *k*, the *k-th* row of *B* is produced by assigning 1 to the *j-th* entry if $G(k, j) > t$, and 0, otherwise.

Several algorithms based on different data mining methodologies and conjectures on the features in the data that signal an expression threshold, have been designed. Their results are often significantly different (Seguel et al., 2013).

In this article, I propose a wisdom-of-crowds methodology for aggregating these algorithmic decisions. The methodology is based on a mathematical structure that I call *multi-algorithm aggregation scheme* (MAS). MAS is inspired in the logic underlying collective decision-making by voting. MAS is a true alternative to average, median, and other common aggregation formulas, as it provides flexibility to select the voting method and a decision-making rule, referred as *doctrine*. This flexibility turns the method into an analytical tool; capable of testing the data with different decision-making parameters. As a mathematical structure, MAS can be used in applications other than gene expression decisions.

The rest of this article is organized as follows: Section 2 is a brief description of the algorithms selected for the proposed multi-algorithmic scheme, together with some basic time and space complexity analysis. Section 3 is a mathematical description of MAS and its implementation for solving the gene expression decision problem. Section 4 reports the results of experiments and comparisons between MAS and other aggregation rules, and Section 5 summarizes some conclusions of this work.

2 SOME GENE EXPRESSION DECISION ALGORITHMS

The gene expression decision algorithms that are the basis of the proposed multi-algorithm method can be classified in three main groups. The first group, referred as *jump-based* methods, consists of four algorithms that determine the threshold on the basis of a jump in the values of the gene expression profile. Methods in this group are labelled *J1, J2, J3* and *J4*. The second group consists of three algorithms that determine a threshold on the basis of approximations to the gene expression profile by one-step functions. These algorithms are denoted *S1, S2* and *S3* and are called *one-step* methods. The threshold returned by one-step methods is the midpoint of the steps in the one-step approximation mapping whose values are further apart.

The third group consists of two data clustering methods, both based on Lloyd's algorithm. These methods are labelled *C1* and *C2*. Next are high-level descriptions of each of these methods.

2.1 Jump-based Methods

Algorithm *J1* sorts the input expression profile in increasing order, and sets as threshold the midpoint between the smallest and the highest jump in the data. Algorithm *J2* is introduced in (Shmulevich et al., 2002) The method sorts the expression profile in increasing order and computes the average of all data jumps. Then, it sets as threshold the first value that exceeds the average. Algorithm *J3* is a variant of Algorithm *J2* that replaces the first value that exceeds the average data jump with the mean of all the values that exceed the average of the data jumps.

The main advantages of algorithms *J1, J2* and *J3* are conceptual and computational simplicity. In fact, they all return the *M* thresholds of a $M \times N$ array *G* in $O(MN)$ time, using $O(N)$ space. Algorithm *J4* is more complex. This method is an implementation of the Binarization Across Multiple Scales (BASC) algorithm (Hopfensitz et al., 2011). BASC approximates the input expression profile sorted in increasing order with a sequence of step functions, each with a different number of steps. It starts with the step function that fits exactly the input data. Then, it produces a sequence of step functions, each with one less step than the previous one. Dynamic programming is used to ensure that each new step function minimizes the Euclidian distance to the sorted expression profile. For each step function in the sequence, the ratio between the highest step jump and the Euclidean distance of the step function to the input data is computed. A high ratio is declared to be a strong discontinuity and its index is saved in a vector *v*. Then, the method computes the median *m* of the indices in *v* and defines the threshold as the average of the data point indexed by *m* and $m + 1$.

Algorithm *J4* returns the M thresholds of a $M \times N$ input array G in $O(MN^3)$ time, using $O(MN^2)$ space.

2.2 One-Step Approximations

The first method in this group, called Algorithm *S1* is inspired on StepMiner (Sahoo et al., 2007). StepMiner adjust a one-step or a two-step function to the data using linear regression with $(3, N - 1)$ degrees of freedom. The least square errors of the approximations provide a set of F-statistics, whose P-value is used for deciding whether the error is significant. StepMiner is not intended to solve GEDP. Algorithm S1 adjusts a one-step function to the expression profile sorted in increasing order, using StepMiner's methodology, and a preset significance of .05. If the subset of one-step mappings satisfying this constraint is empty, the method returns *Not a Number* (NaN). Otherwise, the method selects the step function whose steps are further apart, and sets the midpoint between the function's steps as the threshold.

Algorithm *S2* sorts the input vector in increasing order. Then, for each j from 1 to $N - 1$, computes the median of the data points from 1 to j and that the data points from $j + 1$ to N. Then, it finds the index m where the difference of the medians is maximal, and sets the threshold as the midpoint between the data points indexed by m and $m + 1$. Finally, Algorithm *S3* does the same as algorithm *S2* but using the mean instead of the median. All step methods return the M thresholds of a $M \times N$ input array G in $O(MN^2)$ time, using $O(MN)$ space.

2.3 Clustering Methods

Two methods are in this group. The first classifies a expression profile in two clusters using Lloyd's algorithm, also known k-means clustering. The algorithmic threshold is implicit, in the sense that the method splits the expression profile in two clusters, each centred around a different centroid; without computing a threshold. Algorithm *C1* sets as threshold the mid-point between the cluster's centroids.

Algorithm *C2* implements the iterative clustering variant of Lloyd's algorithm proposed in (Berestovsky et al., 2013) as a way to smooth data oscillations. *C2* starts with an application of the 2^d-means cluster algorithm to the input data. Here d is a user-defined parameter, whose sole restriction is that 2^d cannot be greater than the length of the expression profile. After computing the initial 2^d clusters, the algorithm replaces each element in a cluster with the cluster's mean, and applies the 2^{d-1}–means cluster algorithm to the resulting data. This process is repeated until d = 1.

As in Algorithm *C1*, the threshold returned by Algorithm *C2* is the mid-point between the centroids of the two clusters at the end of the iterations.

2.4 Threshold Correlations

In order to assess similarities and differences in the threshold values returned by the above algorithms, the thresholds of one thousand random 16-point vectors were computed for each algorithm. It was observed that the histograms for the threshold values presented significantly different shapes, and that the correlations among observed threshold values were very weak except in the cases of *S1* and *S3*, and *C1* and *C2* (Seguel et al., 2014). Scatter plots produced with this data confirmed that the threshold values returned by the rest of the algorithms do not have large correlations.

2.5 Threshold Displacements

In time-course data, it is natural to think of the expression profile as an N-point sample of a continuous gene expression function that takes values in a time interval. The size N of the sample may alter significantly the value of the expression threshold. This dependence of the threshold on N can be incorporated in a GEDP method through a statistical estimation of the threshold displacement as a function of N.

Table 1: Expected threshold displacements.

Algorithm	Expected displacement	Variance
J1	0.4609	0.0187
J2	0.7602	0.0516
J3	0.6709	0.0358
J4	0.2474	0.0069
S1	0.1034	0.0122
S2	0.3004	0.0359
S3	0.2777	0.0393
C1	0.1250	0.0001
C2	0.1552	0.0023

I call *threshold displacement* the maximum distance between the threshold computed with a sample of size $N = 2^n + 1$, $n > 2$; and the set of all the thresholds obtained by successively filtering each other data point until $n = 1$. The expected value of the threshold displacement for each of the nine algorithms was computed with four hundred random $(2^n + 1)$-point random vectors, with $n = 12$ for all methods except for *J4*. Because of space and time

limitations, the threshold displacement for algorithm *J4* was computed with (2^n + 1)-point random vectors, with $n = 8$. Table 1 reports the results.

Each expected displacement defines a threshold uncertainty interval. A point in this interval is declared to be *not decidable*. More precisely, if d is the expected displacement of an algorithm, and t is the threshold returned by the same algorithm on input V, then a point $G(k, j)$ in V is *not decidable* if

$$-d \leq G(k,j) - t \leq d. \quad (1)$$

Algorithms with lower expected threshold displacement will eventually decide a larger number of points in the expression profile.

3 MULTI-ALGORITHMIC SCHEMES

The core concepts in this section are borrowed from theories developed in the context of economics, jurisprudence and sociology (List, 2012). I define a multi algorithmic scheme (MAS) as a quadruple (*S, A, R, D*), where *S* is a finite set of *decision algorithms*, *A* is a finite set of logic statements called *agenda*, *R* is an aggregation rule, and *D* is a logical equivalence describing the fact to be determined in terms of the statements in the agenda. *D* is referred as the *doctrine*. Each algorithm in *S* decides whether each of the statements in *A* is true or false. The set of these decisions is called *algorithmic judgment*.

An aggregation rule is a method for determining a *collective judgmen*t from the set of all algorithmic judgments. Some common aggregation rules are majority, supermajority, unanimity and dictatorship. Under majority rule, the truth-value in the collective judgment is the truth-value of at least one half plus one of the algorithmic judgments. Under supermajority, the collective judgment is the truth-value of a preset number of algorithmic judgments that is greater than half plus one of the algorithms, and under unanimity, the truth-value of the collective judgment is to be shared by all algorithmic judgments. Dictatorship, in turn, imposes in the collective judgment the truth-value of a fixed, preselected algorithm. Thus, dictatorship is a degenerate or trivial aggregation rule.

A central concept in aggregation theory is *consistency*. In its simplest form, consistency refers to the preservation of the rules of logic when a doctrine is valuated with the truth-values of the collective judgments. The theory of aggregation devotes a significant effort to the search for conditions in the agenda under which non-trivial aggregation rules produce consistent judgments. In this work, however, no consistency requirement is imposed on the agenda. Instead, MAS interprets inconsistent collective judgments as instances of the GEDP that are *collectively not decidable*.

Not decidable and collectively not decidable points add a third option in the binary vector *B* that is denoted NaN (not a number). From the perspective of an answer to a GEDP, NaN entries in *B* are normally considered to be noisy data points and as such, are usually filtered out in subsequent applications of the GEDP solution.

3.1 A MAS for the GEDP

Let *S* be a subset of $\{J_1, J_2, J_3, J_4, S_1, S_2, S_3, C_1, C_2\}$, the set of gene expression decision algorithms. The previous discussion partitions the solution space of the GEDP into decidable and not decidable data points. Decidable points are further divided into points that indicate an expressed gene state and points that indicate that the gene is in unexpressed state. Let $A = \{U, N\}$, where

$$U =: "G(k,j) > t \wedge |t - G(k,j)| > d" \quad (2)$$

and

$$N =: "|t - G(k,j)| \leq d". \quad (3)$$

Clearly *U* is true if and only if $G(k, j)$ signals an unexpressed gene, and *N* is true if $G(k, j)$ is not decidable. The doctrine *D* is set to be

$$E \Leftrightarrow \sim U \wedge \sim N. \quad (4)$$

Thus, *E* is true if and only if $G(k, j)$ corresponds to an expressed gene. Finally, *R* may be majority, a super majority or the unanimity rule.

Table 2 illustrates a collective judgement that is inconsistent with doctrine *D*.

Table 2: Example of inconsistent collective judgment.

Algorithm	*N*	*U*	*E*
A	1	0	0
B	0	0	1
C	0	1	0
Majority	0	0	0

The algorithmic judgments of {*N, U*} are shown in rows A, B and C, together with the valuations of *E*. The forth row is the simple majority of votes on each predicate. According with the majority, both ~*N* and ~*U* are true while *E* is false. This is inconsistent with the doctrine as a true conjunction is true. a semantic rule of Consequently, the data point whose algorithmic and collective judgments are shown in Table 2 is not decidable.

4 SOME EXPERIMENTS

This section compares the resolution capabilities of MAS against non-voting aggregation formulas. The resolution is measured with a resolution score (RS), defined as

$$RS(V, X) = 1 - Z/N, \quad (5)$$

where Z is the number of points in an N-point expression profile V that are not decided by method X. The closer RS is to 1, the better the resolution of X. All algorithms were implemented in Matlab™.

4.1 MAS-GEDP Pseudo Code

Next is a high-level description of MAS-GEDP.

On input V (an expression profile)
1. For each algorithm in S
   Compute the threshold
2. For each data point in V
   For each algorithm in S
   i. Evaluate N, U and E
   ii. Apply majority rule R
   iii. Check consistency with D
   iv. If inconsistent or $N = 1$, write NaN in B
   v. Else if $E = 1$, write 1 in B
   vi. Else write 0 in B.
3. Output B

4.2 Non-voting Aggregation Formulas

I consider three non-voting aggregation formulas. The first two use the average (AVG), and the median (MED) of the thresholds, respectively. These formulas decide all points. The third aggregation formula decides that a point that is below the lowest threshold returned by the algorithms in S is unexpressed; a point that is above the highest threshold of all algorithms in S is expressed, and a point in between the lowest and highest thresholds is not decidable. I refer to this method as *below minimum and above maximum* (BMAM). Clearly, $0 \leq RS(V, X) \leq 1$ whenever X is a MAS-GEDP or BMAM.

Although BMAM is not a traditional aggregation rule, it is a natural and simple way to aggregate the algorithmic decisions.

4.3 Results

In this subsection I report the results returned by AVG_i, MED_i, $BMAM_i$, and MAS_i, I = 1 or 2. Here the label 1 indicates that the subset of algorithms is

$$S_1 = \{J4, S1, S3, C1, C2\}. \quad (6)$$

These are algorithms whose expected threshold displacement is less than 0.3. Methods labelled 2 use the nine gene expression threshold algorithms. MAS_1 uses simple majority while MAS_1^+ uses a supermajority of four or more votes. Similarly, MAS_2 uses simple majority while MAS_2^+ uses a supermajority of six or more votes.

Table 3: Synthetic expression profile 1.

V	0.080	0.029	0.160	0.960	0.858	0.808
AVG_1	0	0	0	1	1	1
AVG_2	0	0	0	1	1	1
MED_1	0	0	0	1	1	1
MED_2	0	0	0	1	1	1
$BMAM_1$	0	0	0	1	1	1
$BMAM_2$	0	0	NaN	1	NaN	NaN
MAS_1	0	0	0	1	1	1
MAS_1^+	0	0	0	1	1	1
MAS_2	0	0	0	1	1	1
MAS_2^+	0	0	0	1	1	1

Table 4: Synthetic expression profile 2.

V	0.452	0.402	0.502	0.622	0.770	0.809
AVG_1	0	0	0	1	1	1
AVG_2	0	0	0	1	1	1
MED_1	0	0	0	1	1	1
MED_2	0	0	0	1	1	1
$BMAM_1$	0	0	0	1	1	1
$BMAM_2$	0	0	0	NaN	NaN	1
MAS_1	NaN	0	NaN	NaN	1	1
MAS_1^+	NaN	0	NaN	NaN	NaN	1
MAS_2	NaN	NaN	NaN	NaN	NaN	NaN
MAS_2^+	NaN	NaN	NaN	NaN	NaN	NaN

Table 5: Synthetic expression profile 3.

V	0.143	0.279	0.459	0.654	0.813	0.906
AVG_1	0	0	0	1	1	1
AVG_2	0	0	0	1	1	1
MED_1	0	0	0	1	1	1
MED_2	0	0	0	1	1	1
$BMAM_1$	0	0	0	1	1	1
$BMAM_2$	0	0	0	1	1	1
MAS_1	0	0	NaN	NaN	1	1
MAS_1^+	0	0	NaN	NaN	1	1
MAS_2	0	NaN	NaN	NaN	NaN	1
MAS_2^+	0	NaN	NaN	NaN	NaN	1

The synthetic expression profile 1 approximates a one-step function with rather distant steps; synthetic expression profile 2 also approximates a one-step function but with closer steps. Finally, synthetic expression profile 3 approximates a straight line with slope 1.

4.4 Some Statistics

The expected value of the resolution score of

BMAM and MAS methods were computed with 400 randomly generated 16-point expression profiles. The expected resolution scores (5) and their variance are shown in Table 6.

Table 6: Expected resolution scores.

Method	Expected RS	Variance
$BMAM_1$	0.7104	0.0335
$BMAM_2$	0.4935	0.0345
MAS_1	0.7112	0.0160
MAS_1^+	0.5334	0.0188
MAS_2	0.4792	0.0226
MAS_2^+	0.2924	0.0204

The percentage of coincident decisions in the outputs of $BMAM_1$ and MAS_1, and those of $BMAM_2$ and MAS_2 were measured in the same experiment. The results are shown in Table 7.

Table 7: Percentage of coincidences $BMAM_1 - MAS_1$.

	MAS_1 (1)	MAS_1 (0)	MAS_1 (NaN)
$BMAM_1$ (1)	32.02		
$BMAM_1$ (0)		32.70	
$BMAM_1$(NaN)			20.90

Table 8: Percentage of coincidences $BMAM_2 - MAS_2$.

	MAS_2 (1)	MAS_2 (0)	MAS_2 (NaN)
$BMAM_2$ (1)	23.34		
$BMAM_2$ (0)		15.25	
$BMAM_2$(NaN)			40.44

According to the tables, $BMAM_1$ and MAS_1 coincide about 86% of times in their decisions, while $BMAM_2$ and MAS_2 coincide only about 79% of times.

5 CONCLUSIONS

Because of the stochastic nature of gene expression, formulating a hypothesis on the state of a gene at a particular time instant is not a deterministic problem. Nonetheless, deterministic algorithms based on intuitive models and different data mining methodologies provide insights on the gene expression state. Aggregating their solutions is a way around determinism. In this article I introduce MAS, an aggregation method that regards each deterministic answer as a vote and makes a decision on the basis of a majority rule. Points whose aggregated decision contradicts the doctrine, and points that fall within a threshold uncertainty interval, are declared to be not decidable and discarded as noisy data.

There is not a significant agreement between $BMAM_1$ and MAS_1 in the identification of noisy points. In general, as shown in Table 6, methods $BMAM_i$ and MAS_i, i = 1, 2; have comparable scores of resolution when simple majority is used.

ACKNOWLEDGEMENTS

This research was supported in part by grant NIH-MARC 5T36GM095335-02I.

REFERENCES

Greenbaum D, Colangelo C, Williams K, Gerstein M., 2003. Comparing protein abundance and mRNA expression levels on a genomic scale. In *Genome Biology* **4** (9): 117. doi:10.1186/gb-2003-4-9-117. PMC 193646. PMID 12952525.

Bornholdt. S., 2008. Boolean network models of cellular regulations: prospects and limitations. *J. R. Soc. Interface, 5*. Suppl. 1. 85–94.

Kim, Y., Han, S., Choi, S. and Hwang, D., 2013. Inference of dynamic networks using time-course data**.** *Briefings in Bioinformatics*. doi:10.1093/bib/bbt028.

Shmulevich, I., and E. Dougherty, E. Probabilistic Boolean networks: the modeling and control of gene regulatory networks. 2010. SIAM.

Tarca, A. L., Romero, R., and Draghici, S., 2006. Analysis of microarray experiments of gene expression profiling. *American Journal of Obstetrics and Gynaecology*.195(2), 373-388.

Derveaux, S., Vandesompele, J., and Hellemans, J., 2010. How to do successful gene expression analysis using real-time PCR. *Methods*, Vol 50, 227-230.

Matsumura H, Ito A, Saitoh H, Winter P, Kahl G, Reuter M, Krüger DH, Terauchi R., 2005. SuperSAGE. *Cell Microbiol* 7(1):11-18.

Yamamoto, M., Wakatsuki, T., Hada, A. and Ryo, A., 2001. Use of serial analysis of gene expressions (SAGE) technology. *Journal of Immunological Method*s 250:45-66.

Seguel, J., Lluberes, M., 2013., Semantics and accuracy of gene expression threshold algorithms: A case study. *Proc. ADVCOMP 2013*, Oporto, Portugal.

Shmulevich, I., Zhang, W., 2002. Binary analysis and optimization-based normalization of gene expression data. *Bioinformatics.* Vol. 18-4, 555–565.

Hopfensitz, M. et al., 2011., Multiscale binarization of gene expression data for reconstructing Boolean networks. *IEEE/ACM transactions on computational biology and bioinformatics.* Vol. 9, no. 2, 487–498.

Sahoo, D., Dill, D. L., Tibshirani, R., and Plevritis, S. K . 2007. Extracting binary signals from microarray time-course data.. *Nucleic Acids Research Advance Access*. doi:10.1093/nar/gkm284.

Berestovsky, N., Nakhleh. N., 2013. An evaluation of methods for inferring Boolean networks from time-

series data. *Plos One,* doi: 10.1371 /journal .pone. 0066031.

List, C., 2012. The theory of judgment aggregation: An introductory review. *Synthese* 187 (1), 209-221.

Seguel, J. Macchiavelli, R. 2014. Mathematical and statistical characterizations of expression threshold methods. *Unpublished manuscript*.