

## DNA Microarray Data Analysis

IIRIS HOVATTA, KATJA KIMPPA, ANTTI LEHMUSSOLA, TOMI PASANEN,  
JANNA SAARELA, ILANA SAARIKKO, JUHA SAHARINEN, PEKKA TIIKKAINEN  
TEEMU TOIVANEN, MARTTI TOLVANEN, MAUNO VIHINEN AND GARRY WONG  
EDITORS JARNO TUIMALA AND M. MINNA LAINE

CSC

CSC – Scientific Computing Ltd. is a non-profit organization for high-performance computing and networking in Finland. CSC is owned by the Ministry of Education. CSC runs a national large-scale facility for computational science and engineering and supports the university and research community. CSC is also responsible for the operations of the Finnish University and Research Network (Funet).

All rights reserved. The PDF version of this book or parts of it can be used in Finnish universities as course material, provided that this copyright notice is included. However, this publication may not be sold or included as part of other publications without permission of the publisher.

© The authors and  
CSC – Scientific Computing Ltd.  
2005

Second edition

ISBN 952-5520-11-0 (print)

ISBN 952-5520-12-9 (PDF)

<http://www.csc.fi/oppaat/siru/>

<http://www.csc.fi/molbio/arraybook/>

Printed at  
Picaset Oy  
Helsinki 2005

## Preface

This is the second, revised and slightly expanded edition of the DNA microarray data analysis guidebook. As a change to the previous edition, some relatively quickly changing material such as software tutorials have been exclusively published on the book's web site. Please see <http://www.csc.fi/molbio/arraybook/> for more information and to access the extra material.

DNA microarrays generate large amounts of numerical data, which should be analyzed effectively. In this book, we hope to offer a broad view of basic theory and techniques behind the DNA microarray data analysis. Our aim was not to be comprehensive, but rather to cover the basics, which are unlikely to change much over years. Especially, we hope that researchers starting their data analysis can benefit from the book.

The text emphasizes gene expression analysis. Topics, such as genotyping, are discussed shortly. This book does not cover the wet-lab practises, such as sample preparation or hybridization. Rather, we start when the microarrays have been scanned, and the resulting images are being analyzed. Also, we take the files with signal intensities, which usually generate questions such as: "How is the data normalized?" or "How do I identify the genes which are upregulated?", and provide some simple solutions to these specific questions and many others.

Each chapter has a section on suggested reading, which introduces some of the relevant literature. Some chapters have additional information available on the web. In the first edition the software examples were included in the book, but we have now moved them into Internet. This allows us to better keep the material up to date.

Juha Haataja and Leena Jukka are warmly acknowledged for their support during the production of this book.

We are very interested in receiving feedback about this publication. Especially, if you feel that some essential technique has been missed, let us know. Please send your comments to the e-mail address [Jarno.Tuimala@csc.fi](mailto:Jarno.Tuimala@csc.fi).

Espoo, 23rd December 2005

*The authors*

## List of Contributors

Iiris Hovatta  
National Public Health Institute  
Haartmaninkatu 8  
FI-00290 Helsinki  
Finland

Katja Kimppa  
PerkinElmer Life Sciences and Analytical Sciences  
- Wallac Oy  
P.O.Box 10  
FI-20101 Turku  
Finland

M. Minna Laine  
CSC, the Finnish IT center for science  
Keilaranta 14  
FI-02101 Espoo  
Finland

Antti Lehmussola  
Tampere University of Technology  
P.O.Box 553  
FI-33101 Tampere  
Finland

Tomi Pasanen  
University of Helsinki  
P.O.Box 68  
FI-00014 University of Helsinki  
Finland

Janna Saarela  
Biomedicum Biochip Center  
Haartmaninkatu 8  
FI-00290 Helsinki  
Finland

Ilana Saarikko  
University of Helsinki  
P.O.Box 68  
FI-00014 University of Helsinki  
Finland

Juha Saharinen  
National Public Health Institute  
Haartmaninkatu 8  
FI-00290 Helsinki  
Finland

Pekka Tiikkainen  
VTT  
P.O.Box 106  
FI-20521 Turku  
Finland

Teemu Toivanen  
Centre for Biotechnology  
Tykistökatu 6  
FI-20521 Turku  
Finland

Martti Tolvanen  
Institute of Medical Technology  
Biokatu 8  
FI-33520 Tampere  
Finland

Jarno Tuimala  
CSC, the Finnish IT center for science  
Keilaranta 14  
FI-02101 Espoo  
Finland

Mauno Vihinen  
Institute of Medical Technology  
Biokatu 8  
FI-33520 Tampere  
Finland

Garry Wong  
A. I. Virtanen -institute  
University of Kuopio  
FI-70211 Kuopio  
Finland

10.10	Examples . . . . .	113
10.11	Suggested reading . . . . .	113
<b>11</b>	<b>Finding differentially expressed genes</b>	<b>115</b>
11.1	Identifying over- and under-expressed genes . . . . .	115
11.1.1	Filtering by absolute expression change . . . . .	115
11.1.2	Statistical single chip methods . . . . .	116
11.1.3	Noise envelope . . . . .	116
11.1.4	Sapir and Churchill's single slide method . . . . .	116
11.1.5	Chen's single slide method . . . . .	117
11.1.6	Newton's single slide method . . . . .	118
11.2	What about the confidence? . . . . .	119
11.2.1	Only some treatments have replicates . . . . .	119
11.2.2	All the treatments have replicates: two-sample t-test . . . . .	120
11.2.3	All the treatments have replicates: one-sample t-test . . . . .	121
11.2.4	Non-parametric tests . . . . .	121
11.3	Examples . . . . .	122
11.4	Suggested reading . . . . .	122
<b>12</b>	<b>Cluster analysis of microarray information</b>	<b>123</b>
12.1	Basic concept of clustering . . . . .	123
12.2	Principles of clustering . . . . .	123
12.3	Hierarchical clustering . . . . .	124
12.4	Self-organizing map . . . . .	125
12.5	K-means clustering . . . . .	126
12.6	KNN classification - a simple supervised method . . . . .	127
12.7	Principal component analysis . . . . .	129
12.8	Pros and cons of clustering . . . . .	130
12.9	Visualization . . . . .	131
12.10	Function prediction . . . . .	133
12.11	Examples . . . . .	133
12.12	Suggested reading . . . . .	133
<b>III</b>	<b>Data mining</b>	<b>134</b>
<b>13</b>	<b>Gene regulatory networks</b>	<b>135</b>
13.1	What are gene regulatory networks? . . . . .	135
13.2	Fundamentals . . . . .	135
13.3	Bayesian network . . . . .	137
13.4	Calculating Bayesian network parameters . . . . .	139
13.5	Searching Bayesian network structure . . . . .	140
13.6	Conclusion . . . . .	141
13.7	Suggested reading . . . . .	142

---

<b>14</b>	<b>Data mining for promoter sequences</b>	<b>143</b>
14.1	General introduction	143
14.2	Introduction to gene regulation	143
14.3	Input data I: promoter region sequences	145
14.4	Using BioMart to retrieve promoter regions	149
14.4.1	Final warnings regarding upstream data	149
14.5	Input data II: the matrices for transcription factor binding sites	149
14.5.1	Searching known patterns	150
14.5.2	Pattern search without prior knowledge of what you search	151
14.6	Examples	152
14.7	Suggested reading	152
<b>15</b>	<b>Biological sequence annotations</b>	<b>154</b>
15.1	Annotations in gene expression studies	154
15.2	Using annotations with gene expression microarrays	154
15.3	Using Ensembl server to batch process of annotations	155
15.4	Biological pathways and GeneOntologies	156
15.5	Using GeneOntologies for gene expression data analysis	156
15.6	Concluding remarks	157
15.7	References	157
<b>16</b>	<b>Software issues</b>	<b>160</b>
16.1	Data format conversions problems	160
16.2	A standard file format	161
16.3	Programming	161
16.3.1	Perl	161
16.3.2	R	161
16.4	Examples	161
	<b>Index</b>	<b>162</b>

## **Part III**

# **Data mining**

# 13 Gene regulatory networks

Tomi Pasanen, and Mauno Vihinen

## 13.1 What are gene regulatory networks?

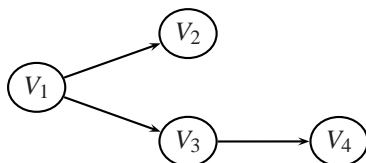
By combining gene expression analysis, perturbations or treatments, and mutations of genes we can study processes like signal transduction and metabolism yielding information on molecular effects or functions of specific genes. However, gene expression data permits us to go beyond the traditional line of research and to study finer structures of molecular pathways exposing causal regulation relations between genes. The finer structures do not only describe the nature of interactions between genes, inhibition or activation, but also exemplify direct and indirect effects of genes. For example, is gene  $A$  regulating gene  $B$  or vice versa, is the regulation direct or indirect where there is a mediating gene  $C$  (or maybe many) so that  $A$  regulates  $C$  and then  $C$  regulates  $B$ . Inspecting the finer structures, which are called regulatory networks, gives us a more intricate view of molecular interactions offering further possibilities for medical interventions.

Inferring regulatory networks from expression data, a process which is called reverse-engineering, is not a computationally simple problem because an enormous amount of time is needed even when a trivial approach is applied. Therefore, we sketch principles and methods with an example approach applicable for the inference process in next sections.

## 13.2 Fundamentals

It is customary to visualize regulatory dependencies between genes by a graph  $G(V, E)$  with a vertex set  $V$  consisting of the genes examined and an edge set  $E$ , see Figure 13.1. A directed edge  $E_{(i,j)}$  between vertices  $V_i$  and  $V_j$  represents regulation between genes attached to vertices  $V_i$  and  $V_j$ , and the direction of the edge, denoted by  $V_i \rightarrow V_j$ , reflects the order of regulation where gene  $V_i$  regulates gene  $V_j$  (up- or downregulation). Node  $V_i$  is called a *parent* of  $V_j$  and node  $V_j$  is called a *child* of  $V_i$ . To present logical structures of group regulations, which appear extensively in nature, hyper edges should be used, where an edge is adjacent to

several genes, but these are used quite rarely because of the convenience of simple drawings. A graph like  $G$  is called *gene regulatory network*.



**Figure 13.1:** Graph  $G$  consisting of four nodes  $V_1, V_2, V_3, V_4$ , and tree edges  $E_{(1,2)}, E_{(1,3)}$ , and  $E_{(3,4)}$ .

By inspecting a putative regulatory graph, like  $G$ , build from earlier experiments, we can pose several questions. First of all, is the structure right, or if there are several possible structures with respect to the experiments done so far, which one is right? So a graph represents hypotheses based on current knowledge. By locating central nodes or nodes of interest we can conduct additional experiments to confirm the dependencies around nodes or get an idea which of the graphs is the correct one. Choosing the best node to inspect more carefully is not trivial, at least not, when we try to make distinctions between putative graphs and we try to minimize experiment costs at the same time. To this category belongs also the question: is the order of regulation correct? Manual perturbation, like the deletion of a gene, might give a clear clue on the order of regulation. Finally, if we stick to the obtained structure there is still questions on the strength of each regulation, whether they are they right.

But before all the above questions, we need to know how to construct such a graph directly from data, because, in theory, any kind of an edge is possible between any set of vertices, and so the problem is to infer the true connections based on the data produced by the experiments. This approach is called *reverse-engineering of gene regulatory networks*. It is interesting to note that similar to biochemical reactions, sequences, and three dimensional structures having their own motifs, it is clear that regulatory networks have to have their own regulatory motifs or circuit motifs based on the lower level motifs.

To infer a regulatory network, several distinct expression samples of the genes of discourse are needed. In a time series analysis, the expression levels are recorded along fixed time points, while in a perturbation analysis, the expression levels are recorded separately for each manual perturbation (over/under, deletion) of specific genes. A fundamental difference between perturbation data and time series data is that perturbation allows a firm inferring order of regulation while time series data can only reveal the probable regulation direction. The problem can be overcome by combining a few manual perturbation experiments with time series data. In both cases, the genes are monitored at fixed points inducing an expression matrix

$$\{X_i[j] \mid 1 \leq i \leq n, 1 \leq j \leq m\},$$

where  $X_i[j]$  denotes the expression level  $X_i$  of gene or node  $V_i$  at point  $j$ , see Table 13.1. Based on the data, we can try to infer the most prominent regulatory

**Table 13.1:** Top, part of the table for time series expression data of genes SWI5, CLN2, CLN3, and CLB1 [1]. The mean of whole data was 0. Bottom, the expression levels are discretized so that values higher than or equal to the mean get a value of 1 and values lower than the mean get a value of 0.

gene	$t_1$	$t_2$	$t_3$	$t_4$	$t_5$	$t_6$	$t_7$	$t_8$	$t_9$
1 (SWI5)	-0.41	-0.97	-1.46	0.16	0.74	0.72	1	0.77	0.3
2 (CLN3)	0.49	0.62	0.05	-0.13	0.02	0.04	-0.14	0.24	0.91
3 (CLB1)	0.6	-0.53	-1.37	1.03	1.13	1.27	1.04	1	0.07
4 (CLN2)	-1.26	1.6	1.54	0.31	-0.14	-0.88	-1.7	-1.88	-1.7
1 (SWI5)	0	0	0	1	1	1	1	1	1
2 (CLN3)	1	1	1	0	1	1	0	1	1
3 (CLB1)	1	0	0	1	1	1	1	1	1
4 (CLN2)	0	1	1	1	0	0	0	0	0

subnetworks, or, if we know the structure of some subnetwork beforehand, we can try to model mathematically the regulations between nodes in the subnetwork.

Because the number of possible networks increases super-exponentially on the number of genes, prominent regulatory networks are usually searched for by using supervised approaches, where some intrinsic partial knowledge about regulations between genes are formulated by implicit or explicit rules. For the mathematical modeling of regulation inside a network, there are many approaches like Bayesian network, Boolean network and its generalization, ordinary and partial differential equations, qualitative differential equations, stochastic master equations, Petri nets, transform grammars, process algebra, and rule-based formalisms, to mention some.

We will use Bayesian network as a sample tool for several reasons: its framework offers a clear separation of structure and parameter optimization, and adding predefined rules and information is easy. Moreover, it is widely used for microarray data.

### 13.3 Bayesian network

Bayesian network modelling consists of two parts: the qualitative part, where direct (causal) influences between nodes  $V$  are depicted as directed edges  $E$  in graph  $G$ , and the quantitative part, where local conditional probability distributions of expression levels are attached to nodes  $V$  of  $G$ . Thus, expression levels  $X_i$  of nodes  $V_i$  are considered as random variables and the edges represent conditional dependencies between distributions of the random variables.

Let us consider a binary case where a gene can be “off”, denoted by 0, or “on”, denoted by 1, but not both (see the bottom values of Table 13.1). Having a fixed structure like graph  $G$  in Figure 13.1, the conditional probabilities are easy to calculate, see Table 13.2.

The benefit of this modeling is that we need to store only  $O(2^k n)$  parameters when each node has at most  $k$  parents, *i.e.*, one “success” probability for each row in each table just calculated.

**Table 13.2:** Conditional probabilities based on the structure of  $G$  and the bottom values of Table 13.1.

$\Pr(X_1 = 1)$	$X_1 \mid \Pr(X_2 = 1)$	$X_1 \mid \Pr(X_3 = 1)$	$X_3 \mid \Pr(X_4 = 1)$
6/9	1   4/6 0   1	1   1 0   1/3	1   1/7 0   1

This is clear advantage over  $O(2^n)$  parameters when the expression value of a node depends on the expression values of all other nodes. Thus, parameters pertain only to local interaction.

Underlying the above discussion there are some basic probability rules that are examined next. The joint probability for random variables  $X$  and  $Y$  (not necessarily independent) is calculated using the conditional probability:

$$\Pr(X \cap Y) = \Pr(X, Y) = \Pr(Y \mid X)\Pr(X) = \Pr(X \mid Y)\Pr(Y),$$

where notation  $\Pr(X \mid Y)$  denotes the probability of observation  $X$  after we have seen evidence  $Y$ , see Figure 13.2. Reorganizing the equation induces Bayes' rule

$$\Pr(X \mid Y) = \frac{\Pr(Y \mid X)\Pr(X)}{\Pr(Y)}$$

while generalization of it forms the chain rule

$$\begin{aligned} \Pr(K, X, Y, Z) &= \Pr(Z \mid K, X, Y)\Pr(K, X, Y) \\ &= \Pr(Z \mid K, X, Y)\Pr(Y \mid K, X)\Pr(K, X) \\ &= \Pr(Z \mid K, X, Y)\Pr(Y \mid K, X)\Pr(X \mid K)\Pr(K). \end{aligned}$$

If variables  $X$  and  $Y$  are independent, then

$$\Pr(X \cap Y) = \Pr(X)\Pr(Y),$$

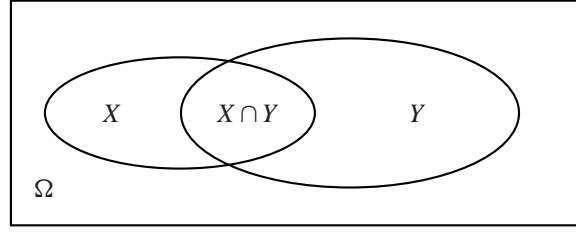
and if variables  $X$  and  $Y$  are independent for given a value of  $Z$ , then

$$\Pr(X \mid Z, Y) = \Pr(X \mid Z).$$

Based on our dependency graph  $G$ , probability  $\Pr(X_1, X_2, X_3, X_4)$  can be simplified as follows

$$\Pr(X_1, X_2, X_3, X_4) = \Pr(X_1)\Pr(X_2 \mid X_1)\Pr(X_3 \mid X_1)\Pr(X_4 \mid X_3).$$

For example,  $\Pr(X_4 \mid X_1, X_2, X_3) = \Pr(X_4 \mid X_3)$  because for given the value of  $X_3$ ,  $X_4$  is independent of the values of  $X_1$  and  $X_2$ . So, it is enough to consider only direct local dependencies from parents when evaluating the probabilities (for  $X_4$  the only parent is  $X_3$ ). We see that in Bayesian networks the probability calculations follow in a natural way the structure of a graph.



**Figure 13.2:** A Venn diagram of the probability space  $\Omega$ , where the areas denote probabilities, for example,  $\Pr(X)$ , is the left circle area divided by the total area of rectangle  $\Omega$ . It is easy to see that  $\Pr(X \cap Y) = \Pr(Y | X)\Pr(X)$ .

### 13.4 Calculating Bayesian network parameters

Having graph  $G$  and the expression matrix  $D = \{X_i[j]\}$ , our aim is to obtain distribution dependency parameters  $\theta = \{\theta_1, \theta_2, \dots\}$  that are fitted best to the structure of  $G$  and data  $D$ . This is done by using the maximal likelihood principle where we maximize the likelihood function  $L(\theta : D)$ :

$$L(\theta : D) = \Pr(D : \theta) = \prod_{j=1}^m \Pr(X_1[j], X_2[j], \dots, X_n[j] : \theta).$$

By using the structure of  $G$  (see again Figure 13.1) we can decompose our optimization problem to independent subproblems:

$$\begin{aligned} L(\theta : D) &= \prod_{j=1}^m \Pr(X_1[j] : \theta_1) \cdot \prod_{j=1}^m \Pr(X_2[j] | X_1[j] : \theta_2) \\ &\quad \cdot \prod_{j=1}^m \Pr(X_3[j] | X_1[j] : \theta_3) \cdot \prod_{j=1}^m \Pr(X_4[j] | X_3[j] : \theta_4), \end{aligned}$$

or, in general,

$$\prod_i L_i(\theta_i : D).$$

The network structure decomposes the parameter set  $\theta$  to independent parameters  $\theta_1, \theta_2, \theta_3$  and  $\theta_4$ , which are computationally fast to estimate.

For simplicity, we consider again the binary case, see bottom of Table 13.1. Since  $V_1$  has no parents, *i.e.*, it is independent of other nodes, its likelihood function  $L_1(\theta_1 : D)$  corresponds to the binomial probability where the probability of getting value 1 is  $\theta_1$  while  $1 - \theta_1$  is the probability of getting value 0. Thus,

$$L_1(\theta_1 : D) = \Pr(D : \theta_1) = \prod_{j=1}^m \Pr(X_i[j] : \theta_1) = (\theta_1)^{N(X_1=1)} (1 - \theta_1)^{m - N(X_1=1)},$$

where  $N(\cdot)$  denotes the number of each occurrence in data with respect to the requirements inside parentheses. The estimator  $\hat{\theta}_1$  for  $\theta_1$  maximizing the product is

simply the already calculated probability for  $\Pr(X_1 = 1)$ , *i.e.*,

$$\hat{\theta}_1 = \frac{N(X_1 = 1)}{N(X_1 = 1) + N(X_1 = 0)} = 6/9.$$

Similarly, the conditional estimators for parameters  $\theta_{(2|X_1)}$ ,  $\theta_{(3|X_1)}$  and  $\theta_{(4|X_3)}$  are the probabilities already calculated:

$$\begin{aligned} \hat{\theta}_{(2|X_1=1)} &= 4/6, & \hat{\theta}_{(2|X_1=0)} &= 1, \\ \hat{\theta}_{(3|X_1=1)} &= 1, & \hat{\theta}_{(3|X_1=0)} &= 1/3, \\ \hat{\theta}_{(4|X_3=1)} &= 1/7, & \text{and } \hat{\theta}_{(4|X_3=0)} &= 1. \end{aligned}$$

Together these estimators form  $\hat{\theta} = \{\hat{\theta}_1, \hat{\theta}_{(2|X_1)}, \hat{\theta}_{(3|X_1)}, \hat{\theta}_{(4|X_3)}\}$ , which is the best explanation for the observed data  $D$  restricted to the structure of  $G$ , and we can compute

$$\begin{aligned} L(\theta : D) &= (\hat{\theta}_1)^6 \cdot (1 - \hat{\theta}_1)^3 \cdot (\hat{\theta}_{(2|X_1=1)})^4 \cdot (1 - \hat{\theta}_{(2|X_1=1)})^2 \\ &\quad \cdot (\hat{\theta}_{(3|X_1=0)})^1 \cdot (1 - \hat{\theta}_{(3|X_1=0)})^2 \cdot (\hat{\theta}_{(4|X_3=1)})^1 \cdot (1 - \hat{\theta}_{(4|X_3=1)})^6 \\ &= (6/9)^6 \cdot (3/9)^3 \cdot (4/6)^4 \cdot (2/6)^2 \cdot (1/3)^1 \cdot (2/3)^2 \cdot (1/7)^1 \cdot (6/7)^6 \end{aligned}$$

as the maximal likelihood value for graph  $G$ . Note that we have dropped out terms  $1^x = 1$  and  $0^0 = 1$ .

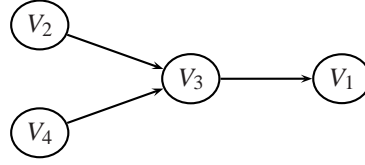
Usage of more states for genes like “low”, “medium”, and “high” follows the multinomial theory conveniently generalizing the binomial theory. Using the parameters, we can formulate hypothetical behaviors of genes in different situations fixing some expression levels and observing others; moreover, updating the parameters is easy as new data arrives.

### 13.5 Searching Bayesian network structure

Given a graph  $G$ , we know now how to calculate the parameter set  $\theta^G$  maximizing the likelihood score  $L(\theta^G : D)$ . To restrict the search space of possible networks, it is practical to have constraints on which kind of networks are allowed, that is, which dependencies are possible between genes. This helps a lot because of the super-exponential number of possible networks; for example, the number of possible graphs consisting of four nodes is 64 since there are  $\binom{4}{2}$  possible (undirected) edges, each of which can be taken to form a graph making  $2^{\binom{4}{2}}$  different graphs in total. For evaluation of the resulting candidate networks, we use again the likelihood score calculation of a network structure using maximum likelihood estimates of  $\theta^G$  when the structure is searched

$$\begin{aligned} L(G, \theta^G : D) &= \prod_m \Pr(X_1[m], X_2[m], \dots, X_n[m] : G, \theta^G) \\ &\quad \prod_m \prod_n \Pr(X_n[m] | \text{Parents}(X_n)[m] : G, \theta_n^G). \end{aligned}$$

Here,  $\text{Parents}(X_n)$  denotes the expression values of all parents of node  $V_n$  and  $\theta_n^G$  denotes the parameter of node  $V_n$  in graph  $G$ . Instead of direct calculation it is more



**Figure 13.3:** A different structure explaining conditional dependencies between the expression values of nodes  $V_1, V_2, V_3, V_4$ .

convenient to use a logarithm of  $L(G, \theta^G : D)$  for comparing different structures. After simplifying we have a surprisingly simple formula

$$\log(L(G, \theta^G : D)) = m \sum_{i=1}^n (I(V_i, \text{Parents}(V_i)) - H(V_i)),$$

where  $H(V_i)$  is the entropy of expression values  $X_i$  of node  $V_i$

$$H(V_i) = \sum_{x=0,1} \Pr(X_i = x) \lg \frac{1}{\Pr(X_i = x)}$$

which can be ignored because it is the same for all network structures. The function  $I(V_i, \text{Parents}(V_i))$  is the mutual expression information between node  $V_i$  and its parent nodes  $\text{Parents}(V_i) = \{P_1, P_2, \dots\}$  defined by

$$\sum_{\substack{x=0,1 \\ (\forall i): x_i=0,1}} \Pr(X_i = x \cap_i P_i = x_i) \lg \frac{\Pr(X_i = x \cap_i P_i = x_i)}{\Pr(X_i = x) \prod_i \Pr(P_i = x_i)}.$$

Function  $I(V_i, \text{Parents}(V_i)) \geq 0$  measures how much information the expression values of nodes  $\text{Parents}(V_i)$  provide about  $V_i$ . If  $V_i$  is independent of parent nodes, then  $I(V_i, \text{Parents}(V_i))$  has the value of zero. On the other hand, when  $V_i$  is totally predictable for given values of  $\text{Parents}(V_i)$ , then  $I(V_i, \text{Parents}(V_i))$  reduces into the entropy function  $H(V_i)$ . It should be noted that in general  $I(X, Y) \neq I(Y, X)$  so the direction of edges matters.

Is the structure presented in Figure 13.1 optimal with respect to the data? What about the structure in Figure 13.3. Because our data set is small we do not resort to the above calculations but we calculate the parameter set  $\hat{\theta}$ :

$$\begin{aligned} \hat{\theta}_2 &= 7/9, & \hat{\theta}_4 &= 3/9, \\ \hat{\theta}_{(3|X_2=0, X_4=0)} &= 1, & \hat{\theta}_{(3|X_2=0, X_4=1)} &= 1, \\ \hat{\theta}_{(3|X_2=1, X_4=0)} &= 1, & \hat{\theta}_{(3|X_2=1, X_4=1)} &= 0, \\ \hat{\theta}_{(1|X_3=0)} &= 1/3, \text{ and } \hat{\theta}_{(1|X_3=1)} &= 1. \end{aligned}$$

The comparison of these figures with the previous ones shows that the new graph is more probable with respect to data.

### 13.6 Conclusion

Our example data was complete without any missing observations of expression values, which happens quite rarely in reality. Two of the most familiar conventions

to deal with the problem of missing expression values is to omit the data rows missing some values or substituting the most common values for the missing values. Of course, what is the best policy, is an everlasting topic to debate on.

Even a more difficult problem is to decide how the discretization from expression values to logical “on/off” values should be done. Can we really use a global step value like in our example or should we use gene-specific values? On the other hand, can we use a single limit because then a small variation in measurements can toss a gene between the “on” and “off” state without any real reason? One cure for the last problem might be to use fuzzy logic, where a gene has an increasing grade of being “on” and after a fixed point it is completely or fully “on”; similarly for “off” but reversed.

### 13.7 Suggested reading

1. D’haeseleer P., Shoudan Liang S., Somogyi R. (2000) Genetic network inference: from co-expression clustering to reverse engineering, *Bioinformatics*, 16, 707-726.
2. Pe’er, D., Regev, A., Elidan, G., Friedman N. (2001) Inferring subnetworks from perturbed expression profiles, *Bioinformatics*, 17, 215S-224.
3. Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., Futcher, B. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, 9, 3273-97.

# 14 Data mining for promoter sequences

Martti Tolvanen, and Mauno Vihinen

## 14.1 General introduction

In all gene expression studies there is an underlying element of gene regulation. In order to be expressed, the gene has to be transcribed to RNA in a regulated process. Therefore, many expression experiments can be used to probe the regulation patterns. This is most obviously the case when you observe gene expression in different conditions or gene expression over time in some process, but comparisons of different cells or tissues or diseased vs. healthy tissues may benefit from analysis of regulation, too. Whatever your study, there will be changes in expression levels of some genes, either upregulation or downregulation. How the expression levels can be correlated to mechanisms of regulation is the topic of this chapter.

We start with an introduction about gene regulation and promoters and then turn to the practical issues of bioinformatics in promoter datamining: where your data comes from and how you can analyze it. We pay special attention to selecting your input data, *i.e.* finding the promoter region sequences (Input data I). This includes a discussion of the pertinent problem of identifying the genes on your chip (an obvious prerequisite), which should actually be part of your data preprocessing. Input data II discusses selection of matrices which define transcription factor binding sites. The analysis part is shorter and divided into three sections: 1) searching known regulatory sites (“pattern matching”, 2) searching potential regulatory patterns that are not known a priori (“pattern recognition”) and 3) analyzing already-assigned regulatory patterns and sites.

## 14.2 Introduction to gene regulation

Changes in expression levels of genes can be due to a number of reasons. First, there is variable chromatin accessibility: some regions of DNA are packed too tightly to allow strand separation or even binding of various factors. One well-known mechanism of chromatin packing is based on methylation of cytidines in CpG islands to effect gene silencing via heterochromatin formation, and another

one is related to histone acetylation/methylation. Second, there are specific transcription factors which may inhibit or activate transcription. The balance between activating and inhibiting transcription factors may determine the expression level in loosely packed euchromatin. Third, there are some genes that seem to be on “by default” without much apparent regulation, the so-called maintenance or household genes which are needed in all cell types. “Household gene”, however, does not imply constant expression; in any gene the expression level will fluctuate over time due to the general metabolic activity of the cell, especially the activity and amount of parts in the transcription machinery itself. If maintenance genes are used for normalization of array data it is necessary to verify the constant expression *e.g.* by RT-PCR in the system and conditions of the experiment. (See also the *Chapter 10, Normalization*)

Data mining for gene regulation is currently feasible only in the second case above, *i.e.* in search and study of transcription factor binding sites. The regulation of chromatin structure changes are still poorly understood even in the experimental level, so there is no basis for theoretical analyses either. However, analyses of DNA methylation are possible now, and may soon be extended to global analyses of methylation of the genome (“methylome” analyses) in various conditions.

Most known transcription factor binding sites (TFBS) are located close to the transcription start site (TSS), particularly in the 500 bp directly upstream (5') of TSS. This upstream region is often referred to as the promoter region or proximal promoter region (but sometimes the binding sites themselves are called promoters, and sometimes the word promoter refers to the so-called “basal promoter”, or the binding region of the RNA-polymerase and other regular components of the transcription machinery). Other regions that activate transcription (enhancers) may occur almost anywhere downstream (3') or upstream from the gene, even 20 kB away from the promoter region, or in introns. The methods for the search of regulatory elements that we describe could be used for potential enhancer regions as well as for the promoter regions, but commonly only promoters are analyzed. This is because a larger search space weakens the signal-to-noise ratio, and the TFBS are quite short and vague, *i.e.* weak signals. Therefore, we discuss mainly methods and examples of data mining in the proximal promoter regions.

Coexpressed genes are often grouped by different clustering methods. It is reasonable to present the hypothesis that some of the similar changes of expression are due to the effect of the same or similar transcription factors - therefore it makes sense to search for shared sequences that could be TFBS. Of course, depending on your clustering (cluster sizes and number), you may have several mechanisms affecting within one cluster, or you may have the same mechanism affecting members of several clusters, but in any case, at least some clusters may show enrichment of similar TFBS. It makes even sense to look for over- and/or under-represented TFBS in sets of genes with increased or decreased expression in one experiment only.

Sometimes you may correlate known functional associations of genes to your promoter analysis results to make more sense of emerging groups of coexpressed genes and the biological meaning of the coexpression.

In brief, it is reasonable to presume that coexpression may imply coregulation, and looking for enriched shared patterns, especially known TFBS, in promoter re-

gions allows you to formulate hypotheses of regulation mechanisms for focused experimental testing. However, before experimental verification, your results will be only sets of potential TFBS of potentially coregulated genes.

### 14.3 Input data I: promoter region sequences

Promoter analyses need the promoter region sequences, but getting them may not be trivial! In this section, we introduce some tools and data sources that we have found useful for retrieving promoter regions for human genes.

First of all, it is instrumental to know precisely which genes are contained in the microarray. The data from the manufacturer is not always systematically presented or complete. There may be a mix of references to GenBank accession numbers, UniGene clusters, gene names, protein accession numbers, RefSeq mRNA or protein accession numbers, Ensembl gene codes and EntrezGene ID codes (previously LocusLink), and if you are lucky, even actual sequences that are contained on the array.

Even if we assume that all data provided by the manufacturer is correct, there are some issues that you should be aware of before you can fully trust your gene assignments:

- gene names have been subject to many changes, so your data may not contain the current official names
- the UniGene clusters of ESTs are dynamic entities that undergo fusions and fissions in every UniGene build as more data becomes available, so your ESTs may be assigned to other genes, or may lack a UniGene association in the current build. New UniGene cluster numbers appear in every new build and old ones are made obsolete. Bottom line: save UniGene codes as a last resort after using other, stable data items.
- the data regarding your filter or chip may have been updated after it left the factory, so the shipped gene lists may not be as complete and up-to-date as you can find at the manufacturer's web site. You should definitely work with the most recent data available as long as it refers to the same manufacturing batch that you actually used.
- you may find several RefSeq or Ensembl mRNA and protein codes for a single gene locus, corresponding to alternatively spliced forms - however, many alternative splice forms are still missing from the databases
- the provided EST code is not always the actual sequence on the array, because the manufacturer may have used a longer and more correct version of the corresponding mRNA
- many genes, perhaps 25 to 30 % of human genes, contain alternative transcription start sites, and therefore alternative promoters (see Database of Transcriptional Start Sites, DBTSS, at <http://dbtss.hgc.jp/>). Currently very few of them are annotated in the genome databases, and there is very

little you can do to make sure you have the correct one for the tissue/system you studied. DBTSS (<http://dbtss.hgc.jp/>) is the best data resource here, but there are not any ready-made tools to utilize their data in promoter analysis.

The ultimate source for the promoter regions is in the genome data. The annotations of gene locations are still incomplete, and there are no ready-made tools for making decisions of the transcription start site and informing the user of the reliability of each decision. Currently (late 2005) we have found that human promoters are easiest to retrieve in large scale from two public data sources. They are multi-species resources, so when more genomes become available, the same approaches work.

- University of California in Santa Cruz (UCSC) provides ready-made collections of upstream sequences of as many genes as they can localize based on RefSeq mRNA sequences - different sets contain different lengths of 5'-sequences preceding the TSS, called upstream1000, upstream2000 etc. The data from the most recent human genome (hg17 build 35, May 2004) is at <http://hgdownload.cse.ucsc.edu/goldenPath/hg17/bigZips/>, but you will always find most recent data at <http://genome.ucsc.edu/downloads.html> under Full data set. Similar data is available for mouse and some other species (but not for all represented species). Start from <http://hgdownload.cse.ucsc.edu/downloads.html> and go to "Full data set" for each species.
- BioMart (<http://www.ensembl.org/Multi/martview>) allows direct retrieval of upstream sequences of any length if you can map your genes to Ensembl gene codes or RefSeq codes (more details follow). Coding sequences, introns and downstream regions can be downloaded as well.

For a smaller number of genes the most reliable data (regarding the placement of TSS) is available in the Eukaryotic Promoter Database (EPD, <http://www.epd.isb-sib.ch/>), which contains promoter regions of 500 bp, only from genes with an experimentally verified TSS. For example, you can find 1,871 human entries in EPD 84 (out of 4800 for all species). DBTSS (<http://dbtss.hgc.jp/>) contains data for over 8,000 human genes as well as thousands from other species. Organism-specific promoter databases may exist to fit your needs, consult the genome site of your favourite organism or Michael Zhang's lab at <http://ruilai.cshl.org/software/index1.htm>.

However, commercial promoter analysis packages may contain ready-made promoter data sets containing quality evaluations and much else, *e.g.* as in ElDorado of the Genomatix package (<http://www.genomatix.de/products/ElDorado/>).

If these sources fail, or if you feel you cannot trust all of your findings, you can always resort to using your own similarity searches to place the gene in the genome assembly, and then pick the region preceding the gene, but this is far from straightforward. For example, with bad-quality ESTs you run a risk of finding homologs as the best hit instead of the actual gene. The same risk applies if your



options and more recent databases available, and therefore gives better results.

- similarity search tools, such as Blast or BLAT (<http://genome.ucsc.edu/cgi-bin/hgBlat?command=start>). For large-scale searches you probably want to install the program locally and use text processing tools such as Perl for extracting relevant information from the result files. BLAT has the advantage of giving directly the coordinates in the genome, and even on-line BLAT accepts modest batches of sequences (25 at a time) if you just want to patch missing data
- Batch Entrez - mass retrieval of sequences from NCBI (<http://www.ncbi.nlm.nih.gov/entrez/batchentrez.cgi?db=Nucleotide>)

If you want to be really certain that you know which genes you are dealing with, you can follow several paths in the diagram to see if they all lead you to same EntrezGene IDs or HUGO names. Remember that there may be several RefSeq mRNAs or Ensembl transcripts per gene, whereas EntrezGene IDs are unique.

Our recommended procedure for the above approach is this (numbers as in Figure 14.1):

1. Starting from whatever codes you have, preferably EntrezGene/LocusLink codes and HUGO/HGNC gene names, fetch at least once the default output from IDconverter (CNIO). Repeat with several code sets to see if you can improve the coverage.
2. If you miss more Ensembl gene codes than you would like, you can use GenBank or UniGene codes for some semi-manual detective work. For GenBank accession numbers, use batch Entrez to retrieve nucleotide sequences, then use Blat to find their locations in the genome and see if there is an annotated gene right there. In our experience, this approach has a high success rate. If you have a file of UniGene codes, you can use batch Entrez for UniGene (<http://www.ncbi.nlm.nih.gov/entrez/batchentrez.cgi?db=UniGene>) to get UniGene links for all of them, and then try to resolve them one by one, for example by copying and pasting a set of representative mRNA codes, which you may apply to nucleotide Batch Entrez and Blat as above. Using UniGene is more laborious, and in some cases impossible (*e.g.* for some retired codes, which have been split into a number of new clusters).
3. Use the Ensembl gene codes to retrieve the upstream sequences as described in detail below.

If you work with the Affymetrix Arabidopsis chips, you may want to look at VIZARD (<http://www.anm.f2s.com/research/vizard/>) which includes annotation and upstream sequence databases for the majority of genes represented on the Affymetrix Arabidopsis GeneChip® array. The Broad Institute at MIT provides upstream sequence data for *Neurospora crassa* and several other organisms, see *e.g.* <http://www.broad.mit.edu/annotation/fgi/>.

## 14.4 Using BioMart to retrieve promoter regions

The preferred option for retrieving upstream sequences is found at the BioMart service (<http://www.ensembl.org/Multi/martview>) of the Ensembl project. This is limited only by the number of Ensembl genes that are annotated and by the accuracy of the annotation. Many genes still have many multiple entries in Ensembl, and some entries are for pseudogenes. However, the service is very easy to use. Note that you will want to use primarily Ensembl data, not Vega (which is manually annotated but still very incomplete data).

For some microarrays (especially Affymetrix chips), BioMart provides direct mappings. Such mappings with microarray contents have become more common in the genome sites, both at Ensembl and at UCSC, and the annotations provided by the manufacturers have improved, too. Therefore, retrieving the upstream sequences is less of a technical problem now, but the problem of correct transcription start sites remains a serious one, especially in the case of alternative promoters.

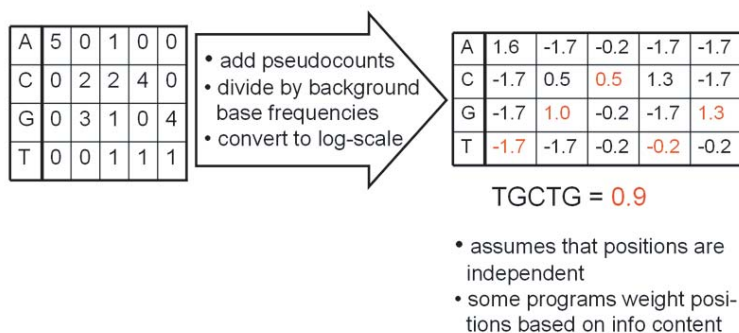
Please see the book web site at <http://www.csc.fi/molbio/arraybook/> for extra material and examples of using BioMart.

### 14.4.1 Final warnings regarding upstream data

- in some cases the genome assembly may be broken close to the start of your gene, so that you do not get a full 1000 (or whatever) nucleotides of sequence. In Ensembl this shows as a long string of N's in the sequence
- the start of a RefSeq mRNA may not always correspond to a true transcription start site. There are many mRNAs which give evidence of longer transcript variants than the current RefSeqs. UCSC genome browser relies directly to the RefSeq start sites in their upstream data sets.
- Ensembl tries to extend the 5'-end as far as possible, but in a few cases this leads erratically long sequences, due to seemingly mis-spliced mRNA versions

## 14.5 Input data II: the matrices for transcription factor binding sites

In the “pattern matching” approach we need to know the TFBS sequences in advance. TFBS are never exact sequences without exceptions, but instead, they are patterns with more or less preferred bases at each nucleotide position in the site. Generally the preferences are represented as a weight matrix, derived from observed base frequencies in each position as shown in Figure 14.2.



**Figure 14.2:** Creation of weight matrices to describe a TFBS (and a scoring example of a sequence against this matrix). Figure by Eija Korpelainen; includes matrix by Wyeth Wasserman.

The largest collections of TFBS matrices come from commercial sources, with the Transfac or Genomatix packages. Transfac database ver. 7.0 (with 398 matrices) is publicly available (<http://www.gene-regulation.com/pub/databases.html#transfac>), whereas the current commercial version, Transfac Professional 9.4., contains 774 matrices. The Genomatix matrix library 6.0 contains 664 matrices for over 2,000 transcription factors. Genomatix offers evaluation accounts with a limited number of searches per month for free (<http://www.genomatix.de/shop/evaluation.html>).

However, the numbers can not be used for direct comparison of the extent or usability of these products, since both program suites and libraries have their special strengths.

JASPAR (<http://jaspar.cgb.ki.se/>), is a completely free academic alternative, a recent (2004) matrix library of 111 matrices, in which great care has been taken in data selection to arrive at good-quality matrices. A combination of JASPAR and the public Transfac might be your best pick if you can not invest in the commercial full programs.

Another approach to improve the quality of TFBS matrices is to allow correlations between base positions in the weight matrices (Zhou and Liu, 2004).

### 14.5.1 Searching known patterns

Once you have found the genes and their 5' regions you can proceed to the analysis of patterns in the regions. The simplistic approach is to search single known sites and estimate the over- or under-representation of the findings. However, because more than 90 % of such findings will have no biological significance, even when you search for known TFBS patterns, you would need additional information, either from other TFBS in the vicinity (to form “modules” or “frameworks”), or from comparative genomics, studying promoters of orthologous genes from several species (“phylogenetic footprinting”).

For more details about comparative genomics in TFBS finding, see the ar-

ticle by Prakash and Tompa (2005). For practical implementations you can try rVista (Loots and Ovcharenko 2004; <http://rvista.dcode.org/>) and Compare-Prospector (Liu et al. 2004; <http://compareprospector.stanford.edu/>).

If you want to run a check of known transcription factor binding sites, Transcription Regulatory Regions Database (TRRD, <http://www.mgs.bionet.nsc.ru/mgs/dbases/trrd4/trrdintro.html>) is a site with lots of information, but only for on-line searching. The complete database is not available. The Transcription Factor Database (TFD) and some tools that utilize the data are available at <http://www.ifti.org/>.

Transfac (<http://www.gene-regulation.de/>) offers PC software for promoter analysis, Transplorer. Their “Transfac Professional” package contains search tools, too, of course, and many tools, *e.g.* Match, P-Match and Patch have public versions available at the Transfac web site (for non-commercial use only).

TransFAT is a recent datamining tool designed to detect under or over-representation of putative TFBS by comparing two sets of genes. This is a promising-looking tool with built-in statistics, but we have not tested it properly yet. It is available at <http://babelomics.bioinfo.cnio.es/transfat/transfat.cgi> and <http://babelomics.bioinfo.cipf.es/transfat/cgi-bin/transfat.cgi>.

Whatever method you choose to search for TF binding sites, you have to bear in mind that they are short patterns that will be found very frequently by chance alone, so you need to confirm the statistical significance of your findings.

You may improve your results and their significance by searching clusters of several sites, *i.e.* module searching. Genomatix (see above) offers the programs FrameWorker and ModelInspector for creating and searching combinations of several TFBS in one promoter. The ModelInspector can also be used for searching with the Genomatix library of experimentally verified modules. The makers of TransFac have made Composite Module Analyst for this purpose (<http://www.gene-regulation.com/pub/programs.html#CMAAnalyst>) and it is freely available for non-commercial use.

### 14.5.2 Pattern search without prior knowledge of what you search

Several programs and different approaches are available for automated pattern discovery in your promoter region sequences. We will mention here a few examples of different programs and strategies, all of which are capable of finding patterns in sequences without their previous alignment. More extensive reviews and comparisons can be found in Kel et al., 2004, and Tompa et al., 2005.

Expression Profiler is a software suite for many tasks in microarray data analysis (<http://ep.ebi.ac.uk/EP/>). It contains the program SPEXS, Sequence Pattern EXhaustive Search, for pattern discovery, which finds regular-expression-like discrete patterns. As an example, this program was used by Vilo et al. (2000) to find regulatory patterns from publicly available yeast expression data. To overcome the fuzziness created by uncertainties in clustering, they generated a large number of different clusters (52,000, with lots of overlapping) and analyzed enriched patterns from all of them. After filtering and grouping the patterns, they had 62 groups, 48 of which contained patterns matching some sites in the *Saccharomyces cerevisiae*

Promoter Database.

MEME (<http://meme.sdsc.edu/meme/website/intro.html>) is based on position-dependent letter-probability matrices which describe the probability of each possible letter at each position in the pattern. The WWW interface analyzes your set of sequences which you believe to contain common patterns, and gives you the strongest patterns that MEME finds. In addition, you get automatically results from a companion program MAST which locates the occurrences of the patterns in your set of sequences.

AlignACE (<http://arep.med.harvard.edu/mrnadata/mrnasoft.html>) provides a Gibbs sampling algorithm for finding patterns in DNA sequences. The program is optimized for finding multiple motifs and it automatically considers both strands in the sequence. The authors used the program to identify successfully several known transcription factor binding sites and to propose some previously unknown regulatory mechanisms based on the earliest public yeast gene expression data sets (Roth et al. 1998).

All of the above methods depend on a clustering that you have made before pattern discovery, and it is presumed that the clustering is correct. Kimono (<http://www.fruitfly.org/~ihh/kimono/>) takes a unified approach that optimizes clustering and pattern discovery together. The approaches described above first find genes with similar expression patterns, and then see if they have similar promoters. Kimono finds clusters of genes that have similar expression patterns and similar promoters as described by Holmes and Bruno (2000).

Credo (Cis-Regulatory Element Detection Online, <http://mips.gsf.de/proj/regulomips/>) is a recent service which combines the algorithms of AlignACE, DIALIGN, FootPrinter, MEME and MotifSampler.

However, none of the pattern recognition programs are directly suitable for large-scale comparisons of promoters from microarray data, but need some scripting skills or manual work for parsing and combining the results.

## 14.6 Examples

Please see the book web site at <http://www.csc.fi/molbio/arraybook/> for extra material on promoter analysis using computer software.

## 14.7 Suggested reading

1. Holmes, I., and Bruno, W. J. (2000) Finding regulatory elements using joint likelihoods for sequence and expression profile data. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 8, 202-10.
2. Kel, A., Tikunov, Y., Voss N., and Wingender E. (2004) Recognition of multiple patterns in unaligned sets of sequences: comparison of kernel clustering method with other methods. *Bioinformatics*, 20, 1512-1516.
3. Liu, Y., Liu, X. S., Liping, W., Russ, B. A., and Batzoglou, S. (2004) Eukaryotic Regulatory Element Conservation Analysis and Identification Using Comparative Genomics. *Genome Res.*, 14, 451-458.

4. Loots, G. G. and Ovcharenko, I. (2004) rVISTA 2.0: evolutionary analysis of transcription factor binding sites. *Nucl. Acids Res.*, 32, W217-W221.
5. Prakash, A., and Tompa, M. (2005) Discovery of regulatory elements in vertebrates through comparative genomics. *Nat. Biotechnol.*, 23, 1249-56.
6. Roth, F. P., Hughes, J. D., Estep, P. W., and Church, G. M. (1998) Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat. Biotechnol.*, 16, 939-45.
7. Tompa, M., Li, N., et al. (2005) Assessing computational tools for the discovery of transcription factor binding sites. *Nat. Biotechnol.*, 23, 137-44.
8. Vilo, J., Brazma, A., Jonassen, I., Robinson, A., and Ukkonen, E. (2000) Mining for Putative Regulatory Elements in the Yeast Genome Using Gene Expression Data. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 8, 384-94.
9. Zhou, Q. and Liu, J. S. (2004) Modeling within-motif dependence for transcription factor binding site predictions. *Bioinformatics*, 20, 909-916.

# 15 Biological sequence annotations

Juha Saharinen

## 15.1 Annotations in gene expression studies

Sequence annotations are, in general, information attached to the regions in the sequences. The majority of the several hundreds of bioinformatics databases currently publicly available either provide annotations to sequences or refer to certain sequences (see [4]). Basically, annotations can be either experimentally proven or computational predictions. Examples of annotations include localization of a gene in a genomic sequence, a SNP marker in a gene, a verified translation start site in mRNA, a predicted transmembrane domain in a protein sequence, an experimentally validated transcription factor binding site in a promoter, localization of a protein in the cytoplasm, involvement of a protein in MAP-kinase signaling cascade, association of a mutation in a gene with a disease etc. After the interesting sets of genes have been discovered by various statistical analysis from expression microarray data, searching and applying the annotations to the genes (transcripts) are used to provide the biological meaning for the observed results.

## 15.2 Using annotations with gene expression microarrays

With expression microarrays, the use of sequence annotations typically involves the retrieval of more information concerning the genes that have been identified as differentially expressed in the studied system. Often the first step is to locate an identifier for the genes. Depending on the used chip platform and the information provided, a suitable identifier to be used in *e.g.* genome browsers can be provided. A number of different sequence / gene / transcript identifier systems are currently used. The most important are the GenBank, EMBL, DDBJ, RefSeq, UniGene, LocusLink, and Ensembl identifiers [2, 5, 8, 10, 19, 22].

UniGene, which is quite commonly used identifier in microarrays, is based on a computational system for automatic clustering of the GenBank data into gene centric clusters, where one cluster is aiming to represent a single gene. Unigene is regularly updated from the source data. In the update process, UniGene clusters can

be split apart or joined together, which often makes the gene identification difficult, since the used cluster ID may have become obsolete. Better than UniGene in this respect, is the RefSeq system, which provides non-redundant, curated and stable identifiers for mRNAs. MatchMiner is a tool that can be used to translate between different identifiers for the same gene, it is available as a web-page as well as a command line tool [3].

Most of the annotation servers are well cross-referenced, *i.e.* they provide links to the corresponding genes also in other annotations servers. Good starting points for getting more biological information about the selected genes are the major publicly accessible genome browsers: Ensembl, NCBI Map Viewer, UCSC Genome Browser, and the GeneCards server [8, 11, 20, 22]. All these servers share the same actual genome builds, thus the physical coordinates are interchangeable, provided that the same genome build revisions are used. More information and practical examples of these genome browsers are available in the User's guide to human genome resources [2]. Also see *Chapter 14* (Data mining for promoter sequences), which also describes some aspects of using the Ensembl and EnsMart server [12]. Using these servers, it is easy to obtain corresponding protein level information (protein domain data provided by pfam, SMART, often accessed by InterPro[1, 14, 16]), expression information in different tissues, typically based on EST or SAGE (serial analysis of gene expression) data, orthologous and paralogous genes, SNPs in the gene locus (typically from dbSNP [22]) and possible identified genetic disorders (OMIM database [6]) for the gene in question.

If no suitable gene identifier is available, the referenced gene on the array can be identified with sequence based homology searches. These are provided by Blast or SSAHA in Ensembl, BLAT in UCSC Genome Browser, and Blast or MegaBlast in NCBI [13, 17, 25].

### 15.3 Using Ensembl server to batch process of annotations

When working with large sets of genes, a programmable interface is preferred to "high-throughput copy-pasting" of www-pages. The majority of the www-based annotation servers are constructed in an identical way as a precomputed relational database, which is accessible through the www-interface. However, Ensembl also provides additional ways to interact with the data. The OpenSource MySQL relational database server, serving/providing the Ensembl data, can be connected directly and data queried via SQL commands. For best performance, the Ensembl data can be downloaded and installed to a local MySQL server: The Ensembl MySQL server is available at address [ensembl.mysql.org](http://ensembl.mysql.org). For Finnish users, Biomedicum Bioinformatics Unit (<http://www.bioinfo.helsinki.fi>) is mirroring Ensembl databases for variety of organisms, giving substantially better performance than the popular [ensembl.mysql.org](http://ensembl.mysql.org) server. Also the BioPerl with Ensembl extensions are installed and available for users.

In addition, a comprehensive Perl API (application program interface) to access the Ensembl data is available. This requires the installation of BioPERL libraries as well as the Ensembl additions [21]. Documentations as well as examples of these are available at <http://www.ensembl.org/Docs/>. Here are two small

examples using these alternative interfaces to the Ensembl data. In example 1 [See online material at <http://www.csc.fi/molbio/arraybook/>], 10 probe sets from Affymetrix U133A chip are queried from Ensembl *Homo sapiens* database and their Ensembl GeneStableIDs, and gene's sequence locations are returned. Example 2 [See online material at <http://www.csc.fi/molbio/arraybook/>], shows a small Perl script, using EnsemblAPI, to retrieve the genomic sequence from a given organism, chromosome and physical position.

#### 15.4 Biological pathways and GeneOntologies

In order to get more insight into the observed gene expression changes, it is meaningful to relate the microarray results with the possible biological context, in the form of, for example, known metabolic or signal transduction pathways. The Kyoto Encyclopedia of Genes and Genomes (KEGG) PATHWAY database has been perhaps the most known resource for mapping genes to known pathways [9]. KEGG pathway and other KEGG databases are available in addition to www-browser also as local installations and as a Webservice, supporting high-throughput analysis from most programming languages using SOAP protocol. Using KEGG over the www-browser typically returns a graph of a pathway, where the gene of interest is located as well as possible links to other pathways. Another place for visualizing pathways is BioCarta (<http://www.biocarta.com/genes/index.asp>).

The GeneOntology (GO) is currently the most comprehensive collaborative effort to assign consistent ontologies to gene products in a species independent manner. In essence, ontologies are controlled vocabularies, providing standardized names or terms for different biological events or properties. GO ontologies belong to three separate domains: cellular components, biological processes and molecular functions, which are considered independent of each other. Examples of these include biological process: transforming growth factor beta receptor activity; cellular component: extracellular matrix; molecular function: calcium ion binding. The data in GO is in a hierarchical format; specifically in a so called diacyclic graph (DAG), which enables a given term to have multiple parents and childrens. An example is protein kinase activity, which is a child of kinase activity and parent of tyrosine kinase and serine/threonine kinase. Gene products in GO can be annotated to any number of GO terms. The evidence for making an association of a gene to a term can be either experimental or a computational prediction, which information is also available.

#### 15.5 Using GeneOntologies for gene expression data analysis

Whilst GO is a very useful tool in deciphering the biological processes, that are linked to the genes identified by microarray as described in other chapters, the organized data in GO can also be used in an analytical manner. Since a large number of genes have been mapped to GO, one can analyze whether certain gene sets, say, significantly differentially expressed gene clusters, are enriched in some GO group. Multiple different software packages exist for this analysis [7, 15, 18, 23, 24] (see also <http://www.geneontology.org/GO.tools.html>). Typically these

tools test the enrichment or depletion of a group of genes in the all possible GO groups. The input is a list of genes, ordered by *e.g.* *p*-value and it is usually tested by using Fisher's exact test, *i.e.* asking the question how likely it is to observe *x* members out of *y* genes in given annotated pathway, within the *z* first genes in the list out of *w* total genes in the array. Differences exist *e.g.* whether the software can deal with multiple testing problem, the way the results are linked to other databases and what are the platform requirements (www-page, Windows, Java).

## 15.6 Concluding remarks

Adding biological annotations to the genes identified with various statistical means from the gene expression data provides the link to the underlying biological phenomena and can provide further information about the biological meaning of the expression changes than often daunting long lists of genes. Especially in the pathway or gene grouping analysis of gene expression data, the GO annotation is very important. In addition, biological annotations often provide the fastest way to get an idea of the many hypothetical genes identified with microarrays, for which no other data is available.

## 15.7 References

1. Bateman, A., et al. (2004) The Pfam protein families database. *Nucleic Acids Res*, 32, Database issue, D138-41. <http://www.sanger.ac.uk/Software/Pfam/>
2. Benson, D.A., et al. (2005) GenBank. *Nucleic Acids Res*, 33, Database Issue, D34-8. <http://www.ncbi.nlm.nih.gov/Genbank/index.html>
3. Bussey, K.J., et al.(2003) MatchMiner: a tool for batch navigation among gene and gene product identifiers. *Genome Biol.*, 4, R27. <http://discover.nci.nih.gov/matchminer>
4. Database issue (2005), *Nucleic Acids Research*, 33.
5. DDBJ, <http://www.ddbj.nig.ac.jp/>.
6. Hamosh, A., et al. (2005) Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. *Nucleic Acids Res.*, 33, Database Issue., D514-7. <http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=OMIM>
7. Hosack, D.A., et al. (2003) Identifying biological themes within lists of genes with EASE. *Genome Biol*, 4, R70.
8. Hubbard, T., et al. (2005) Ensembl 2005. *Nucleic Acids Res*, 33, Database Issue, D447-53. <http://www.ensembl.org/>
9. Kanehisa, M., et al. (2004) The KEGG resource for deciphering the genome. *Nucleic Acids Res.*, 32. Database issue, D277-80. <http://www.genome.jp/kegg/>

10. Kanz, C., et al. (2005) The EMBL Nucleotide Sequence Database. *Nucleic Acids Res.*, 33, Database Issue, D29-33. <http://www.ebi.ac.uk/emb1/>
11. Karolchik, D., et al. (2003) The UCSC Genome Browser Database. *Nucleic Acids Res.*, 31, 51-4. <http://genome.ucsc.edu/>
12. Kasprzyk, A., et al. (2004) EnsMart: a generic system for fast and flexible access to biological data. *Genome Res.*, 14, 160-9.
13. Kent, W.J. (2002) BLAT—the BLAST-like alignment tool. *Genome Res.*, 12, 656-64.
14. Letunic, I., et al. (2004) SMART 4.0: towards genomic data integration. *Nucleic Acids Res.*, 32, Database issue, D142-4. <http://smart.embl-heidelberg.de/>
15. Martin, D., et al. (2004) GOToolBox: functional analysis of gene datasets based on Gene Ontology. *Genome Biol.*, 5, R101.
16. Mulder, N.J. et al. (2005) InterPro, progress and status in 2005. *Nucleic Acids Res.*, 33, Database Issue, D201-5. <http://www.ebi.ac.uk/interpro/>
17. Ning, Z., A.J. Cox, and Mullikin, J.C. (2001) SSAHA: a fast search method for large DNA databases. *Genome Res.*, 2001, 11, 1725-9.
18. Osier, M.V., Zhao, H., and Cheung, K.H. (2004) Handling multiple testing while interpreting microarrays with the Gene Ontology Database. *BMC Bioinformatics*, 5, 124.
19. Pruitt, K.D., Tatusova, T., and Maglott, D. R. (2005) NCBI Reference Sequence (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. *Nucleic Acids Res.*, 33, Database Issue, D501-4. <http://www.ncbi.nlm.nih.gov/RefSeq/>
20. Safran, M., et al. (2002) GeneCards 2002: towards a complete, object-oriented, human gene compendium. *Bioinformatics*, 18, 1542-3. <http://bioinformatics.weizmann.ac.il/cards/>
21. Stajich, J.E., et al. (2002) The Bioperl toolkit: Perl modules for the life sciences. *Genome Res.*, 12, 1611-8. <http://bioperl.org/>
22. Wheeler, D.L., et al. (2005) Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res.*, 33, Database Issue, D39-45.
23. Zeeberg, B.R., et al. (2003) GoMiner: a resource for biological interpretation of genomic and proteomic data. *Genome Biol.*, 4, R28.
24. Zhang, B., et al. (2004) GOTree Machine (GOTM): a web-based platform for interpreting sets of interesting genes using Gene Ontology hierarchies. *BMC Bioinformatics*, 5, 16.

25. Zhang, Z., et al., (2000) A greedy algorithm for aligning DNA sequences. *J. Comput. Biol.*, 7, 203-14.

# 16 Software issues

Jarno Tuimala and Teemu Toivanen

Traditionally, biological data, including DNA and amino acid sequences and DNA microarray results, have been stored in a flat file format. Flat files are basically just text files, which have been formatted in such a way that programs can read their contents. Flat files are not the ideal solution for long term storage of large data sets. They also cause problems when the same data needs to be analyzed using several programs that use a different kind of file format.

The data standardization and storage in central databases (see *Chapter 7* for more details) is one solution to the data format issues. However, this solution has not yet fully matured, and most of the programs used for the DNA microarray data analyses are not able to read the data directly from these databases or even import the XML-format the databases commonly produce. Databases and programs will surely learn to cooperate better over time. The standardization and storage of the data is currently under heavy development, and many software developers are implementing their own solution to these problems. However, the solutions are not always MIAME compliant.

## 16.1 Data format conversions problems

Especially, if freeware tools are used for the analysis, there is often a need to export the partly analyzed data from one program to another. More often than not, the datafile formats are not compatible, which means additional and tedious conversion work. Small amounts of data can quite easily be converted between formats using Microsoft Excel or any other spreadsheet program. However, the work load becomes quickly unbearable when the amount of data increases. Excel has some macro programming capacities, but there exists also other, and possibly better tools, if one is interested in programming (see section on programming). In addition, Excel does not tolerate huge data files (more than 65 536 rows or 256 columns). Sometimes it becomes necessary to transpose (switch the rows and columns) the data file, and if there are more than 256 genes, Excel can't be used. There has been some problems with Excel between its language versions (date fields change, comma and dot are switched in decimal numbers and such). If you run into these problems it's advisable to use some programming language instead of a spreadsheet program.

## 16.2 A standard file format

If the data needs to be imported into several programs at the same time, it is often easier if a “standard file” is first generated from the files written by the chip scanner. Standard files can be produced in Excel or another spreadsheet program easily. Briefly, the first column of the standard file should contain the gene identifiers, for example their common names. The second column should consist of the sequence accession numbers, Genbank accession numbers are probably the most usable ones. The next four columns should contain the spot and background intensities of the used colors. Depending on your analysis needs, it is also possible to use ratios, for example normalized log ratios, instead of intensity values. The standard file should be saved in a tab-delimited text format (usually “export” or “save as” menu option and from there “Text (tab delimited)” or similar). It’s sometimes possible to view data in some text editor, just to have a peek how the actual data is stored.

## 16.3 Programming

When the capacities of the spreadsheet program are not sufficient or many similar files need processing, some programming tools can be used instead for the datafile management purposes or for data analyses. There are actually several programming languages that are especially suited for the data file formatting and other text file manipulations.

### 16.3.1 Perl

One of the most common languages is Perl, which has very powerful and easy to use text manipulation tools. Perl is available for free for UNIX, Linux <http://www.perl.org> and PC machines <http://www.activeperl.com>. Perl is a programming language, which means that getting to know it takes some time and effort. However, if data conversion tools are needed everyday, it would definitely be worthwhile to befriend Perl.

### 16.3.2 R

R is a free statistical analysis tool and a self-sufficient programming language. It is available for UNIX, Linux, Macintosh and PC platforms. In R, scripts for analyses and data file manipulations can be easily constructed. R has many add-on packages for cluster analysis, self-organizing maps, and neural networks, among others. There are also many packages available that have been specifically tailored for DNA microarray data analysis: Bioconductor project develops and updates many of these packages.

## 16.4 Examples

Please see the book web site at <http://www.csc.fi/molbio/arraybook/> for code examples and computer software presentations.