

U N I V E R S I T Y O F T A R T U
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
Institute of Computer Science

Konstantin Tretyakov

G=MAT:
Linking Transcription Factor
Expression with DNA Binding

Master's thesis (40 cp)

Supervisors: Sven Laur, D.Sc. (Tech.)
Jaak Vilo, Ph.D.

Autor: "...." juuni 2008
Juhendaja: "...." juuni 2008
Juhendaja: "...." juuni 2008

TARTU 2008

Contents

Introduction	5
1 Biological Background	7
1.1 The DNA	7
1.2 Gene Expression Mechanisms	8
1.3 Microarray Data	10
1.4 DNA Sequence Data	11
1.5 Gene Ontology Annotation Data	12
2 Mathematical Background	13
2.1 Matrix Algebra	13
2.2 Probability Theory	16
2.3 Optimization Theory	19
2.4 Linear Models	21
2.4.1 Model Parameter Estimation	22
2.4.2 Model Assessment	24
3 The G=MAT Model	27
3.1 Notation	27
3.2 Model Specification	29
3.3 Rationale	30
3.4 Interpretation	32
3.5 Related Work	32
3.6 Example	33
4 Parameter Estimation Methods	37
4.1 Least Squares Regression	37
4.2 Regularized Least Squares Regression	44
4.3 Sparse Regression	46
4.4 Correlation-based Estimate	47
4.5 Randomization-based Attribute Selection	52
4.6 Example (continued)	53

5	Model Performance Analysis	59
5.1	The Spellman Dataset	59
5.2	Evaluation of Results	60
5.3	The Artificial Dataset	62
5.4	Prediction versus Attribute Selection	64
5.4.1	Experimental Setup	64
5.4.2	Influence of Noise	66
5.4.3	Influence of Incomplete Data	67
5.4.4	Theoretical Justification	67
5.5	Comparison of Methods	73
5.5.1	Choice of the λ Parameter	74
5.5.2	Comparison of Performance	75
5.5.3	The Effect of Centering	75
6	Applications of G=MAT	77
6.1	Discovering Process-specific TFs and Motifs	77
6.2	Motif Discovery	80
6.3	Automatic GO Annotation	80
6.4	A Non-biological Application	82
	Summary	85
	Resümee (eesti keeles)	87
	References	89
	Appendices	95

Introduction

In the world, so vast and cheerful,
Every single living creature
Is of little cells consistent
Little cells with many features.
If we study cells in detail,
Understand their complex functions,
We could find a cure for sickness
Be it cancer or infections.

Little cells are very complex,
They have membranes, hormones, enzymes,
All the processes inside them.
Can be complicated sometimes
In this work we look in detail
At transcription regulation.
This is quite a simple process
That we model as equation.

The cell of a living organism is perhaps one of the most studied and at the same time the least understood objects on our planet. Processes that take place in this tiny piece of matter are strikingly complex, yet, concurring in harmony together, they coordinate the life cycle of the whole organism. Understanding these processes is therefore one of the ultimate goals of contemporary medical and biological studies that may lead to better treatment of diseases and novel biotechnological discoveries.

Current biological knowledge suggests that most of the functions in the cell are performed by certain molecules known as *proteins*. Despite the diversity of different proteins (*hormones, enzymes, transport proteins*, etc), their structure is generally the same on the molecular level. Each protein can be constructed as a linear chain of *amino acids*. These linear chains are encoded by the *DNA molecules* that are stored in the chromosomes of the cell.

One of the most fascinating things in cell biology is the fact that, although a multitude of various cell types exists in an organism, *all of them have the same*

DNA. The actual properties of each cell are therefore not defined by what proteins are *encoded* on its DNA, but rather by what proteins are being *produced* there. This, in turn, is determined by what regions of the DNA are *active*. And that is, in turn, influenced by what proteins are present in the cell. Thus, the life of a cell is governed by intricate relations and complex feedback loops. Studying the DNA and the relationships among the proteins may help to unwind these relations and therefore provide the keys to the inner workings of the cell.

In this thesis we address the problem of determining putative relations between DNA-binding proteins (*transcription factors*) and short sites on the DNA, where these proteins might bind themselves (*motifs*). Although current methods of molecular biology allow searching for binding sites of a given transcription factor experimentally, these experiments are generally rather expensive and would benefit from any hints given by computational evidence. Our method provides these hints, exploiting for this purpose the freely available *microarray* and *sequence* data.

Our approach is based on a simple linear model that naturally combines microarray measurements with DNA sequence data. The coefficients of this model, once estimated from data, can be used to infer putative relations between transcription factors and binding sites.

- We present the detailed specification and the biological motivation for the model in chapter 3.
- In chapter 4 we propose a number of methods for the estimation of model parameters.
- We have tested the performance of the model on a real biological dataset and on a specially crafted artificial dataset. The obtained results indicate that the model can be used to discover biologically relevant information and that the proposed parameter estimation methods are reliable. The experiments are documented in chapter 5.
- We illustrate various applications of the model for practical analysis of different kinds of data in chapter 6.
- For a broader distribution of the method we have created a proof-of-concept implementation of the estimation methods in Scilab and also a Web-tool that allows the users to acquaint themselves with our results as well as analyze their own datasets. The web-tool is briefly presented in Appendix A. The developed code is attached on a separate compact disc and is also available from the supplementary website [GMA].
- Before diving into the details, in chapters 1 and 2 we give a brief overview of the biological and mathematical basics that are needed to understand this work.

Chapter 1

Biological Background

People are DNA's way of making more DNA.

Edward O. Wilson

1.1 The DNA

The cell is the fundamental unit of any living organism. The smallest organisms, such as bacteria, might consist of a single cell, larger beings may contain millions of different cells organized in organs and tissues. Despite the small size, a cell is a very complex system on its own and is composed of yet smaller functional units called organelles. The functions of the organelles, as well as many other processes in the cell are mostly performed by *protein* molecules. Proteins vary greatly in their shapes, sizes and properties, yet they are all constructed in the same way: as linear chains of amino acids. The exact order of amino acids in the chain usually uniquely specifies any protein. This order is encoded in the strands of the *DNA molecules*, which form the *genetic material* of the cell.

The DNA (*deoxyribonucleic acid*) is a long spiral-shaped molecule consisting of two complementary strands of *nucleotides*. Each nucleotide consists of a *phosphate group*, sugar *deoxyribose* and a nitrogenous base: *adenine*, *thymine*, *cytosine* or *guanine*. Depending on the base, nucleotides are usually denoted by letters **A**, **T**, **C** and **G** respectively. A DNA strand can be therefore encoded as a string of these four letters. The second strand of the same molecule is uniquely defined by the first one: where the first strand has **A**, the second will have **T**, where the first one has **C**, the second will have **G** and vice-versa.

The meaning of the DNA sequence is not yet completely clear, but something is already known. Firstly, some parts of the DNA, known as the *coding regions* or *genes*, are used to encode proteins. The order of nucleotides in these regions determines precisely the order of the amino acids in the proteins of the cell. It

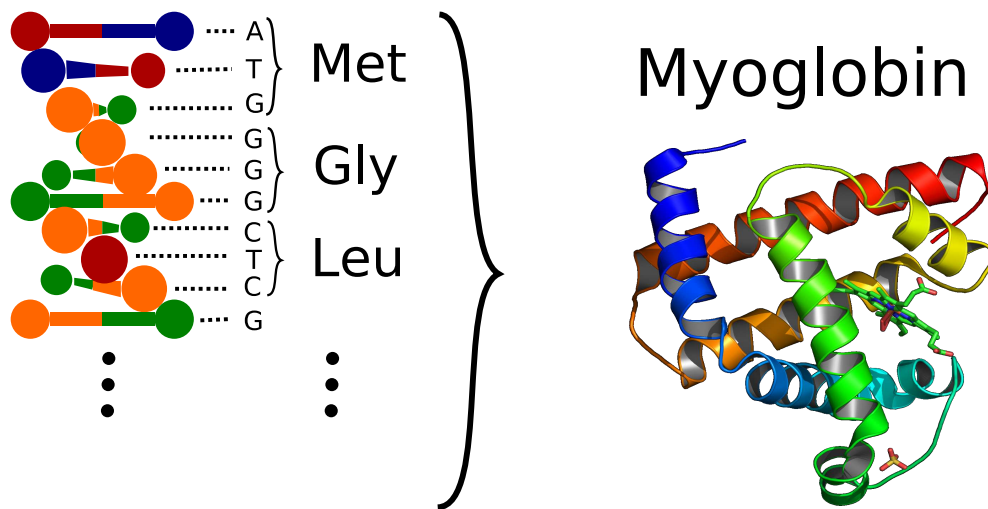


Figure 1.1: DNA consists of a spiral chain of nucleotides. In the gene-coding regions of the DNA, each triplet of nucleotides can be decoded into an amino acid. A chain of amino acids forms a protein.

is known that each amino acid is encoded by a certain triplet of nucleotides, and there are certain nucleotide triples that usually denote the start and the end of the coding region, e.g., **ATG** and **TAG**.

Besides the coding regions there are the *regulatory regions* on the DNA devoted to control mechanisms. These regions contain short sequences, often referred to as *motifs*, that can be *bound* by certain proteins (*transcription factors*). The act of binding most often has a regulatory role: it can open up a nearby coding region for processing thus activating a certain gene or, contrarily, close it down.

The DNA is believed to store the hereditary information of the whole organism. One of the most fascinating facts about cellular biology is that each cell in the organism has *exactly the same DNA*. Despite that, there is a multitude of radically different cell types: blood cells, skin cells and neurons in the brain seem as different as it can get. Therefore, the main functions of a cell are not determined simply by the genes on its DNA, but rather by what genes are being active in that cell. This explains why the questions of *gene expression regulation* comprise one of the major research directions in contemporary molecular biology.

In order to better understand the issue, we need to get acquainted with the process of protein production in the cell first.

1.2 Gene Expression Mechanisms

Proteins encoded on the DNA are produced in two stages: *transcription* and *translation*. In the process of transcription a large molecular complex called *RNA-*

polymerase attaches itself to the segment of DNA containing a certain *promoter sequence* and starts copying DNA from that point producing an equivalent *RNA* molecule. *RNA* (*ribonucleic acid*) is very similar to DNA: it also consists of a chain of nucleotides. However, it is single-stranded, uses the sugar *ribose* in place of DNA's deoxyribose, and base *uracil* instead of thymine. Contrary to the DNA, the RNA molecules are much less stable and get decomposed (*hydrolyzed*) in the cell within minutes. This makes it possible to use them as temporary information carriers between the DNA “data storage” in the nucleus and the main protein-producing machinery outside.

The transcription process stops when the RNA-polymerase encounters a certain *terminator sequence* (also called *stop codon*). It then releases an RNA encoding exactly the same information as that on the corresponding coding region of the DNA. Next, the obtained RNA undergoes *splicing* – a process, during which parts of the RNA chain are removed with the help of some complex molecular machinery. Finally, an RNA strand ready to be translated to proteins is produced. This RNA is known as *mRNA* (short for *messenger-RNA*).

Another large molecular complex, *ribosome*, attaches itself to the mRNA and starts reading it, producing the corresponding protein. This process is known as *translation*.

Altogether, the production of proteins from genes is referred to as *gene expression*. As already mentioned before, not all genes of the genome are expressed in a cell at once. The regulation of which gene will be expressed and which will not is a complex process, not yet completely understood by contemporary biology. By what is known, an important role in expression regulation is played by certain proteins, called *transcription factors* or *TFs*. These proteins attach themselves to *motifs* on the DNA and provoke (or prevent) RNA-polymerase to bind the DNA at these points. This can induce (or suppress) the transcription of the corresponding gene. Sometimes, the transcription regulation patterns can be more complex: certain TFs will attach to the DNA only in presence of some other TFs, and certain TFs, when attached, may block other TFs from binding there.

Transcription regulation is not the only part of expression regulation. After being transcribed to mRNA, the gene will be translated to proteins only under certain circumstances, when all the machinery (i.e., proteins) needed for splicing and translation is available and, depending on the outcome of splicing, the mRNA of the gene may be translated to one or another protein.

In general, the production of any given protein by the cell depends on the amount of other proteins as well as on environmental conditions, such as temperature, humidity, acidity, etc. The relationships among proteins that govern gene expression are often called *genetic networks*, and the problem of determining them is crucial to explaining the life cycle of the cell.

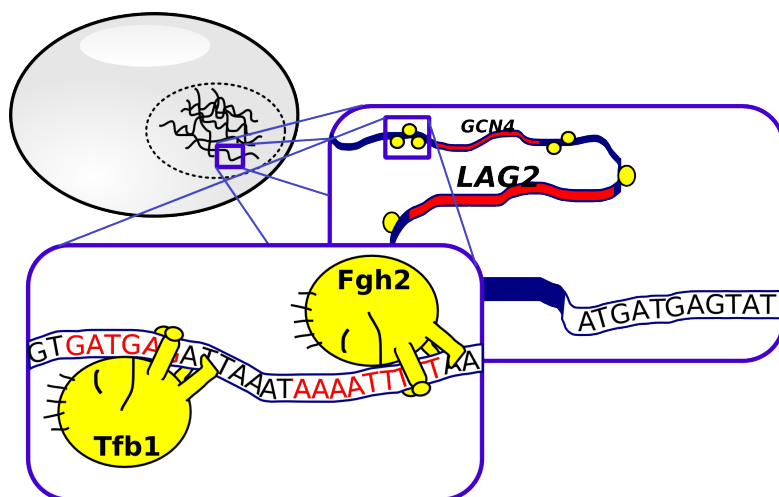


Figure 1.2: The figure depicts strands of DNA in the nucleus of the cell (on the top left), genes (red regions in the middle window), and the nearby regulatory regions together with *transcription factors* (yellow blobs) bound to *motifs*.

1.3 Microarray Data

The *microarray* technology provides efficient methods for measuring expression of many genes at once. A typical microarray experiment goes as follows. First, an array is prepared, which consists of a glass or membrane support with a set of short DNA fragments attached to it. The fragments are carefully arranged in *spots*: each spot contains the fragments of a certain gene only. Next, two cell cultures are selected: a “reference” culture, and a culture of cells under certain stress or environmental conditions (“test” culture). All the mRNA from these cells is reverse-transcribed to create strands of complementary DNA (*cDNA*). The strands are then labeled with fluorescent or radioactive dyes. Labeling of the cDNA strands differs for the two cultures: the reference culture is commonly labeled with a green dye, and the test culture – with a red one. The obtained cDNA is then purged onto the array where it reacts (*hybridizes*) with the DNA in the spots of the array. As a result, all the cDNAs should end up on the spots with matching DNA fragments. Due to the labeling of the cDNA, the color (or radiation spectrum) of each spot will indicate the ratio of the cDNA amounts from the first and the second culture attached to it, hence the ratio of mRNA amounts of the corresponding gene produced in the different cultures. The amount of mRNA for a gene may be regarded as an indicator of the expression of that gene.

Of course, as noted above, transcription is not the only aspect of gene expression, so this is a rather rough indicator indeed.

After the hybridization has been performed, the glass arrays are scanned, the color of the spots recognized and converted to a single real number for each gene

– its *expression value*. Data from several arrays is usually represented as a matrix with rows corresponding to genes and columns to arrays. Each entry specifies the expression of a gene in a given array. Although microarray experiments can be rather costly, every experiment measures the expression levels of thousands of genes at the same time. Therefore, it is typical for the resulting matrix to have thousands of rows, but only about a hundred or so columns. It is this *expression matrix*, that is in the focus of this thesis.

The most common way of exploiting microarray data currently is to look for genes that have similar *expression profiles*, i.e., tend to have similar expression values over many experiments. This *expression similarity* can often be attributed to functional relation of the genes. In this thesis we go a step further and incorporate genomic sequence in the analysis. We consider expression similarity of genes having similar *motifs* in their *promoter regions*.

1.4 DNA Sequence Data

Due to the efficiency of contemporary DNA sequencing technologies, the genomes of many different organisms have been sequenced in their entirety and are freely available for processing. In the practical part of this thesis we focus on the genome of baker's yeast, *Saccharomyces cerevisiae*, which is distributed on the *Saccharomyces Genome Database* website [SGD]. This genome is well studied and most of putative coding regions in it are detected and annotated. For each gene-coding region, the sequence of 600-1000 nucleotides immediately preceding it (the so-called *primary promoter*) is known to comprise the major regulatory region for that gene. Note that things are often not as simple in more complex organisms.

We are mostly interested in the presence of known *motifs* in the promoters. Several databases of known motifs are available, the most famous of them are *JASPAR* [BVT⁺08] and *TRANSFAC* [WDKK96]. In this work we use the TRANSFAC database version 10.3 [WDKK96, MFG⁺03]. The motifs in TRANSFAC are typically represented in the form of *Position Weight Matrices (PWMs)*. For a reader not acquainted with this term it should be enough to know that one can search for “occurrences” (or “matches”) of a PWM in a given sequence in approximately the same manner as one would search for the occurrences of a substring. PWM is different in that it allows to have mismatching characters in a certain way. In our experiments we use the **storm** tool [SSZ07] to determine, for each given gene and each motif, whether the gene has a match of that motif in its promoter or not.

1.5 Gene Ontology Annotation Data

The analysis method presented in this thesis can be easily generalized to other kinds of data besides microarray measurements and DNA sequences. In particular, we attempt to make use of the *Gene Ontology* annotations. *Gene Ontology (GO)* [ABB⁺00] is a controlled vocabulary of terms, describing function, localization and processes that any gene might take part of. The vocabulary allows to manage existing information about the genes in a simple and concise form of *annotations*. An annotation specifies for a gene that it is known to be associated with a certain function or process. The *Saccharomyces Genome Database* website has collected known annotations for yeast genes. We show the use of these annotations in our experiments with quite fruitful results.

Chapter 2

Mathematical Background

The Great Book of Nature is written in the language of mathematics.

Galileo Galilei

This chapter introduces the general mathematical and statistical notions used further in the text. Only the most basic ideas are presented, avoiding detailed explanations, examples and proofs. For a more thorough introduction on the topic the reader is referred to textbooks on the subject, such as [Mey01, GS97, HS03, MS06]. Readers familiar with the topic can safely skip this chapter and refer back to it only when necessary.

2.1 Matrix Algebra

Most of the math that follows is expressed in terms of real-valued vectors and matrices. The notation that we use is presented in Table 2.1.

Recollect that an n -element vector \mathbf{v} is just an array of n real numbers, and that an $n \times m$ matrix \mathbf{M} is a table of real numbers with n rows and m columns. An n -element vector \mathbf{v} can also be regarded as a $n \times 1$ matrix.

Matrix operations. Two equally-sized matrices \mathbf{A} and \mathbf{B} can be added together to obtain their *sum* $\mathbf{C} = \mathbf{A} + \mathbf{B}$ in the following manner:

$$(\mathbf{C})_{ij} = (\mathbf{A})_{ij} + (\mathbf{B})_{ij}.$$

If \mathbf{A} is an $n \times m$ matrix and \mathbf{B} is an $m \times \ell$ matrix, these matrices can be *multiplied* together. The resulting $n \times \ell$ matrix $\mathbf{C} = \mathbf{AB}$ is defined as follows:

$$(\mathbf{C})_{ij} = \sum_{k=1}^m (\mathbf{A})_{ik} (\mathbf{B})_{kj} \tag{2.1}$$

Notion	Notation
Vectors	Boldface lowercase letters: $\mathbf{a}, \mathbf{b}, \mathbf{c}, \dots$
Vector elements	Subscripts: $(\mathbf{a})_i$ or a_i
Matrices	Boldface capital letters: $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$
Matrix elements	Subscripts: $(\mathbf{M})_{ij}$ or $(\mathbf{M})_{i,j}$ or M_{ij} or $M_{i,j}$
Matrix rows	Superscripts: $\mathbf{M}^{(i*)}$
Matrix columns	Superscripts: $\mathbf{M}^{(*j)}$
Matrix transpose	\mathbf{M}^T
Matrix inverse	\mathbf{M}^{-1}
Moore-Penrose pseudoinverse	\mathbf{M}^+
ℓ_2 norm	$\ \mathbf{M}\ $
Matrix rank	$\text{rank}(\mathbf{M})$
Identity matrix	\mathbf{I}

Table 2.1: Matrix notation.

The *transpose* of an $n \times m$ matrix \mathbf{M} is an $m \times n$ matrix that is obtained by switching its rows for columns and vice-versa:

$$(\mathbf{M}^T)_{ij} = (\mathbf{M})_{ji}.$$

For any two matrices \mathbf{A} and \mathbf{B} that can be multiplied together it holds that

$$(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T.$$

Matrix rank. Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ be m -dimensional vectors. These vectors are said to be *linearly independent*, if, for any $\lambda_1, \lambda_2, \dots, \lambda_n \in \mathbb{R}$:

$$\sum_{i=1}^n \lambda_i \mathbf{v}_i = \mathbf{0} \Leftrightarrow \lambda_1 = \lambda_2 = \dots = \lambda_n = 0.$$

The *rank* of a matrix $\text{rank}(\mathbf{M})$ is defined as the maximum number of linearly independent columns of \mathbf{M} . In simple terms, rank measures the “complexity” of a matrix. In the main part of the text we shall use the following facts about rank:

$$\text{rank}(\mathbf{M}) = \text{rank}(\mathbf{M}^T \mathbf{M}) = \text{rank}(\mathbf{M} \mathbf{M}^T), \text{ for any matrix } \mathbf{M}, \quad (2.2)$$

$$\text{rank}(\mathbf{M} \mathbf{M}^T + \lambda \mathbf{I}) = n, \text{ for any } n \times m \text{ matrix } \mathbf{M} \text{ and any } \lambda > 0. \quad (2.3)$$

Matrix inverse. The $n \times n$ *identity matrix* \mathbf{I} is defined as a matrix with ones on the diagonal and zeros elsewhere, that is:

$$(\mathbf{I})_{ij} = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{otherwise.} \end{cases}$$

For a given $n \times n$ matrix \mathbf{M} , its *inverse* \mathbf{M}^{-1} is defined as a matrix, satisfying

$$\mathbf{M}\mathbf{M}^{-1} = \mathbf{M}^{-1}\mathbf{M} = \mathbf{I}.$$

Not every square matrix has an inverse. A matrix that has an inverse is called *invertible*. For any two invertible matrices \mathbf{A} and \mathbf{B} it holds that

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}.$$

The rank of an $n \times n$ invertible matrix is n and vice versa:

$$\text{rank}(\mathbf{M}) = n \Leftrightarrow \mathbf{M} \text{ is invertible.} \quad (2.4)$$

Matrix ℓ_2 -norm. We define the ℓ_2 -norm of a matrix \mathbf{M} as the square root of the sum of squares of its elements:

$$\|\mathbf{M}\| = \sqrt{\sum_{i,j} M_{ij}^2}.$$

Orthogonal matrices. A square matrix \mathbf{M} is said to be *orthogonal* if $\mathbf{M}^{-1} = \mathbf{M}^T$. In other words, the matrix \mathbf{M} is orthogonal iff

$$\mathbf{M}^T\mathbf{M} = \mathbf{M}\mathbf{M}^T = \mathbf{I}.$$

Multiplication of a matrix by an orthogonal matrix does not change the ℓ_2 -norm of that matrix. That is, if \mathbf{U} is orthogonal:

$$\|\mathbf{M}\| = \|\mathbf{UM}\| = \|\mathbf{MU}\|, \quad (2.5)$$

for any \mathbf{M} of proper dimensions.

Singular value decomposition. For any $n \times m$ matrix \mathbf{M} there exist matrices \mathbf{U} , \mathbf{D} , \mathbf{V} , such that

$$\mathbf{M} = \mathbf{UDV}^T, \quad (2.6)$$

where \mathbf{U} is a $n \times n$ orthogonal matrix, \mathbf{V} is a $m \times m$ orthogonal matrix, and \mathbf{D} is an $n \times m$ matrix with nonnegative numbers on the diagonal and zeros off the

diagonal. That is, $D_{ij} = 0$ for $i \neq j$ and $D_{ii} \geq 0$. The elements $d_i = D_{ii}$ are called the *singular values* of \mathbf{M} . The number of nonzero singular values is equal to the rank of the matrix. The decomposition (2.6) is called the *singular value decomposition* (*SVD decomposition*) of \mathbf{M} .

Moore-Penrose pseudoinverse. Let $\mathbf{M} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ be the singular value decomposition of \mathbf{M} . The *Moore-Penrose pseudoinverse* of \mathbf{M} is a matrix \mathbf{M}^+ that is defined as follows:

$$\mathbf{M}^+ = \mathbf{V}\mathbf{D}^+\mathbf{U}^T, \quad (2.7)$$

where

$$(\mathbf{D}^+)_{ij} = \begin{cases} 1/D_{ii}, & \text{if } i = j \text{ and } D_{ii} > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (2.8)$$

Any matrix has a uniquely defined Moore-Penrose pseudoinverse and if a matrix is invertible, $\mathbf{M}^+ = \mathbf{M}^{-1}$.

The Moore-Penrose pseudoinverse can also be defined as a matrix satisfying the following conditions:

$$\mathbf{M}\mathbf{M}^+\mathbf{M} = \mathbf{M}, \quad (2.9)$$

$$\mathbf{M}^+\mathbf{M}\mathbf{M}^+ = \mathbf{M}^+, \quad (2.10)$$

$$\mathbf{M}^+\mathbf{M} = (\mathbf{M}^+\mathbf{M})^T, \quad (2.11)$$

$$\mathbf{M}\mathbf{M}^+ = (\mathbf{M}\mathbf{M}^+)^T. \quad (2.12)$$

In addition, in the main text we refer to the following particular properties of the pseudoinverse, that can be easily derived from the conditions above:

$$\mathbf{M}^T\mathbf{M}\mathbf{M}^+ = \mathbf{M}^T \quad (2.13)$$

$$\mathbf{M}^+\mathbf{M}\mathbf{M}^T = \mathbf{M}^T \quad (2.14)$$

$$\mathbf{M}^+(\mathbf{M}^+)^T\mathbf{M}^T = \mathbf{M}^+ \quad (2.15)$$

$$\mathbf{M}^T(\mathbf{M}^+)^T\mathbf{M}^+ = \mathbf{M}^+ \quad (2.16)$$

2.2 Probability Theory

A work on data analysis cannot do without basic probability theory. Table 2.2 presents the notation used in this work.

Random variables. The notion of a random variable is central to probability theory. Avoiding formalities, we can regard a real-valued random variable \mathbf{X} as a real-valued variable, the exact value of which is uncertain. A typical example of a random variable is the number of points on a die yet to be thrown.

Notion	Notation
Random variables	Serif capital letters: A, B, C, \dots
Probability	For example: $\Pr[X > 0]$
Mean	\bar{X} or $E(X)$
Variance	$D(X)$
Covariance	$\text{cov}(X, Y)$
Almost sure convergence	$X_n \xrightarrow{\text{a.s.}} Y$
Sample	$x \leftarrow X$
Normal distribution	$N(m, \sigma^2)$
Bernoulli distribution	$B(p)$
Beta distribution	$\text{Beta}(a, b)$

Table 2.2: Probability theory notation.

The uncertainty of a random variable is not absolute, though. For any potential value x , we have an idea of how *probable* it is that $X > x$, $X = 0$ or $X < x$. We can also answer more complex questions, such as “what is the probability that $3 < X < 5$?”. This knowledge about X is most conveniently formalized using a *cumulative distribution function (CDF)*:

$$F_X(x) = \Pr[X \leq x].$$

To indicate that a variable X has cumulative distribution F_X we write $X \sim F_X$ and often say simply that “ X has distribution F_X ”. A distribution function can be defined similarly for two or more variables:

$$F_{X,Y}(x, y) = \Pr[X \leq x \text{ and } Y \leq y].$$

In this case we write $(X, Y) \sim F_{X,Y}$. A number of standard distribution functions exist, such as the Bernoulli distribution, normal distribution, beta distribution, etc. Each of these distributions describes the uncertainty of a random variable in its own specific manner.

Random variables can be added, multiplied and divided just as usual real numbers, the only difference is that the result is also a random variable. For example, if X denotes the number of points on one die and Y – the points on a second die, then $X + Y$ would be the total number of points on the two dice and $2X$ would be twice the number of points on the first die.

Mean, variance, covariance. To avoid excessive formalities, we only give an informal definition here. The *mean* of a random variable X is the value that we expect this variable to have “on average”. Let X denote the number of points on

a die. The mean of this random variable is 3.5, because this is the average of the equally probable outcomes of a single throw: 1, 2, 3, 4, 5 or 6. We denote the mean of X as $E(X)$ or \bar{X} .

A useful property of mean is *linearity*. For any $a \in \mathbb{R}$ and random variables X and Y :

$$E(aX + Y) = a E(X) + E(Y) .$$

The *variance* of a random variable is defined as the mean squared deviation of this variable from its mean:

$$D(X) = E((X - \bar{X})^2) .$$

The *covariance* of two variables X and Y is defined as

$$\text{cov}(X, Y) = E((X - \bar{X})(Y - \bar{Y})) .$$

Variance and covariance satisfy the following properties:

$$D(X) = E(X^2) - E(X)^2 , \tag{2.17}$$

$$D(X) = \text{cov}(X, X) , \tag{2.18}$$

$$\text{cov}(X, Y) = E(XY) - E(X) E(Y) , \tag{2.19}$$

$$\text{cov}(X, Y) = \text{cov}(Y, X) , \tag{2.20}$$

$$\text{cov}(aX + Y, Z) = a \text{cov}(X, Z) + \text{cov}(Y, Z) . \tag{2.21}$$

Random variables X and Y are said to be *uncorrelated*, if $\text{cov}(X, Y) = 0$. If X and Y are uncorrelated, then

$$E(XY) = E(X) E(Y) .$$

Independence. We say that random variables X and Y are *independent*, if:

$$F_{X,Y}(x, y) = F_X(x)F_Y(y) .$$

Informally, independence means that observing the value of one of the variables does not provide any knowledge about the other one. For example, two separate die throws are independent, because knowing the outcome of the first throw does not tell anything about the outcome of the second. However, the sum of two throws is not independent of the first throw: if we know that the first throw produced a small number we would expect the sum to be smaller than usual, too. Independent random variables are necessarily uncorrelated.

Sample. Let X be a random variable with distribution F_X . If we get to observe its value x , we say that we obtain a *realization* of this random variable and write

it as:

$$x \leftarrow \mathbf{X} \quad \text{or} \quad x \leftarrow F_{\mathbf{X}}.$$

Let $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n$ be independent random variables, each with distribution $F_{\mathbf{X}}$. Realizations (x_1, x_2, \dots, x_n) of these random variables form an *i.i.d.* (*independent, identically distributed*) sample of $F_{\mathbf{X}}$.

Almost sure convergence. Let $a \in \mathbb{R}$ and let $(\mathbf{X}_n)_{n \in \mathbb{N}}$ be a sequence of random variables. We say that this sequence *converges almost surely* to a and we write $\mathbf{X}_n \xrightarrow{\text{a.s.}} a$, if

$$\Pr \left[\lim_{n \rightarrow \infty} \mathbf{X}_n = a \right] = 1.$$

In other words, nearly any realization of the sequence \mathbf{X}_n converges to a .

Strong law of large numbers. Let $(\mathbf{X}_n)_{n \in \mathbb{N}}$, be a sequence of independent, identically distributed random variables with distribution $F_{\mathbf{X}}$. Let the mean of this distribution be m . The *strong law of large numbers* states that

$$\frac{1}{n} \sum_{i=1}^n \mathbf{X}_i \xrightarrow{\text{a.s.}} m.$$

That means, if $(x_1, x_2, \dots, x_n) \leftarrow F_{\mathbf{X}}$ is an i.i.d. sample of $F_{\mathbf{X}}$, the sample mean will converge to $E(\mathbf{X})$ with probability 1.

Empirical cumulative distribution function. Let (x_1, x_2, \dots, x_n) be an i.i.d. sample of distribution $F_{\mathbf{X}}$. We can use this sample to construct an approximation to the distribution function $F_{\mathbf{X}}$. Namely, let

$$F_n^{\text{emp}}(x) = \frac{\#\{i : x_i \leq x\}}{n},$$

where $\#\{\cdot\}$ denotes the number of elements in a set. The function F_n^{emp} is called the *empirical cumulative distribution function (CDF)* of a given sample. By the law of large numbers, as the sample size increases this function converges to the true distribution function $F_{\mathbf{X}}$:

$$F_n^{\text{emp}}(x) \xrightarrow{\text{a.s.}} F_{\mathbf{X}}(x), \quad \text{for any } x.$$

2.3 Optimization Theory

Optimization theory deals with the task of finding the minima and maxima of functions. In this work, we only use the most basic notion of a partial derivative.

We denote the partial derivative of a function f with respect to a variable x as:

$$\frac{\partial f(x)}{\partial x} \quad \text{or} \quad \frac{\partial}{\partial x} f(x) \quad \text{or} \quad \frac{\partial f}{\partial x}(x),$$

If the parameter of f is a vector \mathbf{v} or a matrix \mathbf{M} , we use the notation such as:

$$\frac{\partial f(\mathbf{v})}{\partial \mathbf{v}} \quad \text{or} \quad \frac{\partial f(\mathbf{M})}{\partial \mathbf{M}},$$

to denote the vector or matrix of corresponding partial derivatives:

$$\left(\frac{\partial f(\mathbf{v})}{\partial \mathbf{v}} \right)_i = \frac{\partial f(\mathbf{v})}{\partial v_i}, \quad \left(\frac{\partial f(\mathbf{M})}{\partial \mathbf{M}} \right)_{ij} = \frac{\partial f(\mathbf{M})}{\partial M_{ij}}.$$

Differentiability. Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a multivariate function. We say that f is *differentiable at point \mathbf{x}^** , if there exists a vector $\mathbf{a}(\mathbf{x}^*)$, such that

$$f(\mathbf{x}^* + \Delta \mathbf{x}) = f(\mathbf{x}^*) + \mathbf{a}(\mathbf{x}^*)^T \Delta \mathbf{x} + o(\Delta \mathbf{x}).$$

We say that function f is *differentiable* if it is differentiable for any $\mathbf{x}^* \in \mathbb{R}$. The vector $\mathbf{a}(\mathbf{x}^*)$ is referred to as the *gradient* of f (at point \mathbf{x}^*), and we denote it by $\frac{\partial f(\mathbf{x}^*)}{\partial \mathbf{x}}$. The elements of this vector are the (univariate) partial derivatives of f , which, we hope, are familiar to the reader from basic calculus.

$$\left(\frac{\partial f(\mathbf{x}^*)}{\partial \mathbf{x}} \right)_i = \frac{\partial f(\mathbf{x}^*)}{\partial x_i} = \frac{\partial f(x_1^*, x_2^*, \dots, x_n^*)}{\partial x_i}.$$

The following well-known properties of gradient and partial derivatives are used in this work:

$$\frac{\partial a}{\partial x} = 0, \text{ if } a \text{ is a constant,} \quad (2.22)$$

$$\frac{\partial ax}{\partial x} = a, \quad (2.23)$$

$$\frac{\partial x^2}{\partial x} = 2x, \quad (2.24)$$

$$\frac{\partial af(\mathbf{x}) + g(\mathbf{x})}{\partial \mathbf{x}} = a \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} + \frac{\partial g(\mathbf{x})}{\partial \mathbf{x}}, \quad (2.25)$$

$$\frac{\partial f(g(\mathbf{x}))}{\partial \mathbf{x}} = \frac{\partial f}{\partial \mathbf{x}}(g(\mathbf{x})) \cdot \frac{\partial g(\mathbf{x})}{\partial \mathbf{x}}. \quad (2.26)$$

Convexity. The function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is called *convex* (or, sometimes, *convex-cup*), if for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}$ and $\lambda \in (0, 1)$:

$$f(\lambda \mathbf{x} + (1 - \lambda) \mathbf{y}) \leq \lambda f(\mathbf{x}) + (1 - \lambda) f(\mathbf{y}).$$

In this work, we use the fact that the function

$$f(\mathbf{x}) = \|\mathbf{Ax} + \mathbf{b}\|^2 \quad (2.27)$$

is convex and differentiable for any $m \times n$ matrix \mathbf{A} and m -element vector \mathbf{b} .

Minimum of a convex function. Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a multivariate function. We say that the point \mathbf{x}^* is a *global minimum* of that function, and write

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} f(\mathbf{x}),$$

when

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \text{ for any } \mathbf{x} \in \mathbb{R}^n.$$

If the function f is convex and differentiable, then a point \mathbf{x}^* is a global minimum of f if and only if

$$\frac{\partial f(\mathbf{x}^*)}{\partial \mathbf{x}^*} = \mathbf{0}. \quad (2.28)$$

2.4 Linear Models

The theory of linear models is, perhaps, one of the most widely treated branches of statistics. Here we present only the basic linear model in its simplest form. Let $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_m$ and \mathbf{Y} be random variables. We say that these variables form a *linear model*, if they satisfy the relation

$$\mathbf{Y} = \beta_1 \mathbf{X}_1 + \beta_2 \mathbf{X}_2 + \dots + \beta_m \mathbf{X}_m + \varepsilon, \quad (2.29)$$

where $\beta_1, \beta_2, \dots, \beta_m \in \mathbb{R}$ are the *model coefficients* and ε is a random variable representing noise. The variable ε is defined to be independent of all \mathbf{X}_k and has normal distribution with zero mean and variance σ^2 :

$$\varepsilon \sim \mathcal{N}(0, \sigma^2).$$

We shall be interested in the task of estimating the values of the parameters β_k from the observed values of the variables \mathbf{Y} and \mathbf{X}_k . Let $i \in \{1, 2, \dots, n\}$ and for each i let $(x_{i1}, x_{i2}, \dots, x_{im}, y_i, \varepsilon_i)$ be an independent realization of $(\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_m, \mathbf{Y}, \varepsilon)$. As the realizations come from a linear model, they must also satisfy equation (2.29):

$$y_i = \sum_{k=1}^m \beta_k x_{ik} + \varepsilon_i, \text{ for all } i \in \{1, 2, \dots, n\}. \quad (2.30)$$

Collect all the values (x_{ij}) into an $n \times m$ matrix \mathbf{X} . This matrix is often referred

to as the *data matrix*. Collect all the values (y_i) into an n -element vector \mathbf{y} and all the values (ε_i) into an n -element vector $\boldsymbol{\varepsilon}$. Finally, let $\boldsymbol{\beta}$ be the vector of parameters (β_k) . Then equation (2.30) can be conveniently expressed in matrix form:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}. \quad (2.31)$$

In practice, we only observe the matrices \mathbf{y} and \mathbf{X} and we are interested in estimating the parameters $\boldsymbol{\beta}$. There are typically two different reasons why this task may be important.

Predictive modeling. It is customary to regard equation (2.29) as a *predictive model*. In this case the variables X_k are referred to as the *predictor variables* and Y as the *output variable*. If we know the model parameters $\boldsymbol{\beta}$, then for any instantiation $\mathbf{x} = (x_1, x_2, \dots, x_m)^T$ of the predictor variables we can predict the output \hat{y} as:

$$\hat{y} = \sum_{k=1}^m \beta_k x_k = \boldsymbol{\beta}^T \mathbf{x}.$$

Such a prediction \hat{y} will have an error ε , but we typically expect this error to be tolerably small. In a practical setting, we do not know the model parameters, but we have a dataset (\mathbf{X}, \mathbf{y}) of “past observations”. We can use this dataset to obtain an estimate for the parameters $\hat{\boldsymbol{\beta}}$ and use them to predict the output \hat{y} for any forthcoming “future observations” \mathbf{x} .

Descriptive analysis. Sometimes, we do not really need a model for making predictions, but the values of the parameters $\boldsymbol{\beta}$ themselves have a meaningful interpretation. The conceptual difference from predictive modeling lies in the fact that we do not expect any future observations. We just search for the parameters $\boldsymbol{\beta}$, that model the linear relation between \mathbf{X} and \mathbf{y} and thus explain or describe the data. This is the way in which we apply linear modeling in this work.

2.4.1 Model Parameter Estimation

Least squares. Perhaps the simplest and the most natural method of parameter estimation for a linear model is the *least squares method*. The idea is to find $\hat{\boldsymbol{\beta}}$ so that the mean squared error of the corresponding model predictions \hat{y}_i would be as small as possible. In formal terms:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2, \quad (2.32)$$

where

$$\hat{y}_i = \sum_{k=1}^m \beta_k x_{ik}$$

is the model prediction for instance i . Equation (2.32) can be rewritten in matrix form as:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2. \quad (2.33)$$

We can also drop the constant $\frac{1}{n}$, as it does not influence the minimum. As we know, the function $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2$ is convex and differentiable. Therefore, we can use the condition (2.28) to find its minimum:

$$\frac{\partial}{\partial \boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 = \mathbf{0}. \quad (2.34)$$

The latter can be simplified using (2.25), (2.26) and (2.24):

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 &= \frac{\partial}{\partial \boldsymbol{\beta}} \sum_{i=1}^n (y_i - (\mathbf{X}\boldsymbol{\beta})_i)^2 = \sum_{i=1}^n \frac{\partial}{\partial \boldsymbol{\beta}} (y_i - (\mathbf{X}\boldsymbol{\beta})_i)^2 \\ &= \sum_{i=1}^n 2(y_i - (\mathbf{X}\boldsymbol{\beta})_i) \frac{\partial (y_i - (\mathbf{X}\boldsymbol{\beta})_i)}{\partial \boldsymbol{\beta}} \end{aligned} \quad (2.35)$$

The partial derivatives can be computed as follows:

$$\begin{aligned} \frac{\partial (y_i - (\mathbf{X}\boldsymbol{\beta})_i)}{\partial \beta_\kappa} &= \frac{\partial y_i}{\partial \beta_\kappa} - \frac{\partial (\mathbf{X}\boldsymbol{\beta})_i}{\partial \beta_\kappa} = -\frac{\partial (\mathbf{X}\boldsymbol{\beta})_i}{\partial \beta_\kappa} = -\frac{\partial \sum_{k=1}^m x_{ik} \beta_k}{\partial \beta_\kappa} \\ &= -\sum_{k=1}^m \frac{\partial x_{ik} \beta_k}{\partial \beta_\kappa} = -\sum_{k=1}^m x_{ik} \frac{\partial \beta_k}{\partial \beta_\kappa} = -x_{i\kappa}. \end{aligned}$$

By substituting this into (2.35), we get:

$$\begin{aligned} \frac{\partial}{\partial \beta_\kappa} \|\mathbf{y} - \mathbf{X}\mathbf{b}\|^2 &= \sum_{i=1}^n 2(y_i - (\mathbf{X}\boldsymbol{\beta})_i)(-x_{i\kappa}) = -2 \sum_{i=1}^n (y_i - (\mathbf{X}\boldsymbol{\beta})_i) x_{i\kappa} \\ &= -2(\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}))_\kappa. \end{aligned}$$

Therefore, equation (2.34) takes the matrix form:

$$-2(\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})) = \mathbf{0},$$

which can be simplified to

$$\mathbf{X}^T \mathbf{X} \boldsymbol{\beta} = \mathbf{X}^T \mathbf{y}.$$

If $\mathbf{X}^T \mathbf{X}$ is invertible, the solution to that equation is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}.$$

Otherwise, multiple solutions exist. All of them can be expressed as

$$\hat{\boldsymbol{\beta}} = \mathbf{X}^+ \mathbf{y} + (\mathbf{X}^+ \mathbf{X} - \mathbf{I}) \mathbf{k},$$

where \mathbf{k} is any m -element vector and \mathbf{X}^+ denotes the Moore-Penrose pseudoinverse of \mathbf{X} .

Ridge regression. Although the least squares solution provides the best fit to the data, it is sometimes better to find a solution that not only fits the data well, but also has reasonably small parameter values. It can be achieved by penalizing large $\boldsymbol{\beta}$ in the following manner:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|^2,$$

where $\lambda > 0$ is the so-called *regularization parameter*. The solution to this problem is:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}.$$

This method is commonly referred to as ℓ_2 -norm regularized regression or *ridge regression*.

2.4.2 Model Assessment

Various metrics exist for the evaluation of the quality of an estimated model. In this work we use three particular metrics: *mean squared error*, *coefficient of determination* and *ROC AUC score*.

Mean squared error. The mean squared error of the model is simply the mean of the squares of prediction errors:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2, \quad \text{where } \hat{y}_i = \sum_{k=1}^m \hat{\beta}_k x_{ik}.$$

Coefficient of determination. Let S^2 be the variance of the output, that is

$$S^2 = \frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})^2 \quad \text{where } \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i.$$

The coefficient of determination of a model, often denoted as R^2 , is defined as follows:

$$R^2 = 1 - \frac{\text{MSE}}{S^2}.$$

The coefficient of determination rescales the mean squared error by the variance of the data and thus measures the fraction of variance that is explained by the model. The value of $R^2 = 1$ means that the model is a perfect predictor, its error is 0. The value of $R^2 = 0$ means the model is no better at prediction than a constant predictor $\hat{y} = \bar{y}$.

ROC AUC score. The *Receiver Operating Characteristic Area Under Curve* (ROC AUC) statistic is rather unrelated to linear models in general, but we use it in the main text to evaluate the ranking of the model parameters. Let $\{a_1, a_2, \dots, a_n\}$ be a set of some objects and let there be a binary variable y_i associated with each object a_i . Those objects, for which the corresponding $y_i = 1$ are called *positives*, and those, for which $y_i = 0$ are called *negatives*. Suppose, some system assigns a rank r_i to every object a_i , aiming to assign higher ranks to positive instances and lower ranks to negative instances. The ROC AUC score is a measure of goodness for such a system. We define ROC AUC score as the probability that the rank of a randomly chosen positive instance is higher than the rank of a randomly chosen negative one:

$$\text{ROC} = \Pr[r_i > r_j \mid y_i = 1, y_j = 0].$$

The value of ROC AUC equal to 1 indicates a perfect ranking, where all positive instances get the highest ranks. The value of ROC AUC equal to 0.5 indicates the worst possible ranking that is not better than random guessing. Finally, the value of ROC AUC equal to 0 indicates a reversed perfect ranking, where all positive instances get the lowest ranks. The ROC AUC statistic can also be interpreted as the area under a certain curve, but in this work we do not need this interpretation.

Chapter 3

The G=MAT Model

All models are wrong, but some of them are useful.

George Box

Biological processes underlying the expression of genes in the cell are complex and not yet completely understood. One of the best studied mechanisms of expression regulation is the regulation of transcription by *transcription factors* (*TFs*), that bind themselves to short *motifs* in the promoter regions of genes. Vast amounts of *microarray data* on gene expression, as well as the availability of complete *genome sequences* make this problem especially suitable for bioinformatical analysis.

In this work we consider the task of determining, for a given set of TFs and motifs, which TFs might bind to which motifs. Although current methods of molecular biology allow to search for binding sites of a given TF experimentally [HS02], these experiments are generally rather expensive and would benefit greatly from any hints given by external computational evidence. Our method provides these hints by highlighting pairs of TFs and motifs that seem to have strong influence on gene expression.

3.1 Notation

Genes and TFs. Consider all genes in an organism. We divide them in two (non-overlapping) classes: TFs and target genes.

- The first class contains genes, that correspond to actual or putative transcription factors. We call these genes *TFs* (or *predictor genes*) and denote them as t_k , $k \in \{1, 2, \dots, n_T\}$ where n_T denotes the number of TFs.
- The second class contains all other, non-TF genes. We refer to them as *target genes* or simply *genes*, and denote them as g_i , $i \in \{1, 2, \dots, n_G\}$ where n_G

is the number of target genes.

Microarray Data. Next, consider a set of microarray experiments. We denote each experiment as a_j , $j \in \{1, 2, \dots, n_A\}$.

- Let \mathbf{G} be the $n_G \times n_A$ *gene expression matrix*. That is, the value G_{ij} denotes the expression of (target) gene g_i in the experiment a_j .
- Similarly, let \mathbf{T} be the $n_T \times n_A$ *TF expression matrix*: the value T_{kj} denotes the expression of TF t_k in the experiment a_j .

Motifs. Finally, consider a set of motifs m_ℓ , $\ell \in \{1, 2, \dots, n_M\}$ and let \mathbf{M} be the *motif matrix*: the value $M_{i\ell}$ is 1, if gene g_i has motif m_ℓ present in its promoter, and 0 otherwise.

The matrices \mathbf{G} , \mathbf{M} and \mathbf{T} make up the data to be analyzed. Figure 3.1 shows a convenient way to visualize these matrices.

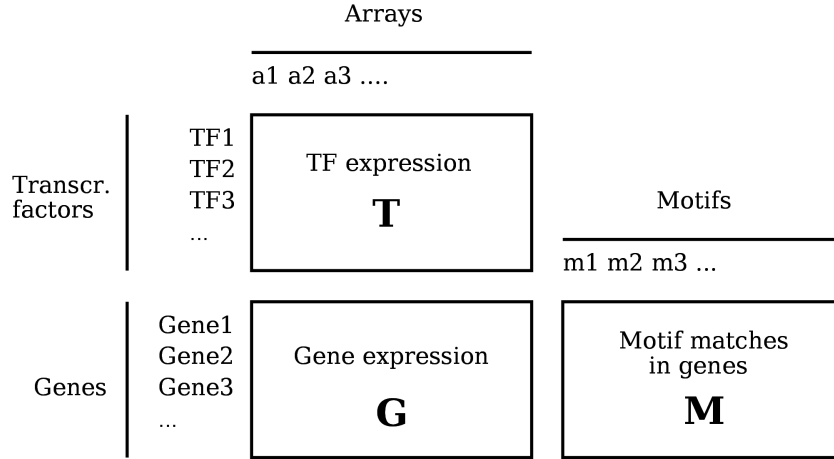


Figure 3.1: The matrices \mathbf{T} (top), \mathbf{G} (bottom left) and \mathbf{M} (bottom right). Each row of \mathbf{G} corresponds to a certain gene, as does each row of \mathbf{M} . Each column of \mathbf{G} corresponds to a certain array, as does each column of \mathbf{T} . The rows of \mathbf{M} can be regarded as descriptive attributes for the rows of \mathbf{G} , and the columns of \mathbf{T} – as the attributes for the columns of \mathbf{G} .

Now we can formulate our task more precisely. We present a model that predicts the expression G_{ij} of gene g_i on the array a_j , given the description of the gene as a vector of its motif counts $\mathbf{M}^{(i*)} = (M_{i1}, M_{i2}, \dots, M_{in_M})$, and the description of the array as a vector of TF expression levels: $\mathbf{T}^{(*j)} = (T_{1j}, T_{2j}, \dots, T_{n_Tj})^T$. We estimate model parameters using the available microarray and sequence data and interpret the obtained parameter values to gain useful insights into the regulatory processes underlying gene expression.

3.2 Model Specification

Here we present the central contribution of this thesis, the G=MAT model. We start with the definition and later on give the biological rationale for our choices.

Definition 3.1 (The G=MAT model) *Let \hat{G}_{ij} denote the prediction for the gene expression value. The G=MAT model defines this value as follows:*

$$\hat{G}_{ij} = \sum_{\ell=1}^{n_M} \sum_{k=1}^{n_T} \alpha_{\ell k} M_{i\ell} T_{kj}, \quad (3.1)$$

where $\alpha_{\ell k} \in \mathbb{R}$ are model parameters.

Note that equation (3.1) is *linear* in terms of pairwise products $M_{i\ell} T_{kj}$ – this puts it into the well-studied realm of linear models. Also note that equation (3.1) can be easily recast in matrix form and besides providing a nice name for the model (the pun on “Gene MATrices” is intentional), this will simplify further computations a lot.

Lemma 3.1 (The G=MAT model, matrix form) *Let $\hat{\mathbf{G}}$ denote the $n_G \times n_A$ matrix with values \hat{G}_{ij} . Then equation (3.1) can be written in the following form:*

$$\hat{\mathbf{G}} = \mathbf{MAT}, \quad (3.2)$$

where \mathbf{A} is the $n_M \times n_T$ matrix with values $\alpha_{\ell k}$.

Proof. By definition (2.1) of matrix multiplication, for any two matrices \mathbf{A} and \mathbf{B} :

$$(\mathbf{AB})_{ij} = \sum_k (\mathbf{A})_{ik} (\mathbf{B})_{kj},$$

and thus for any three matrices \mathbf{A} , \mathbf{B} and \mathbf{C} :

$$(\mathbf{ABC})_{ij} = \sum_k (\mathbf{AB})_{ik} (\mathbf{C})_{kj} = \sum_{\ell} \sum_k (\mathbf{A})_{i\ell} (\mathbf{B})_{\ell k} (\mathbf{C})_{kj}.$$

Therefore,

$$(\mathbf{MAT})_{ij} = \sum_{\ell} \sum_k (\mathbf{M})_{i\ell} (\mathbf{A})_{\ell k} (\mathbf{T})_{kj} = \sum_{\ell=1}^{n_M} \sum_{k=1}^{n_T} \alpha_{\ell k} M_{i\ell} T_{kj},$$

from which the result follows immediately. ■

3.3 Rationale

The choice of equation (3.1) for modeling gene expression is based on several assumptions and simplifications concerning the processes that underlie microarray experiments.

Expression \approx protein abundance. The first major assumption that we make is that microarray expression measurements are a representative measure of protein abundance in the cell. It is well known, however, that microarray measurements only represent the *amount of mRNA* in the cell. This can be different from the actual protein abundance, because mRNA still needs to be translated into protein and this translation process can be under separate regulatory control. However, we can hope that on average the error is not too large. This assumption is rather common (and often implicit) in other similar methods and without it we would have to involve the issues of the translation process into the model. This could make the model too complex to be usable.

Gene expression is regulated by TFs only. Secondly, we assume that in a given microarray experiment, the expression G_{ij} of any gene g_i in any experiment a_j is *only* dependent on the expressions of the transcription factors $(T_{1,j}, T_{2,j}, \dots)$ in that experiment. In formal terms,

$$G_{i,j} = f_i(T_{1,j}, T_{2,j}, \dots, T_{n_T,j}),$$

for some function f_i , that is independent of the experimental setting. That means that if in two different experiments all the transcription factors happen to have the same expression values, all of the genes will have the same expression values too. This is certainly not the case in reality, because there are other factors influencing gene expression, such as microRNAs, methylation, environment and, probably most importantly, tissue-specific external signals. Nonetheless, it is still a reasonable assumption to make, because, on average, TFs are believed to have the most effect on gene expression.

TFs regulate genes by binding to motifs only. Thirdly, we assume that TFs perform their functions by binding to certain motifs on the promoters of the genes. That means, if two genes happen to have identical promoters, they should have identical expression in any experiment. Together with the previous assumptions, it follows that there should exist a *single function* f , common to all genes and experiments, such that:

$$G_{i,j} = f(T_{1,j}, T_{2,j}, \dots, T_{n_T,j}; M_{i,1}, M_{i,2}, \dots, M_{i,n_M}).$$

Moreover, we assume that presence or absence of motifs has a “switching” effect on the transcription factors, that bind to them. If the gene has no binding sites for a certain TF, the TF has no effect on that gene at all, and vice-versa: if the gene has a certain binding site, it will be influenced by all transcription factors that bind to it. We can represent this idea by modulating the transcription factor expression by the (binary) motif presence values $M_{i\ell}$:

$$G_{i,j} = f \begin{pmatrix} M_{i,1}T_{1,j}, & M_{i,2}T_{1,j}, & \dots & M_{i,n_M}T_{1,j} \\ M_{i,1}T_{2,j}, & M_{i,2}T_{2,j}, & \dots & M_{i,n_M}T_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ M_{i,1}T_{n_T,j}, & M_{i,2}T_{n_T,j}, & \dots & M_{i,n_M}T_{n_T,j} \end{pmatrix}.$$

Again, this might not necessarily be true. Processes like DNA methylation and protein phosphorylation can interfere with binding, also the strength of the binding site might be of importance. Nonetheless, according to current knowledge this assumption is a rather viable approximation.

The regulation rule f is *linear*. Finally, in order to be able to construct our model and obtain statistically reliable parameter fits on the data, we have to assume some reasonably simple, yet still informative form for the function f . Assuming that f is continuous, we could use its Taylor expansion to approximate it to arbitrary precision with a polynomial function:

$$\begin{aligned} f(M_{i,1}T_{1,j}, \dots, M_{i,\ell}T_{k,j}, \dots, M_{i,n_M}T_{n_T,j}) &\approx \\ &\approx a + \sum_{k,\ell} b_{k\ell} M_{i,\ell} T_{k,j} + \sum_{k,\ell,\kappa,\lambda} b_{k\ell\kappa\lambda} M_{i,\ell} T_{k,j} M_{i,\lambda} T_{\kappa,j} + \dots \end{aligned}$$

However, already the second order polynomial would have more than $n_M^2 n_T^2$ parameters, which is way more that can be reliably estimated using a dataset of conceivable size. Therefore, we choose to approximate f with a first-order model and obtain our final result:

$$G_{ij} = b + \sum_{\ell=1}^{n_M} \sum_{k=1}^{n_T} \alpha_{\ell k} M_{i\ell} T_{kj} + \varepsilon_{ij},$$

where ε_{ij} is the noise that cannot be explained by the approximation. We drop the offset term b from our model and consider an optional centering of the data instead. In fact, microarray data is always nearly centered.

Clearly, this linear approximation is a strong and most probably a wrong assumption in practice. However, as long as the factors responsible for the noise ε_{ij} are independent of (and thus orthogonal with) the input variables that we consider

in our model, we are still going to obtain meaningful and interpretable parameter values. Experiments on real data show that this is the case here.

3.4 Interpretation

The parameters of the model have a simple and clear interpretation: a large positive (or negative) value of $\alpha_{\ell k}$ shows that expression of TF t_k positively (or negatively) correlates with the expression of genes that have motif m_ℓ in their promoter. In other words, it means that:

- If you increase the expression of transcription factor t_k , leaving everything else intact, the average expression level of genes having motif m_ℓ would increase (or decrease) proportionally.
- If you add motif m_ℓ to the promoters of genes that didn't have it before, the expression level of these genes would increase (or decrease) proportionally with the expression level of TF t_k .

Therefore, a large absolute value of $\alpha_{\ell k}$ could indicate that either there is a direct binding of TF t_k to the motif m_ℓ , or the TF t_k must participate in regulatory process somehow involving motif m_ℓ .

Another way to see this is to consider disjunctive logical rules like the following:

$$\text{UP}(g) = \text{UP}(t_1) \mid \neg \text{UP}(t_2) ,$$

i.e., “gene g is upregulated iff TF t_1 is upregulated or TF t_2 is downregulated”.

Suppose TF t_1 regulates by binding to motif m_1 and TF t_2 regulates by binding to motif m_3 . In this case, the rule can be rewritten as

$$\begin{aligned} \text{UP}(g) = & \text{BINDS}(m_1, t_1) \ \& \ \text{HASMOTIF}(g, m_1) \ \& \ \text{UP}(t_1) \mid \\ & \neg(\text{BINDS}(m_3, t_2) \ \& \ \text{HASMOTIF}(g, m_3) \ \& \ \text{UP}(t_2)) . \end{aligned}$$

Now, if we denote $\text{UP}(t_k)$ by a binary variable T_{k*} , $\text{HASMOTIF}(g, m_\ell)$ by a binary variable $M_{*\ell}$ and $\text{BINDS}(m_\ell, t_k)$ by $\alpha_{\ell k}$, we can rewrite any such rule in a form of a familiar (but this time thresholded) linear function:

$$G = \text{sign}(\alpha_{1,1}M_{*,1}T_{1,*} - \alpha_{3,2}M_{*,3}T_{2,*}) .$$

3.5 Related Work

Finally, we note that our approach fits well with the mindset of many other methods for the analysis of microarray or sequence data.

Coexpression analysis. Perhaps the most common use of microarray data is searching for groups of genes with similar expression profiles. There is a reason to believe that such *coexpressed* genes are also *coregulated*, and thus might contain the same motifs in their promoters. This is, indeed, often the case, and therefore such approaches have been extensively used for *de-novo motif discovery* from genomic sequences [VBJ⁺00]. Our method can be regarded, in some sense, as a search for genes that have some common motif and are coexpressed with some transcription factor at the same time.

Predictive modeling. Various kinds of predictive models have been conceived for microarray data before: predicting gene expression from TF expression [Soi03, SKB03], predicting gene expression from motif presence using a linear model [BLS01] and also predicting gene expression from both kinds of data using combinatorial analysis of motif correlation [PSC01], two-dimensional regression trees [RZ06] and ADT-trees [MKW⁺04]. Our method continues this trend and uses a linear predictive model.

3.6 Example

To better illustrate our ideas, let us design a small example of a regulatory network that works in accordance with the proposed model.

Suppose that there are 5 transcription factors t_k , 5 motifs m_ℓ , and that the following relations hold among them:

- TF t_1 is a transcriptional activator that binds to motif m_1 ,
- TF t_2 is a strong suppressor that binds to motif m_2 ,
- TF t_3 is an activator that also binds to m_2 ,
- TF t_4 can bind to m_3 and m_4 and acts as an activator when bound to m_3 and as a suppressor when bound to m_4 .
- TF t_5 is an activator that can bind to both m_4 and m_5 , but its activity is stronger when it is bound to m_5 .

These relations are depicted in Figure 3.2.

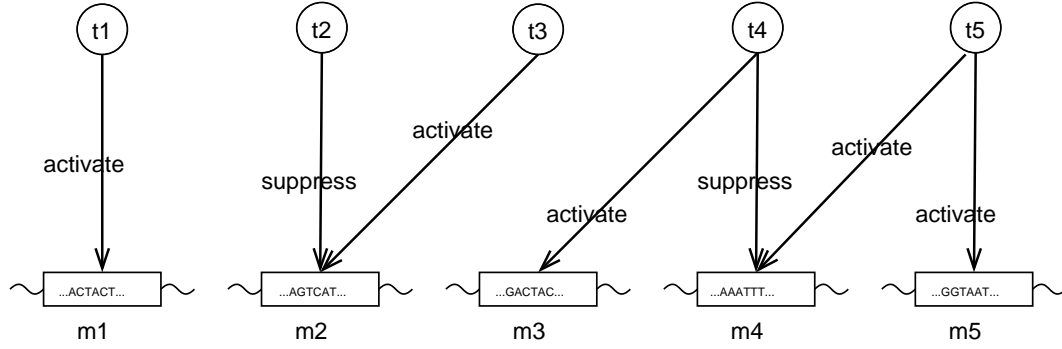


Figure 3.2: The regulatory rules of the example. Arrows indicate which TF-s bind to which motifs and what effect they have on transcription regulation.

The relations are described by the following parameter matrix:

$$\mathbf{A} = \begin{array}{c|ccccc} & t_1 & t_2 & t_3 & t_4 & t_5 \\ \hline m_1 & 1 & 0 & 0 & 0 & 0 \\ m_2 & 0 & -2 & 1 & 0 & 0 \\ m_3 & 0 & 0 & 0 & 1 & 0 \\ m_4 & 0 & 0 & 0 & -1 & 1 \\ m_5 & 0 & 0 & 0 & 0 & 2 \end{array} . \quad (3.3)$$

A positive entry $\alpha_{\ell k}$ indicates that TF t_k induces transcription when bound to motif m_ℓ . A negative entry denotes the suppressive effect of the corresponding TF. Larger values indicate stronger effect.

Suppose we have a set of 5 genes, described by the following motif matrix:

$$\mathbf{M} = \begin{array}{c|ccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ \hline g_1 & 1 & 1 & 0 & 0 & 0 \\ g_2 & 0 & 0 & 1 & 1 & 0 \\ g_3 & 0 & 1 & 0 & 1 & 0 \\ g_4 & 0 & 0 & 1 & 0 & 0 \\ g_5 & 0 & 0 & 1 & 0 & 1 \end{array} . \quad (3.4)$$

That is, the promoter region of gene g_1 contains motifs m_1 and m_2 , the promoter region of g_2 contains m_3 and m_4 , etc.

Imagine that we performed 4 microarray experiments a_1, a_2, a_3, a_4 , and the expression levels of the TF-s in these experiments are given by the following TF

expression matrix:

$$\mathbf{T} = \begin{array}{c|cccc} & a_1 & a_2 & a_3 & a_4 \\ \hline t_1 & 1 & 0 & 1 & 0 \\ t_2 & 0 & 1 & 0 & 1 \\ t_3 & 1 & 1 & 0 & 1 \\ t_4 & 0 & 0 & 1 & 0 \\ t_5 & 0 & 1 & 0 & 0 \end{array} . \quad (3.5)$$

It means that TF t_1 was upregulated in the experiments a_1 and a_3 , TF t_2 was upregulated in the experiments a_2 and a_4 , etc.

At last, suppose that genetic regulation indeed follows the proposed model. Then we have all the information to determine the expression of each target gene g_i in each experiment a_j . For example, the gene g_1 contains motifs m_1 and m_2 and is therefore regulated by the TF-s t_1 , t_2 and t_3 . In the experiment a_1 the TF-s t_1 and t_2 are upregulated. Each of them has an effect of 1 on the gene. Therefore the total expression level of g_1 in a_1 is:

$$\begin{aligned} G_{1,1} &= \alpha_{1,1}M_{1,1}T_{1,1} + \alpha_{2,2}M_{1,2}T_{2,1} + \alpha_{2,3}M_{1,2}T_{3,1} \\ &= 1 \cdot 1 \cdot 1 - 2 \cdot 1 \cdot 0 + 1 \cdot 1 \cdot 1 = 2. \end{aligned}$$

As another example, consider the expression of g_1 in a_2 . Here the TF-s t_2 , t_3 and t_5 are upregulated, therefore:

$$G_{1,2} = 1 \cdot 1 \cdot 0 - 2 \cdot 1 \cdot 1 + 1 \cdot 1 \cdot 1 = -1.$$

Continuing in the same manner we can calculate the full expression matrix:

$$\mathbf{G} = \begin{array}{c|cccc} & a_1 & a_2 & a_3 & a_4 \\ \hline g_1 & 2 & -1 & 1 & -1 \\ g_2 & 0 & 1 & 0 & 0 \\ g_3 & 1 & 0 & -1 & -1 \\ g_4 & 0 & 0 & 1 & 0 \\ g_5 & 0 & 2 & 1 & 0 \end{array} .$$

In reality, we only get to observe the matrices \mathbf{G} , \mathbf{M} and \mathbf{T} . The task is to estimate the matrix \mathbf{A} and use it to make conclusions about the underlying regulatory rules. In the following chapter we describe a number of ways to solve this task and afterwards continue with the example.

Chapter 4

Parameter Estimation Methods

Data mining – torturing the data until it confesses.

Now that we have provided the definition for the G=MAT model, let us present a number of methods for parameter estimation from data.

Although numerous standard methods exist for the estimation of parameters of linear models, none of them can be immediately applied to a realistically-sized G=MAT dataset. Indeed, if we regard equation (3.1) as a linear model with input variables $M_\ell T_k$ and output variable G (and having one data point for each pair i, j), we shall end up with a data matrix of $n_M \times n_T$ variables and $n_G \times n_A$ data points. However, even for a rather small yeast dataset, the number of genes n_G is about 6000 and n_A , n_M and n_T can be of the order of 100. This results in a data matrix of size $600\,000 \times 10\,000$ – way too large for convenient processing. Another drawback of the straightforward linear model representation is that it can be misleading: due to the structure of the problem, the “600 000 data points” in the presented example actually correspond to a much lower input dimensionality and may very well be not enough for a reliable fit of the 10 000 parameters.

Therefore, we propose customized versions of the classical estimation techniques that exploit the structure of the problem.

4.1 Least Squares Regression

The most natural way of approaching the estimation problem is to search for parameter matrix \mathbf{A} , for which the mean squared error of model predictions is minimal. This corresponds to the classical *least squares fit*.

Definition 4.1 (Least squares estimate) Let \mathbf{G} , \mathbf{M} and \mathbf{T} be the proper $G=MAT$ matrices. Define the least squares fit for the parameter matrix $\hat{\mathbf{A}}_{\text{LS}}$ as follows:

$$\hat{\mathbf{A}}_{\text{LS}} = \underset{\mathbf{A}}{\operatorname{argmin}} \|\mathbf{G} - \mathbf{MAT}\|^2, \quad (4.1)$$

where $\|\cdot\|^2$ here denotes the sum of squares of the elements of a given matrix.

The problem (4.1) always has a solution, although sometimes the solution is not unique.

Theorem 4.1 Suppose the columns of \mathbf{M} and the rows of \mathbf{T} are linearly independent, that is,

$$\operatorname{rank}(\mathbf{M}) = n_{\mathbf{M}} \quad \operatorname{rank}(\mathbf{T}) = n_{\mathbf{T}}.$$

Then the problem (4.1) has a unique solution:

$$\hat{\mathbf{A}}_{\text{LS}} = (\mathbf{M}^T \mathbf{M})^{-1} \mathbf{M}^T \mathbf{G} \mathbf{T}^T (\mathbf{T} \mathbf{T}^T)^{-1}, \quad (4.2)$$

where $(\cdot)^T$ denotes matrix transposition.

Proof. First, note that the objective function in the problem (4.1) is a quadratic convex function simply because the problem is equivalent to the least squares problem of a “traditional” linear model (see Section 2.4.1). Therefore, there must exist a minimum, not necessarily unique, which is achieved precisely in the point(s) where the gradient becomes $\mathbf{0}$. Hence, to find the minimum, we solve as follows:

$$\begin{aligned} \frac{\partial \|\mathbf{G} - \mathbf{MAT}\|^2}{\partial \mathbf{A}} &= \mathbf{0}, \\ \frac{\partial}{\partial \mathbf{A}} \sum_{i,j} (\mathbf{G}_{ij} - (\mathbf{MAT})_{ij})^2 &= \mathbf{0}. \end{aligned}$$

We take the derivative of the sum of squares in the same way as it was done in Section 2.4.1:

$$\begin{aligned} \sum_{i,j} 2(\mathbf{G}_{ij} - (\mathbf{MAT})_{ij}) \frac{\partial (-(\mathbf{MAT})_{ij})}{\partial \mathbf{A}} &= \mathbf{0}, \\ \sum_{i,j} 2(\mathbf{G}_{ij} - (\mathbf{MAT})_{ij}) \frac{\partial \left(-\sum_{\ell,k} \alpha_{\ell k} M_{i\ell} T_{kj} \right)}{\partial \mathbf{A}} &= \mathbf{0}, \\ \sum_{i,j} -2(\mathbf{G}_{ij} - (\mathbf{MAT})_{ij}) M_{i\ell} T_{kj} &= 0, \text{ for all } \ell, k. \end{aligned}$$

We divide the last equation by -2 and rewrite it conveniently in matrix form:

$$\begin{aligned} \sum_{i,j} (\mathbf{M}^T)_{\ell i} (\mathbf{G} - \mathbf{M}\mathbf{A}\mathbf{T})_{ij} (\mathbf{T}^T)_{jk} &= 0, \text{ for all } \ell, k, \\ \mathbf{M}^T (\mathbf{G} - \mathbf{M}\mathbf{A}\mathbf{T}) \mathbf{T}^T &= \mathbf{0}, \\ \mathbf{M}^T \mathbf{G} \mathbf{T}^T &= \mathbf{M}^T \mathbf{M} \mathbf{A} \mathbf{T} \mathbf{T}^T. \end{aligned} \quad (4.3)$$

If $\text{rank}(\mathbf{T}) = n_T$ and $\text{rank}(\mathbf{M}) = n_M$, the matrices $\mathbf{T}\mathbf{T}^T$ and $\mathbf{M}^T\mathbf{M}$ are necessarily invertible (refer to equations (2.2) and (2.4) in Section 2.1). Consequently, we can multiply equation (4.3) by $(\mathbf{M}^T\mathbf{M})^{-1}$ on the left and by $(\mathbf{T}\mathbf{T}^T)^{-1}$ on the right, obtaining precisely the solution (4.2). \blacksquare

If the matrices \mathbf{M} and \mathbf{T} do not have full rank, the solution is not unique.

Theorem 4.2 *All solutions to the problem (4.1) can be computed as:*

$$\hat{\mathbf{A}}_{\text{LS}} = \mathbf{M}^+ \mathbf{G} \mathbf{T}^+ + (\mathbf{M}^+ \mathbf{M} - \mathbf{I}) \mathbf{K} + \mathbf{L} (\mathbf{T} \mathbf{T}^+ - \mathbf{I}), \quad (4.4)$$

where $(\cdot)^+$ denotes the Moore-Penrose pseudoinverse of a matrix (see Section 2.1), \mathbf{I} denotes a properly-sized identity matrix and \mathbf{K} and \mathbf{L} are any two $n_M \times n_T$ matrices.

Proof. As we know from the proof of Theorem 4.1, the solutions to the problem (4.1) are precisely the solutions of equation (4.3). It remains to show that these are precisely all matrices of the form (4.4).

SUFFICIENCY: Let $\hat{\mathbf{A}}_{\text{LS}}$ be given by equation (4.4). Then it is a solution of (4.3). To show it, we first note that

$$\mathbf{M}^T \mathbf{M} (\mathbf{M}^+ \mathbf{M} - \mathbf{I}) = \mathbf{M}^T (\mathbf{M} \mathbf{M}^+ \mathbf{M}) - \mathbf{M}^T \mathbf{M} = \mathbf{M}^T \mathbf{M} - \mathbf{M}^T \mathbf{M} = \mathbf{0}, \quad (4.5)$$

$$(\mathbf{T} \mathbf{T}^+ - \mathbf{I}) \mathbf{T} \mathbf{T}^T = (\mathbf{T} \mathbf{T}^+ \mathbf{T}) \mathbf{T}^T - \mathbf{T} \mathbf{T}^T = \mathbf{T} \mathbf{T}^T - \mathbf{T} \mathbf{T}^T = \mathbf{0}, \quad (4.6)$$

due to the property (2.9) of a pseudoinverse. Thus,

$$\begin{aligned} \mathbf{M}^T \mathbf{M} \hat{\mathbf{A}}_{\text{LS}} \mathbf{T} \mathbf{T}^T &= \mathbf{M}^T \mathbf{M} (\mathbf{M}^+ \mathbf{G} \mathbf{T}^+ + (\mathbf{M}^+ \mathbf{M} - \mathbf{I}) \mathbf{K} + \mathbf{L} (\mathbf{T} \mathbf{T}^+ - \mathbf{I})) \mathbf{T} \mathbf{T}^T \\ &= \mathbf{M}^T \mathbf{M} \mathbf{M}^+ \mathbf{G} \mathbf{T}^+ \mathbf{T} \mathbf{T}^T, \end{aligned} \quad (4.7)$$

because after opening the brackets in (4.7) the second and the third term in the sum disappear due to (4.5) and (4.6). Using the pseudoinverse properties (2.13) and (2.14) we can now obtain:

$$\mathbf{M}^T \mathbf{M} \hat{\mathbf{A}}_{\text{LS}} \mathbf{T} \mathbf{T}^T = (\mathbf{M}^T \mathbf{M} \mathbf{M}^+) \mathbf{G} (\mathbf{T}^+ \mathbf{T} \mathbf{T}^T) = \mathbf{M}^T \mathbf{G} \mathbf{T}^T,$$

and therefore $\hat{\mathbf{A}}_{\text{LS}}$ is indeed a solution to (4.3).

NECESSITY: Let \mathbf{A}' be some solution of (4.3). Choose \mathbf{K}' and \mathbf{L}' as follows:

$$\begin{aligned}\mathbf{K}' &= (\mathbf{M}^+\mathbf{M} - \mathbf{I})(\mathbf{A}'\mathbf{T}\mathbf{T}^+ - \mathbf{M}^+\mathbf{G}\mathbf{T}^+) \\ \mathbf{L}' &= (\mathbf{A}' - \mathbf{M}^+\mathbf{G}\mathbf{T}^+)(\mathbf{T}\mathbf{T}^+ - \mathbf{I})\end{aligned}$$

And let

$$\hat{\mathbf{A}}_{\text{LS}} = \mathbf{M}^+\mathbf{G}\mathbf{T}^+ + (\mathbf{M}^+\mathbf{M} - \mathbf{I})\mathbf{K}' + \mathbf{L}'(\mathbf{T}\mathbf{T}^+ - \mathbf{I}). \quad (4.8)$$

Then

$$\hat{\mathbf{A}}_{\text{LS}} = \mathbf{A}'.$$

To see that, simply substitute the expressions for \mathbf{K}' and \mathbf{L}' . We do it in several steps. First, note that for any \mathbf{X} :

$$\begin{aligned}(\mathbf{X}^+\mathbf{X} - \mathbf{I})(\mathbf{X}^+\mathbf{X} - \mathbf{I}) &= \mathbf{X}^+\mathbf{X}\mathbf{X}^+\mathbf{X} - 2\mathbf{X}^+\mathbf{X} + \mathbf{I} \\ &= \mathbf{X}^+\mathbf{X} - 2\mathbf{X}^+\mathbf{X} + \mathbf{I} = \mathbf{I} - \mathbf{X}^+\mathbf{X} \\ (\mathbf{X}\mathbf{X}^+ - \mathbf{I})(\mathbf{X}\mathbf{X}^+ - \mathbf{I}) &= \mathbf{X}\mathbf{X}^+\mathbf{X}\mathbf{X}^+ - 2\mathbf{X}\mathbf{X}^+ + \mathbf{I} \\ &= \mathbf{X}\mathbf{X}^+ - 2\mathbf{X}\mathbf{X}^+ + \mathbf{I} = \mathbf{I} - \mathbf{X}\mathbf{X}^+\end{aligned}$$

Next, consider the term $(\mathbf{M}^+\mathbf{M} - \mathbf{I})\mathbf{K}'$:

$$\begin{aligned}(\mathbf{M}^+\mathbf{M} - \mathbf{I})\mathbf{K}' &= (\mathbf{M}^+\mathbf{M} - \mathbf{I})(\mathbf{M}^+\mathbf{M} - \mathbf{I})(\mathbf{A}'\mathbf{T}\mathbf{T}^+ - \mathbf{M}^+\mathbf{G}\mathbf{T}^+) \\ &= (\mathbf{I} - \mathbf{M}^+\mathbf{M})(\mathbf{A}'\mathbf{T}\mathbf{T}^+ - \mathbf{M}^+\mathbf{G}\mathbf{T}^+) \\ &= \mathbf{A}'\mathbf{T}\mathbf{T}^+ - \mathbf{M}^+\mathbf{G}\mathbf{T}^+ - \mathbf{M}^+\mathbf{M}\mathbf{A}'\mathbf{T}\mathbf{T}^+ + \mathbf{M}^+\mathbf{M}\mathbf{M}^+\mathbf{G}\mathbf{T}^+ \\ &= \mathbf{A}'\mathbf{T}\mathbf{T}^+ - \mathbf{M}^+\mathbf{G}\mathbf{T}^+ - \mathbf{M}^+\mathbf{M}\mathbf{A}'\mathbf{T}\mathbf{T}^+ + \mathbf{M}^+\mathbf{G}\mathbf{T}^+ \\ &= \mathbf{A}'\mathbf{T}\mathbf{T}^+ - \mathbf{M}^+\mathbf{M}\mathbf{A}'\mathbf{T}\mathbf{T}^+\end{aligned} \quad (4.9)$$

Similarly for $\mathbf{L}'(\mathbf{T}\mathbf{T}^+ - \mathbf{I})$:

$$\begin{aligned}\mathbf{L}'(\mathbf{T}\mathbf{T}^+ - \mathbf{I}) &= (\mathbf{A}' - \mathbf{M}^+\mathbf{G}\mathbf{T}^+)(\mathbf{T}\mathbf{T}^+ - \mathbf{I})(\mathbf{T}\mathbf{T}^+ - \mathbf{I}) \\ &= (\mathbf{A}' - \mathbf{M}^+\mathbf{G}\mathbf{T}^+)(\mathbf{I} - \mathbf{T}\mathbf{T}^+) \\ &= \mathbf{A}' - \mathbf{M}^+\mathbf{G}\mathbf{T}^+ - \mathbf{A}'\mathbf{T}\mathbf{T}^+ + \mathbf{M}^+\mathbf{G}\mathbf{T}^+\mathbf{T}\mathbf{T}^+ \\ &= \mathbf{A}' - \mathbf{M}^+\mathbf{G}\mathbf{T}^+ - \mathbf{A}'\mathbf{T}\mathbf{T}^+ + \mathbf{M}^+\mathbf{G}\mathbf{T}^+ \\ &= \mathbf{A}' - \mathbf{A}'\mathbf{T}\mathbf{T}^+\end{aligned} \quad (4.10)$$

Finally, substitute (4.9) and (4.10) into (4.8):

$$\begin{aligned}\hat{\mathbf{A}}_{\text{LS}} &= \mathbf{M}^+\mathbf{G}\mathbf{T}^+ + \mathbf{A}'\mathbf{T}\mathbf{T}^+ - \mathbf{M}^+\mathbf{M}\mathbf{A}'\mathbf{T}\mathbf{T}^+ + \mathbf{A}' - \mathbf{A}'\mathbf{T}\mathbf{T}^+ \\ &= \mathbf{A}' + \mathbf{M}^+\mathbf{G}\mathbf{T}^+ - \mathbf{M}^+\mathbf{M}\mathbf{A}'\mathbf{T}\mathbf{T}^+.\end{aligned} \quad (4.11)$$

As we know, \mathbf{A}' satisfies (4.3). Therefore, it follows that

$$\begin{aligned}\mathbf{M}^T \mathbf{G} \mathbf{T}^T &= \mathbf{M}^T \mathbf{M} \mathbf{A}' \mathbf{T} \mathbf{T}^T, \\ \mathbf{M}^+ (\mathbf{M}^+)^T \mathbf{M}^T \mathbf{G} \mathbf{T}^T (\mathbf{T}^+)^T \mathbf{T}^+ &= \mathbf{M}^+ (\mathbf{M}^+)^T \mathbf{M}^T \mathbf{M} \mathbf{A}' \mathbf{T} \mathbf{T}^T (\mathbf{T}^+)^T \mathbf{T}^+.\end{aligned}$$

Applying pseudoinverse properties (2.15) and (2.16) we obtain

$$\begin{aligned}\mathbf{M}^+ \mathbf{G} \mathbf{T}^+ &= \mathbf{M}^+ \mathbf{M} \mathbf{A}' \mathbf{T} \mathbf{T}^+, \\ \mathbf{M}^+ \mathbf{G} \mathbf{T}^+ - \mathbf{M}^+ \mathbf{M} \mathbf{A}' \mathbf{T} \mathbf{T}^+ &= \mathbf{0}.\end{aligned}\tag{4.12}$$

Substituting (4.12) into (4.11) produces the desired result and thus completes the proof. \blacksquare

In case the solution $\hat{\mathbf{A}}_{\text{LS}}$ is not unique, we shall only be interested in the *minimum-norm* fit: the solution $\hat{\mathbf{A}}_{\text{LS}*}$, having the least possible sum-of-squares $\|\hat{\mathbf{A}}_{\text{LS}*}\|^2$:

Theorem 4.3 *Let $\hat{\mathbf{A}}_{\text{LS}*}$ be the solution to (4.1) having the least possible sum of squares $\|\hat{\mathbf{A}}_{\text{LS}*}\|^2$. Then:*

$$\hat{\mathbf{A}}_{\text{LS}*} = \mathbf{M}^+ \mathbf{G} \mathbf{T}^+.\tag{4.13}$$

Proof. Let $\hat{\mathbf{A}}_{\text{LS}}$ be any solution of (4.1). We are going to show that

$$\|\hat{\mathbf{A}}_{\text{LS}}\|^2 = \|\mathbf{M}^+ \mathbf{G} \mathbf{T}^+\|^2 + \|(\mathbf{M}^+ \mathbf{M} - \mathbf{I})\mathbf{K} + \mathbf{L}(\mathbf{T} \mathbf{T}^+ - \mathbf{I})\|^2\tag{4.14}$$

and therefore

$$\|\hat{\mathbf{A}}_{\text{LS}}\| \geq \|\mathbf{M}^+ \mathbf{G} \mathbf{T}^+\| = \|\hat{\mathbf{A}}_{\text{LS}*}\|,$$

from which the desired result follows.

Let

$$\mathbf{M} = \mathbf{U}_M \mathbf{D}_M \mathbf{V}_M^T, \quad \mathbf{T} = \mathbf{U}_T \mathbf{D}_T \mathbf{V}_T^T\tag{4.15}$$

be the singular value decompositions of \mathbf{M} and \mathbf{T} respectively. Then, by definition given in (2.7):

$$\mathbf{M}^+ = \mathbf{V}_M \mathbf{D}_M^+ \mathbf{U}_M^T, \quad \mathbf{T}^+ = \mathbf{V}_T \mathbf{D}_T^+ \mathbf{U}_T^T\tag{4.16}$$

Substitute the decompositions into equation (4.4):

$$\begin{aligned}\hat{\mathbf{A}}_{\text{LS}} &= (\mathbf{V}_M \mathbf{D}_M^+ \mathbf{U}_M^T) \mathbf{G} (\mathbf{V}_T \mathbf{D}_T^+ \mathbf{U}_T^T) \\ &\quad + (\mathbf{V}_M \mathbf{D}_M^+ \mathbf{D}_M \mathbf{V}_M^T - \mathbf{I}) \mathbf{K} + \mathbf{L} (\mathbf{U}_T \mathbf{D}_T \mathbf{D}_T^+ \mathbf{U}_T^T - \mathbf{I}) \\ &= \mathbf{V}_M (\mathbf{D}_M^+ \mathbf{U}_M^T \mathbf{G} \mathbf{V}_T \mathbf{D}_T^+) \mathbf{U}_T^T \\ &\quad + \mathbf{V}_M (\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I}) \mathbf{V}_M^T \mathbf{K} + \mathbf{L} \mathbf{U}_T (\mathbf{D}_T \mathbf{D}_T^+ - \mathbf{I}) \mathbf{U}_T^T\end{aligned}$$

Due to orthogonality of \mathbf{V}_M and \mathbf{U}_T :

$$\|\hat{\mathbf{A}}_{LS}\|^2 = \|\mathbf{V}_M^T \hat{\mathbf{A}}_{LS} \mathbf{U}_T\|^2, \quad (4.17)$$

where

$$\begin{aligned} \mathbf{V}_M^T \hat{\mathbf{A}}_{LS} \mathbf{U}_T &= \mathbf{D}_M^+ \mathbf{U}_M^T \mathbf{G} \mathbf{V}_T \mathbf{D}_T^+ \\ &\quad + (\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I}) \mathbf{V}_M^T \mathbf{K} \mathbf{U}_T + \mathbf{V}_M^T \mathbf{L} \mathbf{U}_T (\mathbf{D}_T \mathbf{D}_T^+ - \mathbf{I}). \end{aligned}$$

Now note, that because \mathbf{D}_M^+ and \mathbf{D}_T^+ are diagonal matrices

$$(\mathbf{D}_M^+ \mathbf{U}_M^T \mathbf{G} \mathbf{V}_T \mathbf{D}_T^+)_{\ell k} = (\mathbf{D}_M^+)_{\ell \ell} (\mathbf{U}_M^T \mathbf{G} \mathbf{V}_T)_{\ell k} (\mathbf{D}_T^+)_{kk}. \quad (4.18)$$

This value is zero when $(\mathbf{D}_M^+)_{\ell \ell} = 0$ or $(\mathbf{D}_T^+)_{kk} = 0$.

Recall that by definition given in equation (2.8), \mathbf{D}_M^+ is a diagonal matrix with

$$(\mathbf{D}_M^+)_{\ell \ell} = \begin{cases} \frac{1}{(\mathbf{D}_M)_{\ell \ell}}, & \text{if } (\mathbf{D}_M)_{\ell \ell} \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore, $\mathbf{D}_M^+ \mathbf{D}_M$ is a diagonal matrix such that

$$(\mathbf{D}_M^+ \mathbf{D}_M)_{\ell \ell} = \begin{cases} 1, & \text{if } (\mathbf{D}_M)_{\ell \ell} \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, $(\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I})$ is a diagonal matrix, such that:

$$(\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I})_{\ell \ell} = 0 \quad \Leftrightarrow \quad (\mathbf{D}_M)_{\ell \ell} \neq 0.$$

It follows that for each ℓ such that $(\mathbf{D}_M^+)_{\ell \ell} \neq 0$ the corresponding row of $(\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I}) \mathbf{V}_M^T \mathbf{K} \mathbf{U}_T$ is zero. By similar logic, for each k such that $(\mathbf{D}_T^+)_{kk} \neq 0$ the corresponding column of $\mathbf{V}_M^T \mathbf{L} \mathbf{U}_T (\mathbf{D}_T \mathbf{D}_T^+ - \mathbf{I})$ is zero. Therefore, for each (ℓ, k) , such that $(\mathbf{D}_M^+)_{\ell \ell} \neq 0$ and $(\mathbf{D}_T^+)_{kk} \neq 0$

$$((\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I}) \mathbf{V}_M^T \mathbf{K} \mathbf{U}_T + \mathbf{V}_M^T \mathbf{L} \mathbf{U}_T (\mathbf{D}_T \mathbf{D}_T^+ - \mathbf{I}))_{\ell k} = 0.$$

But due to (4.18), these are exactly those (ℓ, k) for which $(\mathbf{D}_M^+ \mathbf{U}_M^T \mathbf{G} \mathbf{V}_T \mathbf{D}_T^+)_{\ell k}$ can be nonzero. Therefore,

$$\begin{aligned} \|\mathbf{V}_M^T \hat{\mathbf{A}}_{LS} \mathbf{U}_T\|^2 &= \|\mathbf{D}_M^+ \mathbf{U}_M^T \mathbf{G} \mathbf{V}_T \mathbf{D}_T^+\|^2 \\ &\quad + \|(\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I}) \mathbf{V}_M^T \mathbf{K} \mathbf{U}_T + \mathbf{V}_M^T \mathbf{L} \mathbf{U}_T (\mathbf{D}_T \mathbf{D}_T^+ - \mathbf{I})\|^2. \end{aligned} \quad (4.19)$$

Using orthogonality of \mathbf{V}_M and \mathbf{U}_T again:

$$\|\mathbf{D}_M^+ \mathbf{U}_M^T \mathbf{G} \mathbf{V}_T \mathbf{D}_T^+\|^2 = \|\mathbf{V}_M \mathbf{D}_M^+ \mathbf{U}_M^T \mathbf{G} \mathbf{V}_T \mathbf{D}_T^+ \mathbf{U}_T^T\|^2 = \|\mathbf{M}^+ \mathbf{G} \mathbf{T}^+\|^2, \quad (4.20)$$

and similarly:

$$\begin{aligned} & \|(\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I}) \mathbf{V}_M^T \mathbf{K} \mathbf{U}_T + \mathbf{V}_M^T \mathbf{L} \mathbf{U}_T (\mathbf{D}_T \mathbf{D}_T^+ - \mathbf{I})\|^2 \\ &= \|\mathbf{V}_M ((\mathbf{D}_M^+ \mathbf{D}_M - \mathbf{I}) \mathbf{V}_M^T \mathbf{K} \mathbf{U}_T + \mathbf{V}_M^T \mathbf{L} \mathbf{U}_T (\mathbf{D}_T \mathbf{D}_T^+ - \mathbf{I})) \mathbf{U}_T^T\|^2 \\ &= \|(\mathbf{M}^+ \mathbf{M} - \mathbf{I}) \mathbf{K} + \mathbf{L} (\mathbf{T} \mathbf{T}^+ - \mathbf{I})\|^2. \end{aligned} \quad (4.21)$$

Finally, substituting (4.17), (4.20) and (4.21) into (4.19), we obtain (4.14), which completes the proof. \blacksquare

The solution to the least squares regression problem can be computed with reasonable efficiency:

Theorem 4.4 *Let $n = \max(n_G, n_A, n_M, n_T)$. The solutions of equations (4.2) and (4.13) can be computed in time $O(n^3)$ and memory $O(n^2)$.*

Proof. Equation (4.2) contains only matrix multiplications and inversions. Each of these operations can be trivially implemented in time $O(n^3)$ and memory $O(n^2)$.

Equation (4.13) contains matrix multiplications and Moore-Penrose pseudoinverses. As the latter can be computed using singular value decomposition in time $O(n^3)$ [Cha82], the whole computation time is again $O(n^3)$. \blacksquare

More precisely, the time complexity of the computation depends linearly on the number of genes n_G and the number of microarrays n_A , and is cubic in the number of motifs and TFs n_M and n_T . Memory requirements are linear in n_G and n_A , and quadratic in n_M and n_T . This is important, as in many practical cases the number of genes n_G is significantly larger than n_A , n_M or n_T .

Unfortunately, there is one serious drawback of the least squares estimate: it can be rather unreliable for noisy data when the number of parameters is large or the input vectors are highly correlated.

In particular, as long as the number of parameters is close to the dimensionality of the input space (i.e., $n_M \approx n_A$ and $n_T \approx n_G$), the solution will be unstable and very sensitive to noise in the data, because in this case the model is “powerful enough” to fit the noise. For example, when \mathbf{T} and \mathbf{M} are square invertible matrices, the model error is always 0, for any dataset. A common way to deal with this problem is *regularization*.

4.2 Regularized Least Squares Regression

The idea of regularization is to “enforce” the solution with the smallest possible parameter values by penalizing the norm of the parameter matrix \mathbf{A} .

Definition 4.2 (ℓ_2 -norm regularized least squares estimate) *Define the regularized least squares fit for the parameter matrix $\hat{\mathbf{A}}_{\text{RLS}}$ as follows:*

$$\hat{\mathbf{A}}_{\text{RLS}} = \underset{\mathbf{A}}{\operatorname{argmin}} \left(\|\mathbf{G} - \mathbf{M}\mathbf{A}\mathbf{T}\|^2 + \lambda \|\mathbf{A}\|^2 \right), \quad (4.22)$$

where $\lambda \geq 0$ is the regularization parameter.

Regularization effectively makes the problem well-posed (the solution is always unique for $\lambda > 0$) and allows to trade off stability versus prediction accuracy. Setting $\lambda = 0$ will give us the best possible prediction, but low stability for noisy data – it is just the usual least squares solution. Setting $\lambda \rightarrow \infty$ will result in a constant solution $\hat{\mathbf{A}}_{\text{RLS}} = \mathbf{0}$, which is very stable, but useless for predicting. By choosing λ somewhere in between, we can obtain both satisfactory stability and prediction quality.

Unfortunately, we could not derive a closed analytical solution for the problem (4.22). We therefore propose to solve it using iterative methods, such as the straightforward gradient descent-based approach. Due to strict convexity of the objective function for $\lambda > 0$, it is guaranteed to converge to the optimal solution, and it can be implemented with marginal, yet still useful efficiency.

Theorem 4.5 *Let $n = \max(n_{\text{G}}, n_{\text{A}}, n_{\text{M}}, n_{\text{T}})$. The solution of equation (4.2) can be computed using gradient descent in time $O(kn^3)$ and memory $O(n^2)$, where k is the number of gradient descent iterations.*

Proof. Each step of the gradient descent requires a computation of the gradient of the objective function. This gradient is:

$$\begin{aligned} \frac{\partial \|\mathbf{G} - \mathbf{M}\mathbf{A}\mathbf{T}\|^2 + \lambda \|\mathbf{A}\|^2}{\partial \mathbf{A}} &= 2\mathbf{M}^T(\mathbf{G} - \mathbf{M}\mathbf{A}\mathbf{T})\mathbf{T}^T + 2\lambda\mathbf{A} = \\ &= 2(\mathbf{M}^T\mathbf{G}\mathbf{T}^T) - 2(\mathbf{M}^T\mathbf{M})\mathbf{A}(\mathbf{T}\mathbf{T}^T) + 2\lambda\mathbf{A}. \end{aligned}$$

Therefore, in each iteration of the algorithm we need to perform a number of matrix multiplications and additions, all of which can be done in time $O(n^3)$ and memory $O(n^2)$. ■

Again, it is worth noting, that the computation complexity is only linear in n_{G} .

To be able to compute a regularized estimate efficiently, we also propose a ridge regression-like estimate that combines the stability of a regularized solution and the efficiency of ordinary least squares.

Definition 4.3 (Ridge regression estimate) *Define the ridge regression fit for the parameter matrix $\hat{\mathbf{A}}_{\text{RR}}$ as follows:*

$$\hat{\mathbf{A}}_{\text{RR}} = (\mathbf{M}^T \mathbf{M} + \lambda_{\text{M}} \mathbf{I})^{-1} \mathbf{M}^T \mathbf{G} \mathbf{T}^T (\mathbf{T} \mathbf{T}^T + \lambda_{\text{T}} \mathbf{I})^{-1}, \quad (4.23)$$

where $\lambda_{\text{M}}, \lambda_{\text{T}} \geq 0$ are the regularization parameters and \mathbf{I} is the identity matrix.

Besides just mimicking the classical ridge regression syntactically, this method does correspond to it in spirit. To demonstrate that, let us first return to the classical least squares estimate and examine an interesting property of that solution.

Theorem 4.6 (2-step linear regression) *Let $\hat{\mathbf{A}}_{\text{LS}}$ be the minimum norm least squares fit (4.13) for a given dataset $(\mathbf{G}, \mathbf{M}, \mathbf{T})$. It is possible to compute $\hat{\mathbf{A}}_{\text{LS}}$ in two steps:*

- First compute $\hat{\mathbf{C}}_{\text{LS}}$ as a minimum norm least squares fit for the model $\mathbf{G} = \mathbf{MC}$.
- Then, compute $\hat{\mathbf{A}}_{\text{LS}}$ as a minimum norm least squares fit for the model $\hat{\mathbf{C}}_{\text{LS}} = \mathbf{AT}$.

Proof. The minimum norm least squares fit $\hat{\mathbf{C}}_{\text{LS}}$ for the model $\mathbf{G} = \mathbf{MC}$ can be computed as:

$$\hat{\mathbf{C}}_{\text{LS}} = \mathbf{M}^+ \mathbf{G}.$$

The minimum norm least squares fit $\tilde{\mathbf{A}}_{\text{LS}}$ for the model $\hat{\mathbf{C}}_{\text{LS}} = \mathbf{AT}$ can be computed as:

$$\tilde{\mathbf{A}}_{\text{LS}} = \hat{\mathbf{C}}_{\text{LS}} \mathbf{T}^+.$$

By combining the two equations above, we see that final $\tilde{\mathbf{A}}_{\text{LS}}$ is indeed equal to $\hat{\mathbf{A}}_{\text{LS}}$ from equation (4.4):

$$\tilde{\mathbf{A}}_{\text{LS}} = \hat{\mathbf{C}}_{\text{LS}} \mathbf{T}^+ = \mathbf{M}^+ \mathbf{G} \mathbf{T}^+ = \hat{\mathbf{A}}_{\text{LS}}.$$

■

The observation above shows, that the matrix \mathbf{A} in $\mathbf{G}=\mathbf{MAT}$ can actually be computed step by step:

- First express the gene expression values \mathbf{G} as linear combinations of motifs: $\mathbf{G} = \mathbf{MC}$. This results in a $n_{\text{M}} \times n_{\text{A}}$ matrix \mathbf{C} , that, in a sense, shows the “activity” of each motif in each experiment.

- Next, express this “motif activity” in each experiment as a linear combination of transcription factor activities: $\mathbf{C} = \mathbf{A}\mathbf{T}$. Now the resulting \mathbf{A} can be interpreted as showing the “transcription factor contribution to motif contribution to gene activity”, which is exactly how we interpret \mathbf{A} in $\mathbf{G} = \mathbf{M}\mathbf{A}\mathbf{T}$. Of course, the order of the steps can be reversed.

Now we can easily interpret the ridge-regression estimate as a two-step linear regression with ℓ_2 -regularization at each step (see Section 2.4.1).

4.3 Sparse Regression

Due to the specifics of the problem, we are only interested in a small number of most influential parameters of our model. Therefore, we would like to obtain a parameter fit with as many zero coefficients as possible. This can be achieved by penalizing every nonzero coefficient. This is the main idea of *sparse regression*.

Definition 4.4 (ℓ_0 -norm regularized estimate) Define the ℓ_0 -norm regularized fit for the parameter matrix $\hat{\mathbf{A}}_{\ell_0}$ as follows:

$$\hat{\mathbf{A}}_{\ell_0} = \underset{\mathbf{A}}{\operatorname{argmin}} \left(\|\mathbf{G} - \mathbf{M}\mathbf{A}\mathbf{T}\|^2 + \lambda \|\mathbf{A}\|_0 \right) , \quad (4.24)$$

where $\lambda \geq 0$ is the regularization parameter and $\|\cdot\|_0$ denotes the ℓ_0 -norm – the number of nonzero elements in a given matrix.

The computation of this ℓ_0 -norm regularized estimate, is a difficult (NP-complete) problem, therefore in practice we must use an approximation. The ℓ_1 -norm regularization is known to result in sparse estimates that are nearly as good as the ℓ_0 -ones.

Definition 4.5 (ℓ_1 -norm regularized estimate) Define the ℓ_1 -norm regularized fit for the parameter matrix $\hat{\mathbf{A}}_{\ell_1}$ as follows:

$$\hat{\mathbf{A}}_{\ell_1} = \underset{\mathbf{A}}{\operatorname{argmin}} \left(\|\mathbf{G} - \mathbf{M}\mathbf{A}\mathbf{T}\|^2 + \lambda \|\mathbf{A}\|_1 \right) , \quad (4.25)$$

where $\lambda \geq 0$ is the regularization parameter and $\|\cdot\|_1$ denotes the ℓ_1 -norm – the sum of absolute values of the elements in a given matrix.

Similarly to the ℓ_2 -norm regularization, no simple closed-form solution probably exists for sparse regression, but we can solve it with iterative methods such as iterative thresholding [DDDM04] in at least $O(kn^3)$ time. Another interesting option is to apply the *Least Angle Regression (LARS)* algorithm [EHJT04]. The LARS algorithm can reconstruct the whole set of solutions to equation (4.25) for all values of λ , because this set can be represented as a single piecewise-linear path

with $O(n_M \times n_T)$ nodes. In practice it means that we can request the algorithm to output N most important coefficients only: these correspond to the first N nodes of the path. The problem with LARS, however, is that although the algorithm does lend itself to some minor G=MAT-specific optimizations, we could not find a way to reduce its overall complexity. In each iteration of the algorithm, a certain *equiangular vector* needs to be computed, which requires an inversion of a $k \times k$ matrix, where k is the iteration number. As the complete run of the algorithm requires about $n_M \times n_T$ iterations, the overall complexity of a full run of LARS is dominated by the term $O(n_M^4 n_T^4)$. Nonetheless, the first iterations of the algorithm are quite fast.

4.4 Correlation-based Estimate

We can regard G=MAT as a generative probabilistic model. It turns out that if we impose the (not very unrealistic) assumption, that the input variables are uncorrelated, we can obtain a simple estimate for the parameters with some nice properties.

In the following, let T_k denote a random variable that represents the expression value of TF t_k in a randomly chosen microarray experiment. Namely, we regard the k -th row of matrix \mathbf{T} as containing the realizations of this random variable. By M_ℓ we denote the random variable that expresses the presence of motif m_ℓ in a randomly chosen gene. The realizations of this variable are given in the ℓ -th column of matrix \mathbf{M} . Finally, G denotes the random variable, representing the expression of a randomly chosen target gene in a randomly chosen experiment. Realizations of this variable are all the values in the matrix \mathbf{G} . In these terms, the G=MAT can be stated as a generative probabilistic model.

Definition 4.6 (Probabilistic G=MAT) *Let $(M_\ell)_{\ell \in \{1 \dots n_M\}}$, $(T_k)_{k \in \{1 \dots n_T\}}$ and G be random variables. We say that these variables are G=MAT-distributed with parameters $(\alpha_{\ell k})$, if they satisfy the equation:*

$$G = \sum_{\ell=1}^{n_M} \sum_{k=1}^{n_T} \alpha_{\ell k} M_\ell T_k + \varepsilon, \quad (4.26)$$

where ε is a random variable, independent of M_ℓ and T_k for all ℓ and k .

If we assume that the input variables M_ℓ and T_k are uncorrelated, we can express the parameters $\alpha_{\ell k}$ in a convenient way.

Theorem 4.7 (Correlation-based estimate) *Let (G, M_ℓ, T_k) be G=MAT-distributed random variables with parameters $(\alpha_{\ell k})$. Suppose that the variables M_ℓ*

and T_k are not constant:

$$D(M_\ell) \neq 0, \quad D(T_k) \neq 0 \text{ for any } \ell \in \{1 \dots n_M\}, k \in \{1 \dots n_T\},$$

and all pairwise uncorrelated:

$$\begin{aligned} \text{cov}(M_\ell, M_\lambda) &= 0, \text{ for any } \ell \neq \lambda, \\ \text{cov}(T_k, T_\kappa) &= 0, \text{ for any } k \neq \kappa, \\ \text{cov}(M_\ell, T_k) &= 0, \text{ for any } \ell, k. \end{aligned}$$

Then it holds that:

$$\alpha_{\ell k} = \frac{\text{cov}(G, (M_\ell - \bar{M}_\ell)(T_k - \bar{T}_k))}{D(M_\ell) D(T_k)}, \quad (4.27)$$

where $\text{cov}(X, Y) = E((X - \bar{X})(Y - \bar{Y}))$ denotes covariance, $D(X) = \text{cov}(X, X)$ denotes variance and $\bar{X} = E(X)$ denotes the mean of a random variable X .

To prove this theorem we shall use the following result.

Lemma 4.8 *Let M_ℓ, T_k be pairwise uncorrelated random variables. Then*

$$\text{cov}(M_\lambda T_\kappa, (M_\ell - \bar{M}_\ell)(T_k - \bar{T}_k)) = \begin{cases} D(M_\ell) D(T_k), & \text{if } \lambda = \ell, \kappa = k, \\ 0, & \text{otherwise.} \end{cases} \quad (4.28)$$

Proof. Using the equation $\text{cov}(X, Y) = E(XY) - E(X)E(Y)$ and remembering that M_ℓ and M_λ are not correlated with T_k and T_κ , we compute:

$$\begin{aligned} \text{cov}(M_\lambda T_\kappa, (M_\ell - \bar{M}_\ell)(T_k - \bar{T}_k)) &= \\ &= E(M_\lambda T_\kappa (M_\ell - \bar{M}_\ell)(T_k - \bar{T}_k)) - E(M_\lambda T_\kappa) E((M_\ell - \bar{M}_\ell)(T_k - \bar{T}_k)) \\ &= E(M_\lambda (M_\ell - \bar{M}_\ell)) E(T_\kappa (T_k - \bar{T}_k)) - E(M_\lambda T_\kappa) E(M_\ell - \bar{M}_\ell) E(T_k - \bar{T}_k) \\ &= E(M_\lambda (M_\ell - \bar{M}_\ell)) E(T_\kappa (T_k - \bar{T}_k)). \end{aligned} \quad (4.29)$$

For $\ell \neq \lambda$, this equation evaluates to:

$$E(M_\lambda (M_\ell - \bar{M}_\ell)) E(T_\kappa (T_k - \bar{T}_k)) = E(M_\lambda) E(M_\ell - \bar{M}_\ell) E(T_\kappa (T_k - \bar{T}_k)) = 0,$$

and similarly for $k \neq \kappa$. Finally, for $\ell = \lambda, k = \kappa$:

$$E(M_\ell (M_\ell - \bar{M}_\ell)) E(T_k (T_k - \bar{T}_k)) = D(M_\ell) D(T_k),$$

which completes the proof. ■

We can now prove the main theorem.

Proof of Theorem 4.7. By definition of random variables $\mathbf{G}, \mathbf{M}_\ell$ and \mathbf{T}_k :

$$\mathbf{G} = \sum_{\lambda, \kappa} \alpha_{\lambda\kappa} \mathbf{M}_\lambda \mathbf{T}_\kappa + \varepsilon.$$

Therefore,

$$\begin{aligned} \text{cov}(\mathbf{G}, (\mathbf{M}_\ell - \overline{\mathbf{M}_\ell})(\mathbf{T}_k - \overline{\mathbf{T}_k})) = \\ \sum_{\lambda, \kappa} \alpha_{\lambda\kappa} \text{cov}(\mathbf{M}_\lambda \mathbf{T}_\kappa, (\mathbf{M}_\ell - \overline{\mathbf{M}_\ell})(\mathbf{T}_k - \overline{\mathbf{T}_k})) + \text{cov}(\varepsilon, (\mathbf{M}_\ell - \overline{\mathbf{M}_\ell})(\mathbf{T}_k - \overline{\mathbf{T}_k})), \end{aligned} \quad (4.30)$$

because of the linearity (2.21) of the covariance operation. The last term in the sum (4.30) is zero:

$$\text{cov}(\varepsilon, (\mathbf{M}_\ell - \overline{\mathbf{M}_\ell})(\mathbf{T}_k - \overline{\mathbf{T}_k})) = 0,$$

because ε is independent of \mathbf{M}_ℓ and \mathbf{T}_k .

All the remaining terms can be evaluated using Lemma 4.8. It follows that the only nonzero term in the sum (4.30) is the one where $\lambda = \ell$ and $\kappa = k$. The whole equation now becomes

$$\text{cov}(\mathbf{G}, (\mathbf{M}_\ell - \overline{\mathbf{M}_\ell})(\mathbf{T}_k - \overline{\mathbf{T}_k})) = \alpha_{\ell k} \mathbf{D}(\mathbf{M}_\ell) \mathbf{D}(\mathbf{T}_k),$$

from which the result follows. ■

Note that the assumption of pairwise uncorrelation is not very unrealistic. Indeed, experiments show that the motif presence vectors for all reasonably different motifs are nearly perfectly uncorrelated on real biological data. The expression profiles of transcription factors do often have a degree of correlation, however on average it is rather low – less than 0.33 in 90% of cases, see Section 5.3. Also note that we do not impose any other constraints on the distributions of the variables.

In return for this mild assumption we get a method of estimation of $\alpha_{\ell k}$, that *only requires the data about motif m_ℓ and TF t_k* . That means, that even if some of the important motifs or TFs are missing from our dataset, we can still use this method to find out the parameters $\alpha_{\ell k}$ for those motifs and TFs that are available.

The computation of a single coefficient with this method requires a covariance computation involving the whole matrix \mathbf{G} , therefore to estimate the whole matrix \mathbf{A} , $O(n_G n_M n_A n_T)$ operations need to be performed. It is one order of magnitude less efficient than the least squares or ridge regression estimate, but still quite tolerable for many datasets. This method lends itself easily to nearly unlimited parallelization, i.e., each coefficient can be computed independently of the others, and the covariation computation for each coefficient is highly parallelizable.

Additionally, it turns out that it is possible to compute a good approximation to the correlation-based estimate efficiently using the familiar least-squares technique.

Centered Least Squares

Let $(\mathbf{G}, \mathbf{M}_\ell, \mathbf{T}_k)$ be $\mathbf{G}=\mathbf{MAT}$ -distributed random variables that satisfy the conditions of Theorem 4.7. Let $(\mathbf{G}, \mathbf{M}, \mathbf{T})$ be a $\mathbf{G}=\mathbf{MAT}$ dataset obtained as a sample of these variables. It turns out that if we apply the least squares method to the *centered* versions of matrices \mathbf{G} , \mathbf{M} and \mathbf{T} , we shall obtain consistent estimates for $\alpha_{\ell k}$.

In the following, let $\overline{\mathbf{M}}^{(*\ell)}$ denote the average of the values in the ℓ -th column of \mathbf{M} , let $\overline{\mathbf{T}}^{(k*)}$ denote the average of the k -th row of \mathbf{T} and let $\overline{\mathbf{G}}$ denote the average of all elements of \mathbf{G} . Let \mathbf{M}' be the column-wise centered version of the matrix \mathbf{M} , \mathbf{T}' be the row-wise centered version of the matrix \mathbf{T} , and \mathbf{G}' be the centered version of the matrix \mathbf{G} . That is,

$$(\mathbf{M}')_{i\ell} = M_{i\ell} - \overline{\mathbf{M}}^{(*\ell)}, \quad (\mathbf{T}')_{kj} = T_{kj} - \overline{\mathbf{T}}^{(k*)}, \quad (\mathbf{G}')_{ij} = G_{ij} - \overline{\mathbf{G}}.$$

Finally, let $\hat{\mathbf{A}}'_{\text{LS}}$ be the least-squares estimate for the model:

$$\hat{\mathbf{G}}' = \mathbf{M}' \mathbf{A} \mathbf{T}'.$$

Then the following result holds.

Theorem 4.9 (Centered least squares) *Let $(\mathbf{G}, \mathbf{M}_\ell, \mathbf{T}_k)$ be random variables, satisfying the conditions of Theorem 4.7. Let $(\mathbf{G}, \mathbf{M}, \mathbf{T})$ be a sample, obtained from these variables and let $\hat{\mathbf{A}}'_{\text{LS}}$ be obtained as described above. Then as the sample size increases, i.e., $n_{\mathbf{G}}, n_{\mathbf{A}} \rightarrow \infty$, the elements of the matrix $\hat{\mathbf{A}}'_{\text{LS}}$ converge almost surely to the true model parameters $(\alpha_{\ell k})$.*

Proof. First, note that in the process $n_{\mathbf{G}} \rightarrow \infty$ the matrix $\frac{1}{n_{\mathbf{G}}} \mathbf{M}'^T \mathbf{M}'$ converges to the matrix of motif covariances. Indeed,

$$\left(\frac{1}{n_{\mathbf{G}}} \mathbf{M}'^T \mathbf{M}' \right)_{\ell\lambda} = \frac{1}{n_{\mathbf{G}}} \sum_i (M_{i\ell} - \overline{\mathbf{M}}^{(*\ell)}) (M_{i\lambda} - \overline{\mathbf{M}}^{(*\lambda)}),$$

which, by strong law of large numbers, converges almost surely to $\text{cov}(\mathbf{M}_\ell, \mathbf{M}_\lambda)$, and therefore

$$\left(\frac{1}{n_{\mathbf{G}}} \mathbf{M}'^T \mathbf{M}' \right)_{\ell\lambda} \xrightarrow{\text{a.s.}} \text{cov}(\mathbf{M}_\ell, \mathbf{M}_\lambda) = \begin{cases} \text{D}(\mathbf{M}_\ell), & \text{if } \ell = \lambda, \\ 0, & \text{otherwise.} \end{cases} \quad (4.31)$$

Similarly, in the process $n_{\mathbf{A}} \rightarrow \infty$:

$$\left(\frac{1}{n_{\mathbf{A}}} \mathbf{T}' \mathbf{T}'^T \right)_{k\kappa} \xrightarrow{\text{a.s.}} \text{cov}(\mathbf{T}_k, \mathbf{T}_\kappa) = \begin{cases} \text{D}(\mathbf{T}_k), & \text{if } k = \kappa, \\ 0, & \text{otherwise.} \end{cases} \quad (4.32)$$

Thus, the matrices $\mathbf{M}'^T \mathbf{M}'$ and $\mathbf{T}' \mathbf{T}'^T$ converge with probability 1 to diagonal matrices with nonzero elements on the diagonal. The latter matrices are invertible and therefore, for sufficiently large n_G and n_A , the matrices $\mathbf{M}'^T \mathbf{M}'$ and $\mathbf{T}' \mathbf{T}'^T$ are also invertible, because the determinants of those matrices must converge to nonzero values. It follows that for sufficiently large n_G and n_A the least squares estimate $\hat{\mathbf{A}}'_{\text{LS}}$ can be computed using equation (4.2):

$$\begin{aligned} \hat{\mathbf{A}}'_{\text{LS}} &= (\mathbf{M}'^T \mathbf{M}')^{-1} \mathbf{M}'^T \mathbf{G}' \mathbf{T}'^T (\mathbf{T}' \mathbf{T}'^T)^{-1} \\ &= \left(\frac{1}{n_G} \mathbf{M}'^T \mathbf{M}' \right)^{-1} \left(\frac{1}{n_G n_A} \mathbf{M}'^T \mathbf{G}' \mathbf{T}'^T \right) \left(\frac{1}{n_A} \mathbf{T}' \mathbf{T}'^T \right)^{-1}. \end{aligned} \quad (4.33)$$

Finally, consider the matrix $\frac{1}{n_G n_A} \mathbf{M}'^T \mathbf{G}' \mathbf{T}'^T$:

$$\left(\frac{1}{n_G n_A} \mathbf{M}'^T \mathbf{G}' \mathbf{T}'^T \right)_{\ell k} = \frac{1}{n_G n_A} \sum_{i,j} (G_{ij} - \overline{G})(M_{i\ell} - \overline{M}^{(*\ell)})(T_{kj} - \overline{T}^{(k*)}),$$

which converges to $\text{cov}(\mathbf{G}, (\mathbf{M}_\ell - \overline{\mathbf{M}}_\ell)(\mathbf{T}_k - \overline{\mathbf{T}}_k))$:

$$\left(\frac{1}{n_G n_A} \mathbf{M}'^T \mathbf{G}' \mathbf{T}'^T \right)_{\ell k} \xrightarrow{\text{a.s.}} \text{cov}(\mathbf{G}, (\mathbf{M}_\ell - \overline{\mathbf{M}}_\ell)(\mathbf{T}_k - \overline{\mathbf{T}}_k)). \quad (4.34)$$

The equations (4.31), (4.32) and (4.34) demonstrate that all terms on the right side of the equation (4.33) converge. Therefore, $\hat{\mathbf{A}}'_{\text{LS}}$ must converge too:

$$(\hat{\mathbf{A}}'_{\text{LS}})_{\ell k} \xrightarrow{\text{a.s.}} \frac{1}{D(\mathbf{M}_\ell)} \cdot \text{cov}(\mathbf{G}, (\mathbf{M}_\ell - \overline{\mathbf{M}}_\ell)(\mathbf{T}_k - \overline{\mathbf{T}}_k)) \cdot \frac{1}{D(\mathbf{T}_k)} = \alpha_{\ell k}$$

by Theorem 4.7. ■

Note that this is a rather unintuitive result. It states that in order to estimate the $\mathbf{G}=\text{MAT}$ parameters for given matrices $\mathbf{G}, \mathbf{M}, \mathbf{T}$ we can instead estimate them for a *completely different* model, not the one that was used to generate these matrices.

For example, let $\mathbf{G}, \mathbf{M}, \mathbf{A}, \mathbf{T}$ be the matrices, satisfying

$$\mathbf{G} = \mathbf{MAT}.$$

and let $\mathbf{G}', \mathbf{M}', \mathbf{T}'$ the properly centered versions of these matrices. Then, in general,

$$\mathbf{G}' \neq \mathbf{M}' \mathbf{A} \mathbf{T}'.$$

This result allows us to compute an approximation to the estimate (4.27) in time $O(n^3)$, where $n = \max(n_G, n_A, n_M, n_T)$. The experiments in Section 5.5.3 demonstrate that this approximation can be quite good and especially useful for

datasets where some motifs and TFs are missing.

4.5 Randomization-based Attribute Selection

Finally, we propose three randomization techniques that aim to identify the most “interesting” coefficients, rather than estimate their exact values. All of them are designed as wrappers around an existing method for parameter estimation. In the following we denote this method by $A(\cdot)$. In formal terms, let A denote some function, that, when given matrices \mathbf{G} , \mathbf{M} and \mathbf{T} , outputs an estimate for G=MAT parameters:

$$\hat{\mathbf{A}} = (\hat{\alpha}_{\ell k}) \leftarrow A(\mathbf{G}, \mathbf{M}, \mathbf{T}).$$

We consider a probabilistic view on G=MAT, presented in Definition 4.6. That is, we regard the given matrices \mathbf{G} , \mathbf{M} and \mathbf{T} as realizations of G=MAT-distributed random variables \mathbf{G} , \mathbf{M}_ℓ and \mathbf{T}_k . In this case, the estimated parameters $\hat{\alpha}_{\ell k}$ can also be regarded as realizations of some random variables $\mathbf{A}_{\ell k}$. In particular, we write $\mathbf{A} = A(\mathbf{G}, \mathbf{M}, \mathbf{T})$ to indicate, that the variables $\mathbf{A}_{\ell k}$ are obtained by generating a G=MAT dataset $(\mathbf{G}, \mathbf{M}, \mathbf{T})$ as a realization of variables $(\mathbf{G}, \mathbf{M}, \mathbf{T})$ and using the function $A(\cdot)$ to estimate its parameters.

The first method we propose is based on the idea, that the estimated values for the most interesting coefficients should be positive with high probability.

Definition 4.7 (Positivity-based attribute selector) *Let $\mathbf{A} = A(\mathbf{G}, \mathbf{M}, \mathbf{T})$ be the G=MAT parameter estimate obtained using method $A(\cdot)$.*

Define for each (ℓ, k) the positivity weight of attribute $\alpha_{\ell k}$ as the probability that its estimated value will be greater than zero:

$$w_{\ell k} = \Pr[\mathbf{A}_{\ell k} > 0]. \quad (4.35)$$

In practice we can estimate the positivity weight in a cross-validation-like manner: by selecting random subsets of rows and columns of \mathbf{G} together with corresponding rows of \mathbf{M} and columns of \mathbf{T} , obtaining the estimate $\hat{\mathbf{A}}$ and counting for each coefficient the fraction of cases when it turns out to be greater than zero.

Another way to assess the relevance of the parameters is to compare the true estimated values with estimates obtained on a randomized dataset.

Definition 4.8 (P-value-based attribute selector) *Let \mathbf{G}^{RND} be a matrix, obtained by randomly permuting rows and columns of \mathbf{G} . Let $\mathbf{A}^{\text{RND}} = A(\mathbf{G}^{\text{RND}}, \mathbf{M}, \mathbf{T})$ be the estimate obtained on this randomized dataset.*

Define for each (ℓ, k) the p-value score of the coefficient $\alpha_{\ell k}$ as

$$p_{\ell k} = P(\mathbf{A}_{\ell k}^{\text{RND}} < \hat{\alpha}_{\ell k}).$$

In practice, we obtain the p-value estimate by shuffling the values of \mathbf{G} several times and for each coefficient counting the fraction of iterations, where the randomized estimate is less than the estimate obtained on the non-permuted data.

This way, we obtain for each $\alpha_{\ell k}$ an estimate of how improbable would it be to have a parameter value that large on a randomized dataset. Therefore, parameters $\alpha_{\ell k}$, that turn out to have p-values close to 1 can be believed to have a significant effect on the true output \mathbf{G} .

A similar, yet slightly more robust approach would be to compute the *Z-score* of true estimates with respect to the distribution of the estimates on random data.

Definition 4.9 (Z-score-based attribute selector) *Let $\mathbf{A}_{\ell k}^{\text{RND}}$ be the randomized estimate obtained as described in previous definition.*

Define for each (ℓ, k) the z-score of attribute $\alpha_{\ell k}$ as

$$z_{\ell k} = \frac{\hat{\alpha}_{\ell k} - \mathbb{E}(\mathbf{A}_{\ell k}^{\text{RND}})}{\sqrt{\mathbb{D}(\mathbf{A}_{\ell k}^{\text{RND}})}}, \quad (4.36)$$

where $\mathbb{E}(\cdot)$ denotes the mean and $\mathbb{D}(\cdot)$ – variance of a random variable.

This way, for each estimated coefficient we obtain a score of how large is this estimate in comparison to estimates, obtained on randomized data. The computation of z-scores is similar to that of p-values. This time, instead of counting the fraction of iterations where the true estimate is greater than the randomized one, we compute the mean and standard deviation of the randomized estimates and use these values to normalize the true estimate according to equation (4.36).

In our experiments we use the ridge-regression estimation method for the function $A(\cdot)$, because it is the most efficient of the available strategies. In this case all three of the methods above can be implemented in time $O(cn^3)$, where c is the number of randomization iterations.

4.6 Example (continued)

Let us continue the example from Section 3.6 and illustrate how the proposed methods would restore the “regulatory relations” \mathbf{A} , from the matrices \mathbf{G} , \mathbf{M} and \mathbf{T} .

Least squares regression. As the number of TFs in the example is greater than the number of arrays, the matrix $\mathbf{T}\mathbf{T}^T$ is rank-deficient and the solution to (4.1) is not unique. We therefore employ the minimum-norm solution given by

equation (4.13). We obtain then:

$$\hat{\mathbf{A}}_{\text{LS}*} = \begin{array}{c|ccccc} & t_1 & t_2 & t_3 & t_4 & t_5 \\ \hline m_1 & 0.75 & -0.25 & 0.25 & 0.25 & 0 \\ m_2 & 0.75 & -1.25 & 0.25 & -0.75 & 0 \\ m_3 & 0.25 & 0.25 & -0.25 & 0.75 & 0 \\ m_4 & -0.25 & -0.25 & 0.25 & -0.75 & 1 \\ m_5 & 0 & 0 & 0 & 0 & 2 \end{array} .$$

Naturally, the estimated parameter values are not exactly equal to the true ones. Nonetheless, they do show quite nicely, which coefficients have the most importance. For example, the parameters with the largest positive and largest negative values are correctly identified, 4 out of 5 parameters with (true) values 1 have their values estimated as ≥ 0.75 , etc.

Regularized regression. As already noted, the least squares estimate can behave badly when data is noisy and inputs are correlated. Regularized estimates are more robust in this setting, because they penalize unnecessarily large model coefficients. To illustrate this issue, we shall modify the example by considering the following matrix \mathbf{T} :

$$\mathbf{T} = \begin{array}{c|cccc} & a_1 & a_2 & a_3 & a_4 \\ \hline t_1 & 1 & 1 & 1 & 0 \\ t_2 & 1 & 1 & 1 & 0.25 \\ t_3 & 1 & 1 & 1 & 0.5 \\ t_4 & 1 & 1 & 0 & 0.75 \\ t_5 & 1 & 1 & 0 & 1 \end{array} . \quad (4.37)$$

We recompute the new value for the matrix \mathbf{G} and add gaussian noise with standard deviation 0.5 to its elements. Now that the data is noisy and the TF expression profiles are so similar, any estimation method will certainly have difficulties in determining which of the TFs are actually responsible for gene expression. Regularized methods, however, should outperform the straightforward least squares regression. We shall demonstrate this by considering a simple problem of detecting the parameter with the largest value (which is, as we know, $\alpha_{5,5}$). We shall measure over 1000 iterations (each time with a different \mathbf{G} due to noise), how often the least squares estimate reports $\hat{\alpha}_{5,5}$ as the largest coefficient, and how it compares to the performance of the estimates obtained using ℓ_2 -norm regularization, ridge regression and sparse ℓ_1 -norm regularization. We run the algorithms with a number of different values of the regularization parameter (where for ridge regression we only consider λ_{T} and set $\lambda_{\text{M}} = 0$). The results are presented in Figure 4.1, that clearly demonstrates the advantages of regularization.

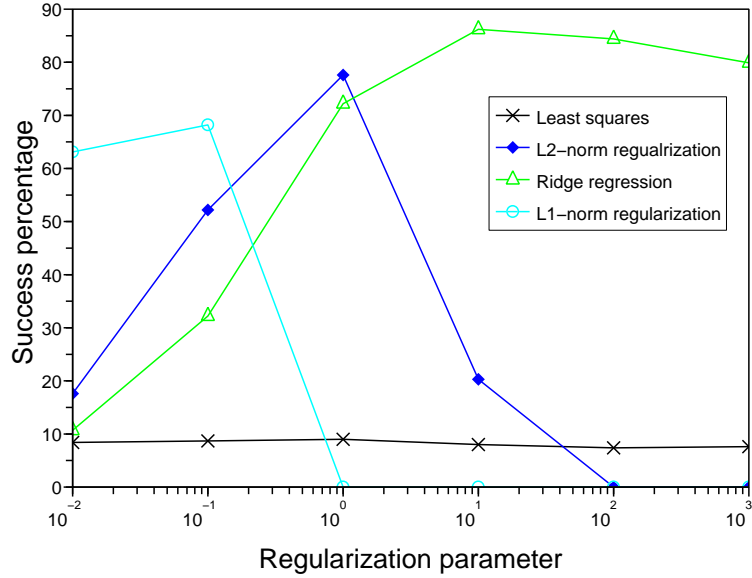


Figure 4.1: The graphs show how often, on average, different estimation methods report $\hat{\alpha}_{5,5}$ as the largest coefficient (methods were tested over a range of values for the regularization parameter λ). Results clearly demonstrate how regularized estimates outperform ordinary least squares regression.

It should be noted though, that if the input variables (i.e., TF expression profiles and motif presence vectors) were orthogonal, regularization would result in just a uniform scaling (in case of ℓ_2 -norm) or shrinking (in case of ℓ_1 -norm) of parameters and would therefore provide no significant benefits for our purposes.

Randomization-based methods. The idea of the proposed randomization-based methods is to take an existing method $A(\cdot)$, and “improve” it for the purposes of attribute selection using randomization. Let us take the same task of detecting the maximum-value coefficient $\alpha_{5,5}$ and compare the performance of the ridge regression estimate, already considered above, with a P-value-based and a Z-score-based estimators built with it. As in previous case, we run 1000 iterations, each time adding $N(0, 0.5)$ Gaussian noise to \mathbf{G} , and compare the results for several values of the regularization parameter λ_T . Figure 4.2 depicts the results. It is clearly seen how randomization helps to increase the performance for at least some values of the regularization parameter. As it will become clear further, in other settings randomization-based techniques can show even better results.

Correlation-based estimate. To illustrate the properties of the correlation-based estimate, we shall consider a noise-free, full-rank dataset. We shall use

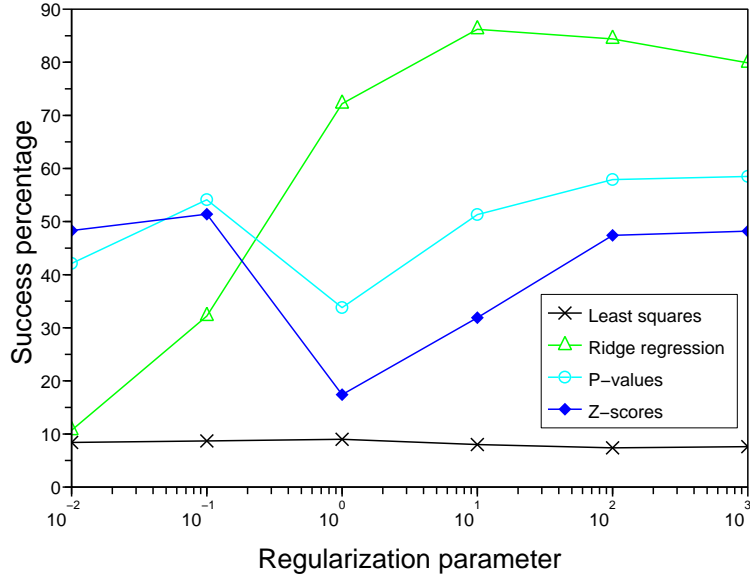


Figure 4.2: The graphs show how often, on average, different estimation methods report $\hat{\alpha}_{5,5}$ as the largest coefficient. Results demonstrate how randomization can sometimes boost the performance of a non-randomized method.

the original example matrices \mathbf{A} , \mathbf{M} and \mathbf{T} from equations (3.3), (3.4) and (3.5) respectively, and append one additional column to the matrix \mathbf{T} :

$$\mathbf{T} = \begin{array}{c|ccccc} & a_1 & a_2 & a_3 & a_4 & a_5 \\ \hline t_1 & 1 & 0 & 1 & 0 & 0 \\ t_2 & 0 & 1 & 0 & 1 & 0 \\ t_3 & 1 & 1 & 0 & 1 & 0 \\ t_4 & 0 & 0 & 1 & 0 & 1 \\ t_5 & 0 & 1 & 0 & 0 & 1 \end{array}.$$

The resulting full-rank dataset contains enough information for the least squares method to restore the true \mathbf{A} perfectly, yet the performance of the correlation-based estimate is rather poor. The estimated parameters are the following:

$$\hat{\mathbf{A}}_{\text{CORR}} = \begin{array}{c|ccccc} & t_1 & t_2 & t_3 & t_4 & t_5 \\ \hline m_1 & 2.5 & -1.7 & 0 & 0 & -2.1 \\ m_2 & 1.8 & -1.0 & 0.8 & -0.8 & -1.5 \\ m_3 & -1.8 & 1.0 & -0.8 & 0.8 & 1.5 \\ m_4 & -0.6 & 0.8 & 1.1 & -1.1 & 0.3 \\ m_5 & -1.6 & 0.4 & -1.0 & 1.0 & 2.1 \end{array}.$$

The reason for this low performance lies in two properties of our dataset. First of all, the method does not tolerate *any* strong correlations of the input variables, that are present in our data. We can clearly see here, how the perfectly (negatively) correlated columns m_2 and m_3 of the matrix \mathbf{M} result in the corresponding rows of the matrix \mathbf{A} being also perfectly correlated. For the similar reason the columns t_3 and t_4 make no sense. Secondly, the method requires a much larger dataset to obtain meaningful estimates. That is why it is close to useless on our small example. As we shall see later, in a different setting this method can significantly outperform other approaches.

Chapter 5

Model Performance Analysis

Experience is something you don't get until just after you need it.

Murphy's Law

In the previous chapter we have presented a number of methods for estimating parameters of G=MAT model from data. In this chapter we analyze the applicability of these methods to biological data. We start by considering a classical yeast microarray dataset by Spellman et al. [SSZ⁺98]. The analysis of this dataset highlights some important issues, which we then study in detail using a similar artificially generated dataset. Finally, we use the artificial dataset to compare the performance of different G=MAT estimation methods.

5.1 The Spellman Dataset

For the first application of our method on real biological data we consider the microarray dataset of Spellman et al. [SSZ⁺98]. The dataset contains 77 microarray experiments, measuring the expression of 6178 yeast genes at different points of the cell life cycle.

Data preparation Some preprocessing was required to obtain the matrices **G** and **T**.

- The dataset contained missing values. We imputed them using the *KNN-impute* algorithm [TCS⁺01] implemented in Matlab.
- Of all the genes present in the dataset, 318 had the *transcription regulator activity* GO annotation (according to the data of the *SGD* project [SGD]). We selected these 318 genes as transcription factors. The expression values of these TFs were collected in the 318×77 matrix **T**.

- From other 5860 genes we selected 5766 for which the genomic sequence was available via the *SGD* website. The expression values of these genes were collected in the 5766×77 matrix **G**.

Next, we prepared the motif matrix **M** as follows.

- For each target gene we downloaded its promoter region: 800-nucleotide-long genomic sequence upstream of the gene’s translation start site. We obtained the data from the *SGD* website.
- We used the TRANSFAC database of regulatory motifs and selected the 36 known yeast motifs there.
- We used the **storm** [SSZ07] tool to match TRANSFAC motifs on the promoter. As a result we obtained the 5766×36 binary matrix **M**.

G=MAT analysis Once the data is presented in the form of the **G**, **M** and **T** matrices, we are free to apply any of the G=MAT parameter estimation methods. We chose to apply the least-squares estimate (4.13), because this method is the most straightforward and requires setting no parameters. Recall that each coefficient $\hat{\alpha}_{\ell k}$ of the resulting parameter matrix $\hat{\mathbf{A}}_{\text{LS}^*}$ measures a putative association between motif m_ℓ and TF t_k . Table 5.1 presents the pairs of motifs and TFs that correspond to the 5 coefficients of the matrix $\hat{\mathbf{A}}_{\text{LS}^*}$ with the largest values.

5.2 Evaluation of Results

As we have chosen a true biological dataset for the experiment, we can not know the corresponding “true” parameter matrix **A**. As a result, we lack any gold standard for objectively assessing the relevance of the obtained results and have to limit ourselves with the following two options:

- evaluating the biological meaningfulness of the results manually,
- examining the predictive performance of the model.

The two evaluations seem to differ radically.

Biological relevance of the results. The top-scoring motif-TF pairs, presented in Table 5.1, make complete sense. In particular, the top three entries associate the known binding site of the Gal4p transcription factor with the Gal1p, Gal3p and Gal80p proteins. This is in perfect accordance with current biological knowledge [LVZ95].

- Overexpression of galactokinase-coding gene *GAL1* results in activation of Gal4p protein [BOH90, BH92]. This process is probably the reason for strong

Motif	TF	Score
F\$GAL4_01 Binding site for GAL4.	<i>GAL1</i> Galactokinase, phosphorylates alpha-D-galactose to alpha-D-galactose-1-phosphate in the first step of galactose catabolism.	0.30
F\$GAL4_01 Binding site for GAL4.	<i>GAL3</i> Transcriptional regulator involved in activation of the GAL genes in response to galactose.	0.26
F\$GAL4_01 Binding site for GAL4.	<i>GAL80</i> Transcriptional regulator involved in the repression of GAL genes in the absence of galactose.	0.18
F\$MCM1_02 Binding site for MCM1 and SFF.	<i>SFG1</i> Nuclear protein, putative transcription factor required for growth of superficial pseudohyphae (which do not invade the agar substrate) but not for invasive pseudohyphal growth.	0.12
F\$MCM1_02 Binding site for MCM1 and SFF.	<i>ACE2</i> Transcription factor that activates expression of early G1-specific genes, localizes to daughter cell nuclei after cytokinesis and delays G1 progression in daughters, localization is regulated by phosphorylation.	0.12

Table 5.1: G=MAT analysis of the Spellman dataset. The table presents five motif-TF pairs having the largest values of the corresponding parameters $\hat{\alpha}_{\ell k}$. Motifs are in the leftmost column and are identified by their TRANSFAC identifiers. The middle column contains TFs, which are identified by their gene names. The rightmost column contains the corresponding values $\hat{\alpha}_{\ell k}$.

positive association between *GAL1* expression and presence of the Gal4p binding site: high expression of *GAL1* induces expression of Gal4p, which binds to that motif.

- *GAL3* is a gene highly similar to *GAL1* [PRHR00]. The corresponding protein Gal3p forms a complex with Gal80p and thus relieves inhibition of Gal4p [LVZ95]. This explains the positive association of *GAL3* with the Gal4p binding motif.
- Gal80p is a transcriptional inhibitor of GAL genes (in particular, *GAL4*). Inhibition is relieved by Gal1p or Gal3p binding [TRR02]. Therefore, we might expect Gal80p to be strongly *negatively* associated with the Gal4p binding site: the higher the expression of Gal80p, the more repressed is Gal4p, the less effect it has on the genes with the corresponding motif in the promoter. Our analysis, on the contrary, indicates a positive association. This might be due to the complex regulatory feedback loops involved in the

regulation of the whole family of GAL genes. Such nonlinear relations cannot be accounted for by our simple linear model. Nonetheless, we consider the discovered relation a success rather than a failure of the analysis.

It is very unlikely that this result (three obviously successful pairs among the top five) could be obtained by chance. To verify that, we considered all motif-TF pairs, where the TF belonged to the same family of proteins as the binding factor for the motif. For example, in the above case, *GAL1*, *GAL3* and *GAL80* all belong to the same family as *GAL4* which is the binding factor for motif F\$GAL4_01. There were 67 such matching pairs among the 36×318 coefficients in the parameter matrix. If we were to pick 5 pairs at random, we would expect less than 0.03 hits on average. Getting 3 hits instead exceeds the expectations more than 100-fold.

Predictive performance. We could expect that if our model parameters have biological relevance, the model should predict well. Unfortunately, it turns out that the predictive performance of the constructed model is very poor. The coefficient of determination (see Section 2.4.2) of the model is barely above 0.05, see below.

Mean squared error:	0.1494
Variance of \mathbf{G} :	0.1576
Coefficient of determination, R^2 :	0.0520

It might seem surprising, that a model which produces biologically meaningful results predicts so badly. In the following we study this issue on an artificially generated dataset and propose an explanation for this phenomenon.

5.3 The Artificial Dataset

To study the properties of $\mathbf{G}=\mathbf{MAT}$ estimates, we need a dataset that is similar to a real one yet for which we know the true value of \mathbf{A} . To construct such a dataset, we first studied the statistical properties of the Spellman dataset considered in the previous sections. We then randomly generated matrices \mathbf{M} , \mathbf{A} and \mathbf{T} to obtain a dataset with similar properties.

We generated a dataset with 100 transcription factors, 100 motifs and 1000 genes.

The motif matrix \mathbf{M} . The matrix \mathbf{M} is a binary matrix of motif occurrences. As a natural simplification, we regard each column of this matrix as a set of i.i.d. realizations of Bernoulli-distributed random variables. That is:

$$M_{i\ell} \sim \text{B}(p_\ell),$$

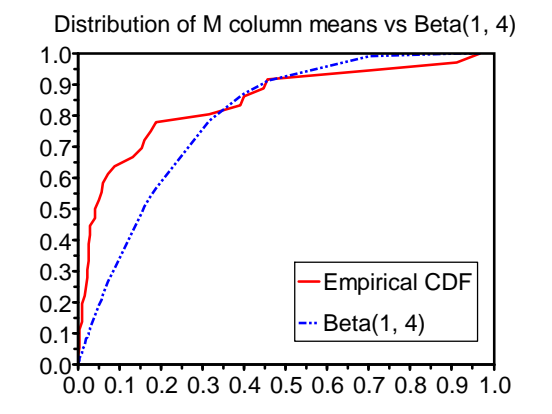


Figure 5.1: The empirical cumulative distribution function of motif presence probabilities p_ℓ , estimated on the Spellman dataset, compared to the cdf of the Beta(1, 4) distribution.

where p_ℓ is the probability of motif m_ℓ presence in a randomly chosen gene.

By analyzing the Spellman dataset we see that the average absolute correlation of the columns of the matrix \mathbf{M} is less than 0.02. We can therefore safely assume the columns to be independent. Finally, we observe that the distribution of probabilities p_ℓ can be reasonably well described by a Beta(1, 4) distribution, see Figure 5.1. Therefore, we generate the artificial matrix \mathbf{M} as follows:

$$\begin{aligned} p_\ell &\leftarrow \text{Beta}(1, 4) \text{ independently for each column } \ell, \\ M_{i\ell} &\leftarrow \text{B}(p_\ell) \text{ independently for each row } i. \end{aligned}$$

The TF expression matrix \mathbf{T} . The matrix \mathbf{T} of the Spellman dataset is more difficult to model than the motif matrix. Visual inspection (see Figure 5.2) shows no clear column or row-based structure. The mean absolute correlation of matrix rows is 0.16, and of the columns: 0.14. We chose to model \mathbf{T} as a set of independent columns. Even though it does not correspond exactly to what we observe on the Spellman dataset, the assumption of independent microarray experiments is quite viable.

The distribution of values in each column of \mathbf{T} can be very well described by normal distribution (see Figure 5.3). Examination of the means and standard deviations of different \mathbf{T} columns shows that these can also be reasonably well described by normal distributions (Figure 5.4). Therefore, we generate the artificial

matrix \mathbf{T} as follows:

$$\begin{aligned} m_j &\leftarrow N(0, 0.08^2) \text{ independently for each column } j, \\ \sigma_j &\leftarrow N(0.35, 0.09^2) \text{ independently for each column } j, \\ T_{kj} &\leftarrow N(m_j, \sigma_j^2) \text{ independently for each row } k. \end{aligned}$$

The parameter matrix \mathbf{A} . We generate matrix \mathbf{A} as follows: first set all values to zero, then choose 3% of the coefficients randomly and set their values to 0.05. The reason for the choice of such a strategy is the following. Firstly, we believe the true \mathbf{A} to contain a very small number of nonzero elements. Secondly, we wish the distribution of values in \mathbf{T} to be reasonably similar to the distribution of values in \mathbf{G} in terms of mean and variance. Experiments showed that this can be achieved by setting the nonzero values of \mathbf{A} to 0.05.

The gene expression matrix \mathbf{G} . The matrix \mathbf{G} is generated according to the model: $\mathbf{G} = \mathbf{M}\mathbf{A}\mathbf{T}$. In some experiments we also add noise to it, but we discuss this further in the text.

5.4 Prediction versus Attribute Selection

As we saw in Section 5.2, a $\mathbf{G}=\mathbf{M}\mathbf{A}\mathbf{T}$ model can discover relevant parameters without achieving a satisfactory predictive performance. In this section we explain this phenomenon by demonstrating both experimentally and theoretically how and why this can happen. We start with an artificial dataset introduced in the previous section and consider two factors that influence model prediction error: noise and incomplete data. We show that although these factors can significantly decrease the model’s predictive ability, they do not prevent the model from discovering relevant parameters. In the following we refer to this capability as *attribute selection*.

5.4.1 Experimental Setup

Let $\mathbf{G}^a, \mathbf{M}^a, \mathbf{A}^a$ and \mathbf{T}^a denote the matrices of the artificial dataset, generated as described Section 5.3. In the experiments that follow, we introduce certain modifications to these matrices (such as add noise to \mathbf{G}^a or drop rows from \mathbf{T}^a), estimate the parameter matrix $\hat{\mathbf{A}}$ using different methods and examine the performance of the estimated models in prediction and attribute selection. We measure this performance as follows.

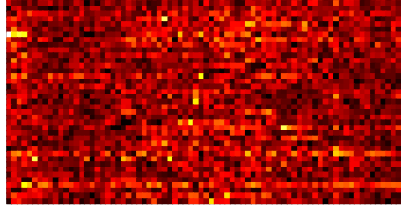


Figure 5.2: Visualization of the first 40 rows of the \mathbf{T} matrix of the Spellman dataset. Dark points denote elements with smaller absolute values.

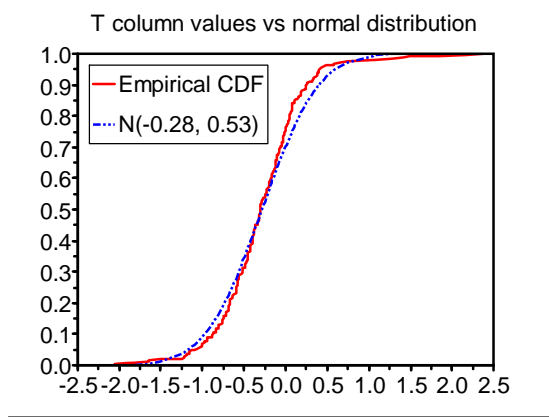


Figure 5.3: The empirical cumulative distribution of values in the second column of \mathbf{T} can be approximated well by a Gaussian distribution $N(-0.28, 0.53)$. The situation is similar for other columns.

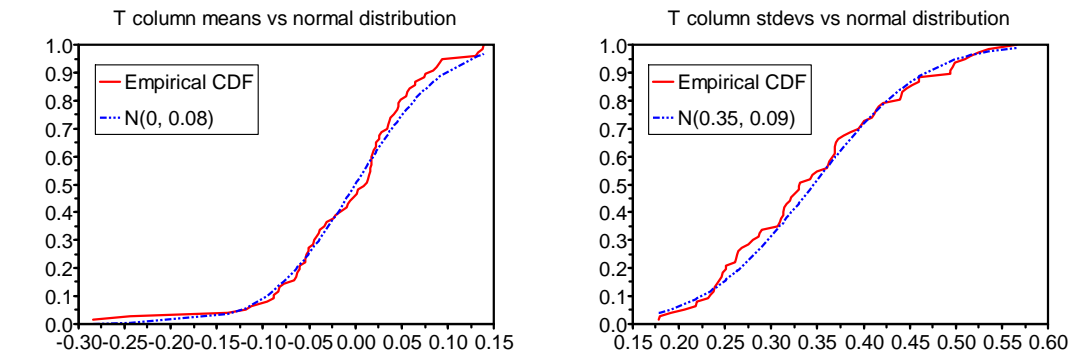


Figure 5.4: The empirical cumulative distribution of the means (left) and standard deviations (right) of \mathbf{T} columns can be well approximated by a Gaussian distribution.

Predictive performance. To measure predictive performance we simply consider the mean squared error of the model:

$$\text{MSE}(\hat{\mathbf{A}}) = \frac{1}{n_G \times n_A} \|\mathbf{G}^a - \mathbf{M}^a \hat{\mathbf{A}} \mathbf{T}^a\|^2.$$

Attribute selection performance. To measure how well the estimated matrix can be used for attribute selection we use the *ROC area under curve (ROC AUC)* statistic (see Section 2.4.2). We consider those model parameters for which the true value $\alpha_{\ell k}^a$ was nonzero (in other words, parameters with values 0.05), as *positives*, and all the rest as *negatives*. We then sort the parameters according to their estimated values $\hat{\alpha}_{\ell k}$ and evaluate the ROC AUC of this sorted list. If the positives all turn out to be on top of the list (i.e., their values estimated as the largest), the value of ROC AUC will be 1. This means the model is perfect for attribute selection. The ROC AUC score of 0.5 indicates a model that is not better than random in guessing the relevant parameters. Finally, the ROC AUC score is 0 if all the positives end up on the bottom of the list. This corresponds to a model that is good for attribute selection, but somehow orders attributes in reverse.

5.4.2 Influence of Noise

Consider the following modification to the matrix \mathbf{G}^a of the artificial dataset:

$$\mathbf{G}^a = \mathbf{M}^a \mathbf{A}^a \mathbf{T}^a + \varepsilon, \quad (5.1)$$

where ε is Gaussian noise with mean 0 and variance σ_ε^2 . This corresponds to incorporating into the data a multitude of independent factors that cannot be accounted for by the model parameters: external influences, nonlinear expression regulation rules, measurement errors, etc. Let us examine how the prediction and attribute selection performance depends on σ_ε^2 . For that we created 11 artificial datasets, differing only in the variance of the noise σ_ε^2 , and applied the least squares method and the ridge regression method with parameters $\lambda_M = \lambda_T = 100$ to these datasets.

Effect on prediction error. When the data is noise-free, i.e., $\sigma_\varepsilon^2 = 0$, the least squares method will restore \mathbf{A}^a precisely and the prediction error will be zero. When $\sigma_\varepsilon^2 > 0$, the least squares estimate is not able to account for most of the noise and the predictive error is proportional to σ_ε^2 . Other methods, such as ridge regression, behave similarly. Their prediction error, by definition, is always greater or equal than that of the least squares estimate. Experiment results are presented in Figure 5.5 (left).

Effect on attribute selection. Figure 5.5 (right) shows how ROC AUC score depends on the noise. As we see, the ROC AUC of the least squares estimate deteriorates quite quickly with the introduction of noise. This is a natural result of *overfitting* that takes place here due to the improperly large number of parameters of the model. If the number of TFs n_T was less than the number of arrays n_A , the ROC AUC of the least squares estimate would deteriorate much slower. We illustrate this by considering an artificial dataset with only 50 transcription factors instead of 100, the results are presented in Figure 5.6. Note that in both cases the ridge regression estimate avoids overfitting and has a high ROC AUC score despite the noise.

5.4.3 Influence of Incomplete Data

Another important factor that can influence the predictive performance of the model in practice is the incompleteness of data. Let us now generate the noise-free version of the artificial dataset $(\mathbf{G}^a, \mathbf{M}^a, \mathbf{A}^a, \mathbf{T}^a)$ and then drop the first n rows of \mathbf{T}^a and the first n columns of \mathbf{M}^a . In some sense this is similar to adding noise to \mathbf{G}^a , but conceptually this is somewhat different. This time we say that our model is true, we just do not have enough data.

Effect on prediction error. When $n = 0$ the prediction error is 0 and when $n = 100$ the prediction error equals $\|\mathbf{G}^a\|^2$, increasing rather uniformly in between. See Figure 5.7 (left).

Effect on attribute selection. Figure 5.7 shows that when we remove some TFs and motifs from the dataset, thus fitting a model with a smaller number of parameters than needed to explain the data completely, the ROC AUC score still stays satisfactorily high.

5.4.4 Theoretical Justification

In addition to the obvious experimental evidence, we provide an elegant theoretical justification, demonstrating how the model can estimate relevant parameters from incomplete data without being able to predict well. Let us return once more to the example with the Spellman dataset from Section 5.2.

Motifs are the bottleneck. The major bottleneck for model performance on the Spellman dataset was actually the small number of motifs. Indeed, if the number of motifs n_M were greater or equal to the number of genes n_G , model error would necessarily be 0. In our case, however, the number of motifs $n_M = 36$ is significantly smaller than the number of genes $n_G = 5766$. It follows from basic

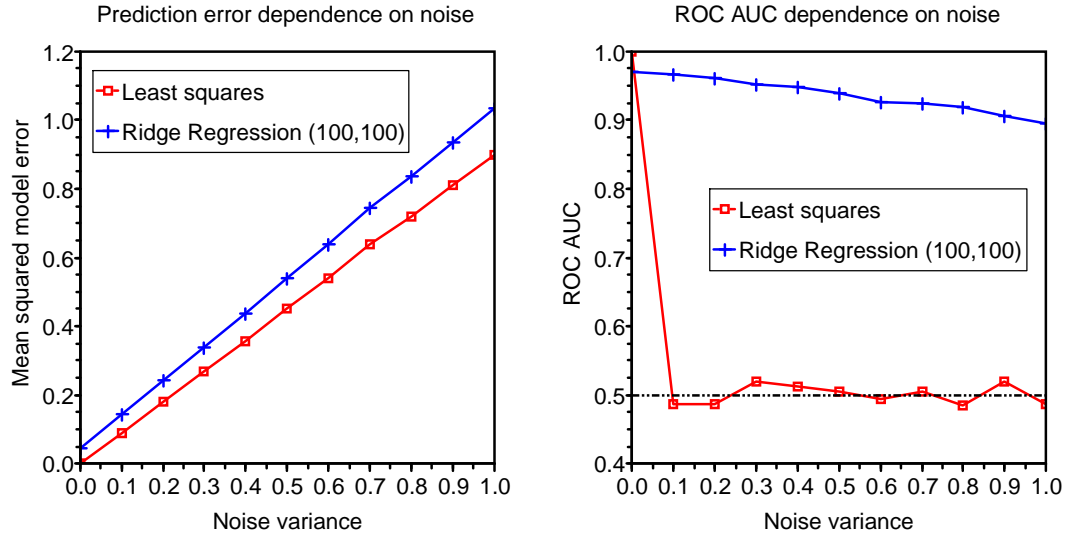


Figure 5.5: Influence of noise on prediction error and ROC AUC. The left plot demonstrates that the mean squared error of the least squares and ridge regression estimates is proportional to the variance of the noise. The right plot shows that the ROC AUC score of the least squares estimate deteriorates rapidly with the introduction of noise due to overfitting, but the ridge regression estimate stays resistant to noise.

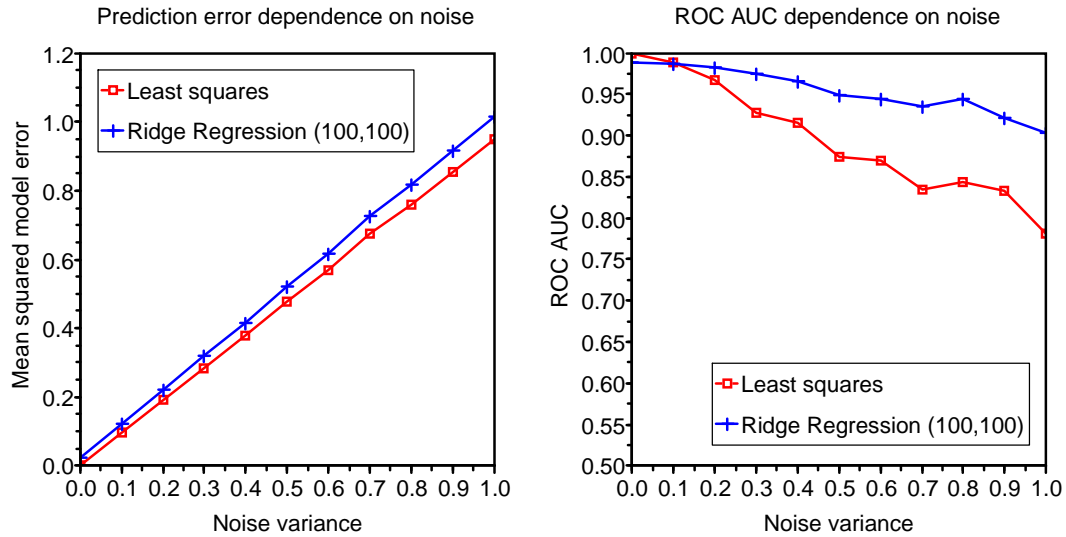


Figure 5.6: Influence of noise on prediction error and ROC AUC. Here we used an artificial dataset with only 50 transcription factors rather than 100. Unlike the situation in Figure 5.5, there is no overfitting here and the least squares estimate can tolerate the noise well. Nevertheless, the ridge regression estimate is still more stable.

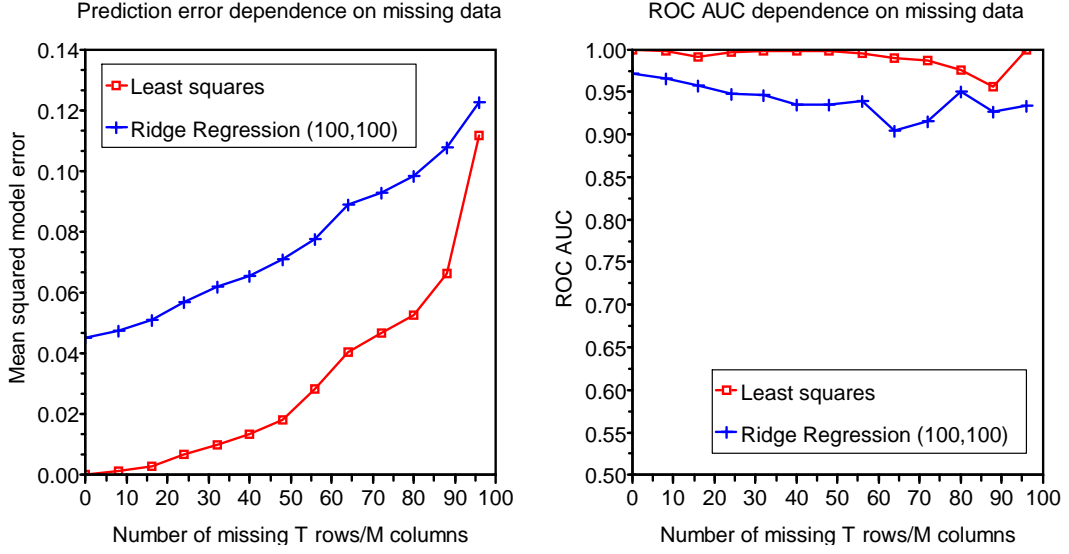


Figure 5.7: Influence of missing data on prediction error and ROC AUC. If we drop columns from \mathbf{T}^a and rows from \mathbf{M}^a , thus hiding some TFs and motifs from the dataset, the model error increases (left) but the ROC AUC stays high (right).

linear algebra, that an arbitrary vector of dimension 5766 can not in general be represented as a linear combination of just 36 base vectors. Therefore, also the $G=MAT$ model error must be high.

To illustrate this more formally, we consider the following model:

$$\hat{\mathbf{G}} = \mathbf{MC}, \quad (5.2)$$

where \mathbf{C} is an $n_M \times n_A$ matrix of parameters. First, note that this model is better than the $G=MAT$ model in terms of prediction.

Theorem 5.1 *Let $\mathbf{G}, \mathbf{M}, \mathbf{T}$ be $G=MAT$ matrices. Let $\hat{\mathbf{C}}_{LS}$ be the least squares fit for the model (5.2) and let*

$$\text{MSE}_{GMC}(\hat{\mathbf{C}}_{LS}) = \frac{1}{n_G \times n_A} \|\mathbf{G} - \mathbf{M}\hat{\mathbf{C}}_{LS}\|^2$$

be the corresponding model error. Let $\hat{\mathbf{A}}_{LS}$ be the $G-MAT$ least squares fit and let

$$\text{MSE}_{GMAT}(\hat{\mathbf{A}}_{LS}) = \frac{1}{n_G \times n_A} \|\mathbf{G} - \mathbf{M}\hat{\mathbf{A}}_{LS}\mathbf{T}\|^2$$

be the corresponding model error. Then

$$\text{MSE}_{GMC}(\hat{\mathbf{C}}_{LS}) \leq \text{MSE}_{GMAT}(\hat{\mathbf{A}}_{LS}).$$

Proof. Let $\hat{\mathbf{C}}_* = \hat{\mathbf{A}}_{\text{LS}}\mathbf{T}$. The error of the G=MC model for such value of \mathbf{C} is then equal to

$$\begin{aligned}\text{MSE}_{\text{GMC}}(\hat{\mathbf{C}}_*) &= \frac{1}{n_{\text{G}} \times n_{\text{A}}} \|\mathbf{G} - \mathbf{M}\hat{\mathbf{C}}_*\|^2 \\ &= \frac{1}{n_{\text{G}} \times n_{\text{A}}} \|\mathbf{G} - \mathbf{M}\hat{\mathbf{A}}_{\text{LS}}\mathbf{T}\|^2 = \text{MSE}_{\text{GMAT}}(\hat{\mathbf{A}}_{\text{LS}})\end{aligned}$$

By definition, $\hat{\mathbf{C}}_{\text{LS}}$ is the matrix of model parameters for which the model error is minimal and therefore

$$\text{MSE}_{\text{GMC}}(\hat{\mathbf{C}}_{\text{LS}}) \leq \text{MSE}_{\text{GMC}}(\hat{\mathbf{C}}_*) = \text{MSE}_{\text{GMAT}}(\hat{\mathbf{A}}_{\text{LS}}).$$

■

Next, note that in the G=MC model (5.2) each column of \mathbf{G} is represented as a linear combination of the columns of \mathbf{M} , with coefficients given in the corresponding column of \mathbf{C} . As we have noted above, this is rather improbable that there exists an exact representation of the 5766-dimensional column of \mathbf{G} in terms of just 36 columns of \mathbf{M} and therefore it is not surprising that with high probability there does not exist a \mathbf{C} for which the error of the G=MC model would be low on the Spellman dataset. As it follows from the above theorem, the error of the G=MAT model must be at least as high as that of the G=MC model. Evaluation on data shows that, in fact, the error of the G=MC model on the Spellman dataset is exactly equal to the error of the G=MAT model. This explains why the bottleneck lies precisely in the low number of motifs: it is just not possible to predict better using a linear model with so few motifs.

Introducing latent motifs. We could improve the predictive performance of the model if we added additional motifs $m_{n_{\text{M}}+1}, m_{n_{\text{M}}+2}, m_{n_{\text{M}}+3}, \dots, m_{n_{\text{G}}}$ to the data. Suppose that these motifs do exist and let the unknown motif occurrence matrix for these new motifs be \mathbf{M}_{new} . The full motif matrix \mathbf{M}_{full} is then the concatenation of the matrices \mathbf{M} and \mathbf{M}_{new} :

$$\mathbf{M}_{\text{full}} = (\mathbf{M} \ \mathbf{M}_{\text{new}}).$$

We also incorporate new rows into the parameter matrix to account for the new motifs. The full parameter matrix \mathbf{A}_{full} is therefore:

$$\mathbf{A}_{\text{full}} = \begin{pmatrix} \mathbf{A} \\ \mathbf{A}_{\text{new}} \end{pmatrix}.$$

The G=MAT model in the presence of the new motifs now takes the following

form:

$$\begin{aligned}
\hat{\mathbf{G}} &= \mathbf{M}_{\text{full}} \mathbf{A}_{\text{full}} \mathbf{T} = \\
&= (\mathbf{M} \mathbf{A} + \mathbf{M}_{\text{new}} \mathbf{A}_{\text{new}}) \mathbf{T} = \\
&= \mathbf{M} \mathbf{A} \mathbf{T} + \mathbf{M}_{\text{new}} \mathbf{A}_{\text{new}} \mathbf{T} = \mathbf{M} \mathbf{A} \mathbf{T} + \mathbf{B} \mathbf{T},
\end{aligned} \tag{5.3}$$

where \mathbf{B} is the additional matrix of parameters that needs to be estimated.

It is natural to estimate the parameters (\mathbf{A}, \mathbf{B}) of the model (5.3) using the least squares method with a penalty on \mathbf{B} .

Definition 5.1 Define the least squares fit for the parameter matrices $(\hat{\mathbf{A}}_{\text{LS}}, \hat{\mathbf{B}}_{\text{LS}})$ of the model (5.3) as follows:

$$(\hat{\mathbf{A}}_{\text{LS}}, \hat{\mathbf{B}}_{\text{LS}}) = \underset{\mathbf{A}, \mathbf{B}}{\operatorname{argmin}} \|\mathbf{G} - \mathbf{M} \mathbf{A} \mathbf{T} - \mathbf{B} \mathbf{T}\|^2 + \lambda \|\mathbf{B}\|^2, \tag{5.4}$$

where $\lambda > 0$ is the penalty coefficient.

Quite surprisingly, the solution $\hat{\mathbf{A}}_{\text{LS}}$ to (5.4) is *exactly* equal to the least squares solution of $\mathbf{G} = \mathbf{M} \mathbf{A} \mathbf{T}$ (4.4).

Theorem 5.2 The solution to the problem (5.4) can be computed as follows:

$$\hat{\mathbf{A}}_{\text{LS}} = \mathbf{M}^+ \mathbf{G} \mathbf{T}^+ + (\mathbf{M}^+ \mathbf{M} - \mathbf{I}) \mathbf{K} + \mathbf{L} (\mathbf{T} \mathbf{T}^+ - \mathbf{I}), \tag{5.5}$$

$$\hat{\mathbf{B}}_{\text{LS}} = (\mathbf{G} - \mathbf{M} \hat{\mathbf{A}}_{\text{LS}} \mathbf{T}) \mathbf{T}^T (\mathbf{T} \mathbf{T}^T + \lambda \mathbf{I})^{-1}, \tag{5.6}$$

where \mathbf{K} and \mathbf{L} are any $n_{\text{M}} \times n_{\text{T}}$ matrices.

Proof. The proof is similar to the proof of Theorem 4.1. The objective function in problem (5.4) is a convex quadratic function and to find its minimum we search for the points where the gradient is zero:

$$\frac{\partial \|\mathbf{G} - \mathbf{M} \mathbf{A} \mathbf{T} - \mathbf{B} \mathbf{T}\|^2 + \lambda \|\mathbf{B}\|^2}{\partial \mathbf{A}} = \mathbf{0}, \tag{5.7}$$

$$\frac{\partial \|\mathbf{G} - \mathbf{M} \mathbf{A} \mathbf{T} - \mathbf{B} \mathbf{T}\|^2 + \lambda \|\mathbf{B}\|^2}{\partial \mathbf{B}} = \mathbf{0}. \tag{5.8}$$

We start with equation (5.7) and solve as in Theorem 4.1:

$$\begin{aligned}
& \frac{\partial}{\partial \mathbf{A}} \sum_{i,j} (\mathbf{G}_{ij} - (\mathbf{MAT})_{ij} - (\mathbf{BT})_{ij})^2 = \mathbf{0} \\
& \sum_{i,j} 2(\mathbf{G}_{ij} - (\mathbf{MAT})_{ij} - (\mathbf{BT})_{ij}) \frac{\partial(-(\mathbf{MAT})_{ij})}{\partial \mathbf{A}} = \mathbf{0}, \\
& \sum_{i,j} -2(\mathbf{G}_{ij} - (\mathbf{MAT})_{ij} - (\mathbf{BT})_{ij}) M_{i\ell} T_{kj} = 0, \text{ for all } \ell, k, \\
& \mathbf{M}^T (\mathbf{G} - \mathbf{MAT} - \mathbf{BT}) \mathbf{T}^T = \mathbf{0}. \tag{5.9}
\end{aligned}$$

Next, we proceed with equation (5.8) similarly:

$$\begin{aligned}
& \frac{\partial}{\partial \mathbf{B}} \sum_{i,j} (\mathbf{G}_{ij} - (\mathbf{MAT})_{ij} - (\mathbf{BT})_{ij})^2 + \frac{\partial}{\partial \mathbf{B}} \lambda \sum_{i,k} B_{ik}^2 = \mathbf{0} \\
& \sum_{i,j} 2(\mathbf{G}_{ij} - (\mathbf{MAT})_{ij} - (\mathbf{BT})_{ij}) \frac{\partial(-(\mathbf{BT})_{ij})}{\partial \mathbf{B}} + 2\lambda \mathbf{B} = \mathbf{0}, \tag{5.10}
\end{aligned}$$

Consider the derivative of $-(\mathbf{BT})_{ij}$ with respect to $B_{\iota\kappa}$:

$$\frac{\partial(-(\mathbf{BT})_{ij})}{\partial B_{\iota\kappa}} = \frac{\partial(-\sum_k B_{ik} T_{kj})}{\partial B_{\iota\kappa}} = \begin{cases} -T_{\kappa j}, & \text{if } i = \iota, \\ 0, & \text{otherwise.} \end{cases} \tag{5.11}$$

Substituting it into (5.10):

$$\sum_j -2(\mathbf{G}_{\iota j} - (\mathbf{MAT})_{\iota j} - (\mathbf{BT})_{\iota j}) T_{\kappa j} + 2\lambda B_{\iota\kappa} = 0, \text{ for any } \iota, \kappa,$$

which can be rearranged to the matrix form:

$$(\mathbf{G} - \mathbf{MAT} - \mathbf{BT}) \mathbf{T}^T = \lambda \mathbf{B}. \tag{5.12}$$

Now substitute (5.12) into (5.9):

$$\mathbf{M}^T (\mathbf{G} - \mathbf{MAT} - \mathbf{BT}) \mathbf{T}^T = \lambda \mathbf{M}^T \mathbf{B} = \mathbf{0},$$

and thus, because $\lambda \neq 0$:

$$\mathbf{M}^T \mathbf{B} = \mathbf{0}.$$

As a result, we can transform equation (5.9) to the familiar form:

$$\begin{aligned}\mathbf{M}^T(\mathbf{G} - \mathbf{MAT} - \mathbf{BT})\mathbf{T}^T &= \mathbf{M}^T(\mathbf{G} - \mathbf{MAT})\mathbf{T}^T - \mathbf{M}^T\mathbf{BTT}^T \\ &= \mathbf{M}^T(\mathbf{G} - \mathbf{MAT})\mathbf{T}^T - \mathbf{0} \\ &= \mathbf{M}^T(\mathbf{G} - \mathbf{MAT})\mathbf{T}^T = \mathbf{0}.\end{aligned}$$

The solutions to this equation are exactly the solutions of (4.3) given by Theorem 4.2:

$$\hat{\mathbf{A}}_{\text{LS}} = \mathbf{M}^+\mathbf{GT}^+ + (\mathbf{M}^+\mathbf{M} - \mathbf{I})\mathbf{K} + \mathbf{L}(\mathbf{TT}^+ - \mathbf{I}). \quad (5.13)$$

Finally, substituting (5.13) into (5.9) we get:

$$\begin{aligned}(\mathbf{G} - \mathbf{M}\hat{\mathbf{A}}_{\text{LS}}\mathbf{T} - \mathbf{BT})\mathbf{T}^T &= \lambda\mathbf{B}, \\ (\mathbf{G} - \mathbf{M}\hat{\mathbf{A}}_{\text{LS}}\mathbf{T})\mathbf{T}^T - \mathbf{BTT}^T &= \lambda\mathbf{B}, \\ (\mathbf{G} - \mathbf{M}\hat{\mathbf{A}}_{\text{LS}}\mathbf{T})\mathbf{T}^T &= \mathbf{BTT}^T + \lambda\mathbf{B}, \\ (\mathbf{G} - \mathbf{M}\hat{\mathbf{A}}_{\text{LS}}\mathbf{T})\mathbf{T}^T &= \mathbf{B}(\mathbf{TT}^T + \lambda\mathbf{I}).\end{aligned}$$

The matrix $(\mathbf{TT}^T + \lambda\mathbf{I})$ is always invertible for $\lambda > 0$ (see equation (2.3) in Section 2.1), and therefore, the solution $\hat{\mathbf{B}}_{\text{LS}}$ is:

$$\hat{\mathbf{B}}_{\text{LS}} = (\mathbf{G} - \mathbf{M}\hat{\mathbf{A}}_{\text{LS}}\mathbf{T})\mathbf{T}^T(\mathbf{TT}^T + \lambda\mathbf{I})^{-1}.$$

■

The result of Theorem 5.2 means that by introducing a number of unknown (*latent*) motifs into $\mathbf{G}=\mathbf{MAT}$, we still keep the value and interpretation of the parameter matrix $\hat{\mathbf{A}}_{\text{LS}}$ at the same time elegantly getting rid of the “motif bottleneck”. The predictive error of the new model (5.3) is significantly lower. For example, on the Spellman dataset the $\mathbf{G}=\mathbf{MAT}+\mathbf{BT}$ model has a mean squared error of 0.

To summarize, we have presented both experimental and theoretical evidence for the possibility of our model to perform well for attribute selection despite the very high mean squared error.

5.5 Comparison of Methods

In this section, we use the artificial dataset to compare the attribute selection performance of the different $\mathbf{G}=\mathbf{MAT}$ estimation methods presented in Chapter 4.

5.5.1 Choice of the λ Parameter

A number of methods require setting the regularization parameter λ . To perform a fair comparison, we evaluate the performance of these methods for several values of λ on a single dataset.

Test dataset. We constructed the test dataset as follows. First we generated a noise-free dataset $(\mathbf{G}^a, \mathbf{M}^a, \mathbf{A}^a, \mathbf{T}^a)$ as described in Section 5.3. Let $\sigma^2 = D(\mathbf{G}^a)$ be the variance of \mathbf{G}^a . We added zero-mean Gaussian noise to \mathbf{G}^a with variance $4\sigma^2$. Finally, we dropped 80 rows of \mathbf{T}^a and 80 columns of \mathbf{M}^a thus leaving only 20 TFs and 20 motifs in the dataset. We believe that such a noisy setup with a significant lack of information might correspond closely to the real biological situation.

Methods. In the experiment we compared the following G=MAT methods: regularized least squares (Definition 4.2), ridge regression with $\lambda_M = \lambda_T = \lambda$ (Definition 4.3), sparse regression (Definition 4.5), p-value-based attribute selector for ridge regression (Definition 4.8), z-score-based attribute selector for ridge regression (Definition 4.9) and positivity-based attribute selector for ridge regression (Definition 4.7). For each method and for each value of the regularization parameter λ we computed the ROC AUC score as described in Section 5.4.1.

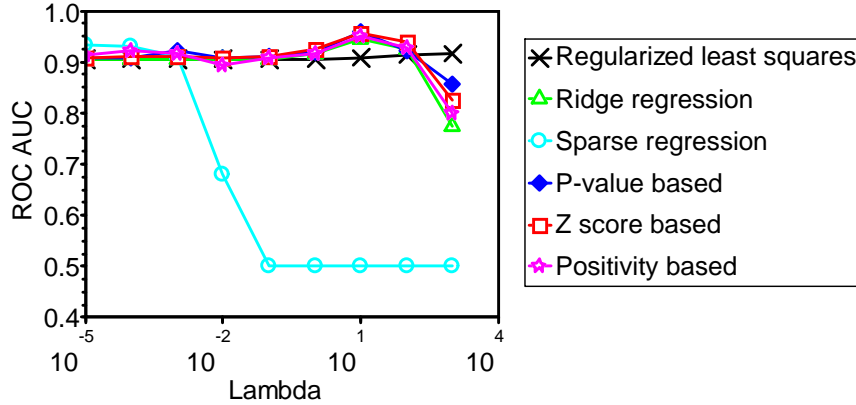


Figure 5.8: The ROC AUC score of different estimation methods for different values of λ .

Results. The results are depicted in Figure 5.8. We can clearly see that there was no significant difference in performance among all methods on this dataset. The sparse regression estimate only worked with low values of the regularization parameter, but for other methods λ could be rather freely chosen within the range

0...10 with little effect on performance: the relatively small number of model parameters prevents overfitting and hence there is no real need for regularization.

5.5.2 Comparison of Performance

In the previous section we compared the parameterized algorithms on a single dataset and discovered that most of them perform well when $\lambda = 10$. Here we fix this value of λ for all parameterized methods except sparse regression, for which we fix $\lambda = 10^{-5}$. This way we get rid of the parameterization issue, and can perform a fair comparison among all methods.

Test setup. We test each method 100 times by generating a new random dataset on every iteration. Each dataset is generated as described in the previous section. The ROC AUC scores of all runs are averaged to produce the final score.

Methods. We add the following parameter-free methods into the comparison: least squares regression (Theorem 4.3), correlation-based estimate (Theorem 4.7), and the p-value, z-score and positivity based attribute selectors for least squares.

Results. The results are presented in Figure 5.9. In general, all methods perform quite well, most of them achieving ROC AUC score higher than 0.9. The best performance (0.983) is achieved by the correlation-based estimate. Although it differs only slightly from the second best result (ridge regression z-score, 0.959), the difference is statistically significant. In 79 iterations out of 100 the correlation-based estimate was better than the one based on ridge-regression z-score.

The results also clearly demonstrate how the p-value and z-score-based randomization techniques improve the performance of the base methods.

5.5.3 The Effect of Centering

The correlation-based estimate, despite its good performance on our test dataset, can be computationally expensive. In Theorem 4.9 we have demonstrated that an approximation can be obtained by applying the least squares method to the properly centered data. Here, we demonstrate that this is indeed the case. Figure 5.10 introduces the centered least squares and centered ridge regression into comparison as well as their p-value and z-score randomizations. We see that the centered versions of the least squares and ridge regression perform even better than the correlation-based approach, thus beating all other methods. It is worth noting that the p-value and z-score randomizations could not further boost their performance.

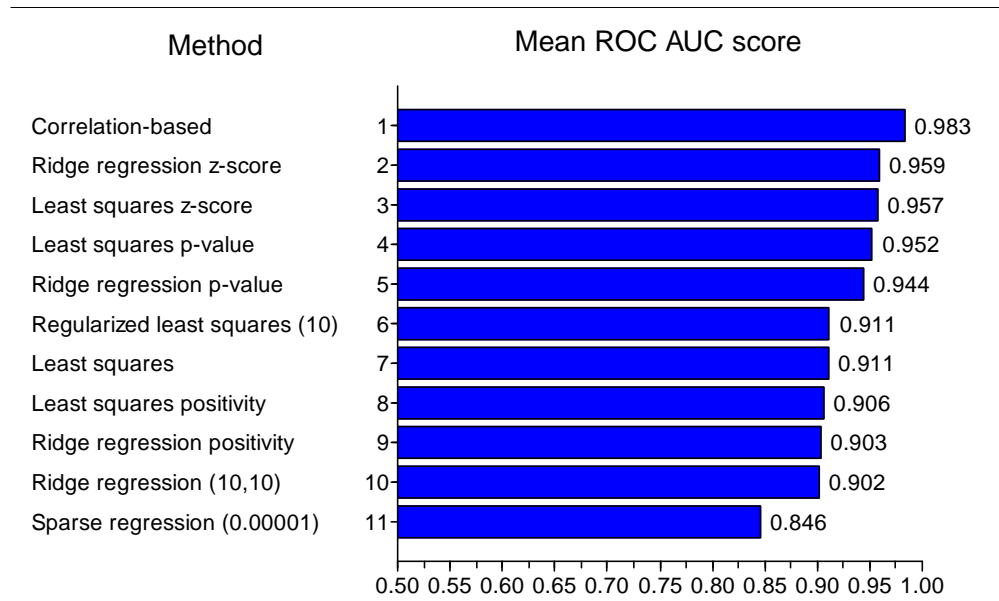


Figure 5.9: The ROC AUC score of different estimation methods, averaged over 100 runs.

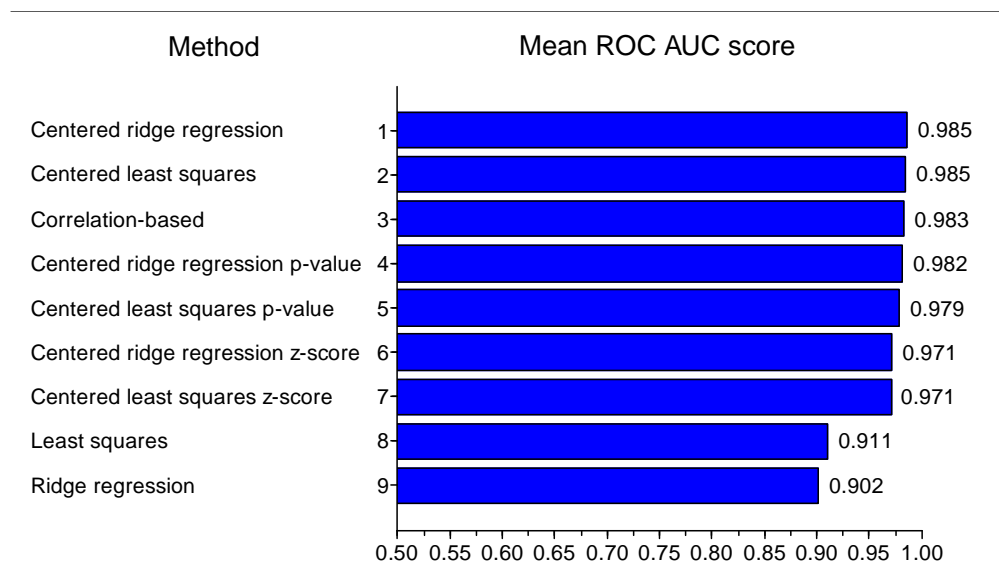


Figure 5.10: The ROC AUC score of different estimation methods, averaged over 100 runs.

Chapter 6

Applications of G=MAT

In theory, there is no difference between theory and practice.
But in practice there is.

Jan L. A. van de Snepscheut

As explained and illustrated in the previous chapters, G=MAT analysis can be used to discover putative associations between motifs and transcription factors. However, this is not the only task that can be addressed using G=MAT. In this chapter, we present a number of examples demonstrating various other applications of G=MAT analysis in practical settings. The detailed results of all the experiments are available via the supplementary web tool, described in Appendix A.

6.1 Discovering Process-specific TFs and Motifs

The most obvious application for the G=MAT model is the discovery of putative TF-motif associations from gene expression and motif presence data. An example of such analysis has already been presented in Section 5.1. However, quite often the discovered associations are rather indirect and require extensive biological knowledge to be verified. The results are easier to interpret if we consider the top-scoring TFs and the top-scoring motifs as two separate lists. These lists contain TFs and motifs that are specific to the processes measured in the microarray data.

Related work. Such an approach was taken in the work of Middendorff et al. [MKW⁺04], where the authors applied their GeneClass algorithm to yeast stress response data. The GeneClass algorithm works in the same setting as G=MAT. Namely, it is a predictive model that uses TF-motif pairs to predict expression of

target genes. Unlike G=MAT, the GeneClass algorithm is based on a much more complex model – an *alternative decision tree*.

The GeneClass algorithm is reported to predict expression values quite well, but its main use is the ranking of most influential TF-motif pairs. In their paper, the authors apply this algorithm to a yeast stress response dataset. They observe that the TFs and the motifs in the top-scoring pairs are indeed known to be related to stress response. We applied G=MAT on the same dataset and observed similar results.

Data. Unfortunately, it was not possible to obtain exactly the same data as the one that was used in the GeneClass experiments due to a minor, but unrecoverable error in the supplementary materials of the GeneClass paper. However, following the instructions provided in the paper, we reconstructed a similar dataset. The dataset consists of microarray data by Gasch et al. [GSK⁺00] and known yeast binding sites from TRANSFAC, matched on 500bp upstream sequences from SGD using the PATCH tool that comes with TRANSFAC.

Results. We applied G=MAT on the dataset and examined the top-scoring coefficients of the model. In general, the exact ranking of the coefficients varied depending on the chosen G=MAT estimation method and its parameters. Nonetheless, a certain small set of TFs and motifs consistently occupied the top-scoring positions. This is rather similar to the situation in the GeneClass paper, where the exact ranking varied depending on the scoring algorithm, yet several TFs were consistently present in the top.

Table 6.1 presents the result of centered ridge regression (with $\lambda_M = \lambda_T = 1$), applied to the dataset. The top-scoring transcription factor, *USV1* coincides with the top-scoring regulator obtained by GeneClass. The remaining regulators differ from those reported by GeneClass, yet we believe our list to make no less sense. Indeed, the discovered TFs and motifs are known to be involved in the processes related to stress response.

- The *RSF2* gene is known to be involved in glycerol-based growth and respiration [LROH05]. These processes have a clear relation to stress response, because use of glycerol is one of the reactions of yeast to hyperosmotic stress [Att97].
- The *SHP1* gene has been predicted to have a role in stress response [SSR⁺03].
- The *MSN1* gene is known to be involved in hyperosmotic stress [RRG⁺99].
- It is thought that the major function of the *MIG1* regulator is to repress the transcription of genes that are responsible for sugar utilization [Car99].

Motif	TF	Score
Y\$GAL1_15 Binding site for MIG1.	<i>USV1</i> Putative transcription factor containing a C2H2 zinc finger; mutation affects transcriptional regulation of genes involved in protein folding, ATP binding, and cell wall biosynthesis.	0.63
Y\$HSP12_01 Binding site for ABF1.	<i>USV1</i> Putative transcription factor containing a C2H2 zinc finger; mutation affects transcriptional regulation of genes involved in protein folding, ATP binding, and cell wall biosynthesis.	0.52
Y\$HSP12_01 Binding site for ABF1.	<i>RSF2</i> Zinc-finger protein involved in transcriptional control of both nuclear and mitochondrial genes, many of which specify products required for glycerol-based growth, respiration, and other functions.	0.50
Y\$CHA1_04 Binding site for ABF1.	<i>SHP1</i> UBX (ubiquitin regulatory X) domain-containing protein that regulates Glc7p phosphatase activity and interacts with Cdc48p; noone reads this, anyway. SHP1 interacts with ubiquitylated proteins in vivo and is required for degradation of a ubiquitylated model substrate.	0.50
Y\$GAL1_15 Binding site for MIG1.	<i>MSN1</i> Transcriptional activator involved in regulation of invertase and glucoamylase expression, invasive growth and pseudohyphal differentiation, iron uptake, chromium accumulation, and response to osmotic stress; localizes to the nucleus.	0.48

Table 6.1: G=MAT analysis of the Gasch dataset. The table presents five motif-TF pairs having the largest values of the corresponding parameters $\hat{\alpha}_{\ell k}$. The parameter values are given in the rightmost column.

- The gene *ABF1* encodes a multifunctional regulator particularly involved in different chromatin-related events [RSOC89]. The highly-scoring binding site Y\$HSP12_01 of this protein was originally discovered in the promoter of the *HSP12* heat shock gene [PM90].

Other G=MAT estimates produced different, but still meaningful results. For instance, the heat shock factor *HSF1* occupies several top-scoring positions in the G=MAT correlation-based results. As several of the microarray experiments were measuring the response of yeast to heat shock, this result makes sense.

6.2 Motif Discovery

In the examples considered so far, we used a rather small set of well-known motifs and aimed at identifying the most influential out of these. Alternatively, we can use a large set of motifs encompassing all the substrings of a given length. Finding the most influential out of that set is equivalent to identifying biologically meaningful sequences in DNA – a task known as *motif discovery*. This task is one of the key subjects in bioinformatics. A good overview of motif discovery methods and applications is provided in the master’s thesis of M. Haeussler [Hae05].

Related work. An approach similar to G=MAT has already been used for motif discovery in the work of Bussemaker et al. [BLS01], where the authors applied their REDUCE algorithm for yeast promoter sequences. In brief, the idea of the REDUCE algorithm is to use the $\hat{\mathbf{G}} = \mathbf{MC}$ model (described in Section 5.4.4) to score motifs and select the highest scoring ones as biologically significant. In their paper, the authors applied this idea to microarray data by Spellman et al. [SSZ⁺98]. Their approach was to iteratively construct a set of 7-nucleotide motifs that correlate most with the gene expression values. Conceptually, this is quite similar to what is done using G=MAT.

Data. We considered all possible 7-mers of letters $\{\mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{G}\}$ and matched them on the promoters (800bp upstream sequences) of the 5766 genes of the Spellman dataset. The resulting motif matrix contained $4^7 = 16384$ motifs, which was significantly larger than the number of genes $n_G = 5766$ and could lead to overfitting. To reduce the number of motifs, we selected about 4000 of the 7-mers that were present in at least 351 promoters. The microarray dataset that we used is the one described in Section 5.1.

Results. The top scoring motif of the least squares estimate was **AAATCTT**. This does not differ much from the two top-scoring results of the REDUCE algorithm: **AAAATTT** and **AAATTTT**. Also interesting was the top-scoring motif of the G=MAT correlation-based estimate, **CGATGAG**. This motif is the fourth highest on the REDUCE result list. Notably, both motifs have also been discovered from the same data by Vilo et al. in [VBJ⁺00]. The other high-scoring motifs were different from the REDUCE results, and might even contain novel discoveries.

6.3 Automatic GO Annotation

Automated assignment of relevant *Gene Ontology* (GO) annotations to genes is an important problem and a popular research direction in contemporary com-

Motif	TF	Score
GO:0000747 Conjugation with cellular fusion	<i>KAR4</i> Transcription factor required for gene regulation in response to pheromones.	0.07
GO:0043332 Mating projection tip	<i>KAR4</i> Transcription factor required for gene regulation in response to pheromones.	0.06
GO:0005762 Mitochondrial large ribosomal subunit	<i>RGM1</i> Putative transcriptional repressor with proline-rich zinc fingers.	0.05
GO:0006999 Nuclear pore organization and biogenesis	<i>CRF1</i> Transcriptional corepressor involved in the regulation of ribosomal protein gene transcription via the TOR signaling pathway and protein kinase A, phosphorylated by activated Yak1p which promotes accumulation of Crf1p in the nucleus.	0.05
GO:0005763 Mitochondrial small ribosomal subunit	<i>RGM1</i> Putative transcriptional repressor with proline-rich zinc fingers.	0.05

Table 6.2: G=MAT for GO annotation on the Spellman dataset. The table presents five (GO term, TF) pairs having the largest values of the corresponding parameters $\hat{\alpha}_{\ell k}$.

putational biology [RKP⁺07]. In this section, we demonstrate how G=MAT can be employed for this purpose.

In all our previous experiments, the values of model parameters could be interpreted as follows: a high $\hat{\alpha}_{\ell k}$ indicates that the expression of transcription factor t_k correlates well with the expression of genes that have motif m_ℓ in their promoter. In this experiment, we propose to replace motifs with GO terms, and the motif matrix \mathbf{M} with the binary matrix of GO annotations. Formally, let $\{m_1, m_2, \dots, m_{n_M}\}$ be a set of GO terms, and let

$$M_{i\ell} = \begin{cases} 1, & \text{if the gene } g_i \text{ is annotated with the term } m_\ell, \\ 0, & \text{otherwise.} \end{cases}$$

In this case, the interpretation of model parameters changes to the following: a high $\hat{\alpha}_{\ell k}$ indicates that the expression of transcription factor t_k correlates well with the expression of genes that are annotated with the GO term m_ℓ . Therefore, a

high value of $\hat{\alpha}_{\ell k}$ suggests that the TF t_k is also somehow related to the term m_ℓ . This allows to use G=MAT for discovering putative GO annotations. We illustrate the idea with an experiment.

Data. We used the Spellman dataset, described in Section 5.1, for the **G** and **T** matrices. To construct the matrix **M**, we selected 200 GO terms that had the greatest number of genes associated with them and created a 5766×200 binary matrix of annotations as described above.

Results. Ridge regression with $\lambda_M = \lambda_T = 1$, produced quite interesting results on this dataset. Out of the ten top-scoring pairs of TFs and GO terms, one corresponded to a known GO annotation. This is statistically significant. Indeed, if we picked ten pairs randomly, the expected number of guesses would only be 0.10. Moreover, the ten pairs with the lowest scores contained two known annotations, which is even more significant. Finally, consider the five top-scoring pairs presented in Table 6.2. The discovered pairs are, at the very least, quite consistent.

- The *KAR4* gene is associated to the terms “*conjugation with cellular fusion*” and “*mating projection tip*”. Both terms are related to the mating process, and the *KAR4* gene is actually known to be involved in this process. In fact, its current true annotation is “*karyogamy during conjugation with cellular fusion*”.

Also, note that we can regard the obtained result as two separate lists, as we did it in Section 6.1. In this case, the list of top-scoring GO terms represents the important processes that were measured in the expression data.

6.4 A Non-biological Application

The G=MAT model can also be useful outside of biological applications. For instance, consider the following generalization of the G=MAT context. Let there be a set of *subjects* $\{g_1, g_2, \dots, g_{n_G}\}$, a set of *objects* $\{a_1, a_2, \dots, a_{n_A}\}$ and suppose that each subject g_i has provided a *score* G_{ij} to each object a_j . Next, suppose that subjects have attributes $\{m_1, m_2, \dots, m_{n_M}\}$ and for each subject g_i the value of the attribute m_ℓ is given by $M_{i\ell}$. Finally, suppose that objects have attributes $\{t_1, t_2, \dots, t_{n_T}\}$ and for each object a_j the value of the attribute t_k is given by T_{kj} . In the previous examples, the subjects were genes, objects were experiments, subject attributes were motifs or GO annotations, and object attributes were transcription factors. We have shown that the G=MAT model makes sense for such a biological setup. However, other contexts exist, where the scores G_{ij} can be meaningfully modeled with G=MAT.

Consider the following example. Let subjects be people, objects be movies and let each person provide each movie with a rating that represents his opinion about the quality of that movie. As a simplification, we can assume that people judge movies by assessing certain features separately and adding up the results. For example, if John loves dramas with happy ending, he will give 1 point to any movie that is a drama, and an additional 1 point to any movie that has a happy ending. Similarly, if Jane strongly dislikes horror movies, she will reduce the score of any movie by 2 points if it has elements of horror. Let us select movie attributes “*drama*”, “*horror*” and “*happy ending*” and person attributes “*likes drama*”, “*likes horror*” and “*likes happy ending*”. It is possible to see that with such attributes, and in the simplified context described above, movie ratings can indeed be modeled using G=MAT.

Experimental Setup. We have performed a simplistic experiment illustrating this idea. We created an online questionnaire, in which visitors were asked to estimate their current mood (whether they feel elevated, calm or depressed) and select from a given set of popular movies those that would suit their current mood best, i.e., they would agree to watch those movies now. About 100 Tartu University students have responded to our call and filled in the questionnaire.

Data. We constructed a G=MAT dataset ($\mathbf{G}, \mathbf{M}, \mathbf{T}$) as follows. The subjects g_i were the people that took part in the survey, the objects a_j were the movies that the people could chose from, and the matrix \mathbf{G} was constructed as a binary matrix where

$$G_{ij} = \begin{cases} 1, & \text{if the person } g_i \text{ chose the movie } a_j \text{ as fitting its current mood,} \\ 0, & \text{otherwise.} \end{cases}$$

The set of movie attributes t_k was chosen as the set of genres

$$\{\text{Cartoon, Thriller, Drama, Comedy, Sci-Fi}\},$$

and the matrix \mathbf{T} was a binary matrix where

$$T_{kj} = \begin{cases} 1, & \text{if the movie } a_j \text{ had elements of genre } t_k \text{ in it,} \\ 0, & \text{otherwise.} \end{cases}$$

Finally, the set of person attributes m_ℓ was the set of mood types

$$\{\text{Happy, Calm, Sad}\},$$

and the matrix \mathbf{M} was a binary matrix where

$$M_{i\ell} = \begin{cases} 1, & \text{if the person } g_i \text{ reported her current mood as } m_\ell, \\ 0, & \text{otherwise.} \end{cases}$$

Results. The result of a G=MAT analysis of this dataset provides a ranked list of associations between person attributes (mood) and movie attributes (genre). Before performing the experiment, we expected to see positive associations between *happy* mood and *comedy* movies, also *sad* mood and *drama* movies. And indeed, the pair relating *happy* mood to *comedy* movies was the highest scoring pair according most estimates, scoring significantly higher than all the other associations. However, our second guess was not reflected in the results: the association of *drama* to *sadness* did not get a high score.

Summary

A conclusion is the place where you got tired of thinking.

The development and functions of every part of our body, every single cell, are encoded in the DNA: a molecule present in nearly all our cells. A multitude of cell types exist, such as skin cells, blood cells, neurons, etc., yet all of these cells contain *exactly the same* copy of the DNA. The type of a cell is therefore not only determined by the DNA sequence, but rather by the complex processes of *gene regulation* constantly taking place inside each cell. In particular, there exist certain *transcription factor* proteins, that can stick to certain sites on the DNA and thus switch the corresponding DNA regions “on” or “off”. The question of which transcription factors bind to which DNA sites and when they do it, is of great interest for contemporary biology. For instance, it might show ways of converting skin cells into brain, bone or blood cells. This would have significant implications for medicine. Unfortunately, it is rather difficult and expensive to figure out the answer to this question using methods of experimental biology.

We proposed a statistical model to address this problem. Using the widely and freely available microarray data together with the DNA sequences, our method can propose potential DNA-binding candidates together with the places where they could bind. The main idea is to use a linear model that predicts gene expression values G_{ij} as a weighted sum of products of expression values of transcription factors T_{kj} and motif counts $M_{i\ell}$:

$$G_{ij} = \sum_{\ell=1}^{n_M} \sum_{k=1}^{n_T} \alpha_{\ell k} M_{i\ell} T_{kj}.$$

Each parameter $\alpha_{\ell k}$ of this model represents the association strength between some motif m_ℓ and some transcription factor t_k . Given a dataset of expression measurements and motif counts, represented in the form of three matrices $(\mathbf{G}, \mathbf{M}, \mathbf{T})$, it is possible to estimate the values of the model parameters $(\hat{\alpha}_{\ell k})$. The estimated parameters with the largest values correspond to putative associations between motifs and transcription factors.

In this work, we discussed and illustrated several methods for computing parameter estimates, such as least squares regression, regularized least squares regression, ridge regression and a correlation-based method.

We studied the applicability of the model to biological data. Experiments on both real and artificial data demonstrated that our model is not predictive, but purely descriptive. That is, the prediction error of the model is very large, but the estimated parameters are still reliable and biologically meaningful. For instance, we have shown that associations discovered using our model from the well-known Spellman microarray dataset correspond to known indirect relations between transcription factors and motifs. Additionally, we illustrated how the G=MAT model can be applied in several other contexts besides the discovery of TF-motif associations. We demonstrated how G=MAT can be applied for the discovery of process-specific TFs and motifs, for motif discovery and for GO annotation. Finally, we presented an example of a completely non-biological application in the context of movie preferences.

G=MAT: meetod geeniekspressiooni ja DNA seondumisandmete ühendamiseks

Magistritöö (40ap)

Konstantin Tretjakov

Resümee

DNA on molekul, mis kannab pärilikku informatsiooni ning määrab suures osas terve meie keha ehituse ja arengu. Meie keha koosneb erinevatest rakutüüpidest, kuid kõikides rakkudes on sama DNA. Iga raku omadused on peamiselt määratud selle poolt, millised DNA alad (geenid) on antud rakus aktiivsed ja millised mitte. Erilist rolli geenide aktiivsuse määramisel mängivad transkriptsioonifaktorid. Need on valgud, mis on võimelised seonduma teatud kohtadega (saitidega) DNA ahelal ning selle kaudu lülitada “sisse” või “välja” vastavaid DNA alasid (geene). Üks oluline bioloogiline küsimus seisneb selles, millised transkriptsioonifaktorid seonduvad milliste saitidega. Sellele küsimusele vastuse leidmiseks veel lihtsat ja odavat tehnoloogiat ei eksisteeri.

Antud töös kirjeldatakse statistilist mudelit, mille abil on võimalik otsida potentsiaalseid seoseid transkriptsioonifaktorite ja seondumissaitide vahel kasutades olemasolevaid mikrokiibi ning DNA järjestuse andmeid. Meetodi põhiidee on kasutada lineaarset mudelit, mis ennustab geenide ekspressiooni transkriptsioonifaktorite ekspressiooni ning seondumissaitide olemasolu põhjal. Selle mudeli parameetreid võib interpreteerida kui faktorite ja saitide vaheliste seoste tugevust.

Töös on välja toodud mitu algoritmi mudeli parameetrite hindamiseks ning on põhjalikult analüüsitud mudeli sobivust bioloogilisteks rakendusteks. Kuigi mudel ei ennusta hästi, on tema abil hinnatud parameetrite väärtused siiski usaldusväärsed ning seega on mudel kasutatav kirjeldava analüüsi jaoks. Töös demonstreeritakse mitut erinevat mudeli rakenduslikku näidet nii bioloogiliste kui ka mittebioloogiliste andmete peal.

References

- [ABB⁺00] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat Genet*, 25(1):25–29, May 2000.
- [Att97] P. V. Attfield. Stress tolerance: the key to effective strains of industrial baker’s yeast. *Nat Biotechnol*, 15(13):1351–1357, Dec 1997.
- [BH92] P. J. Bhat and J. E. Hopper. Overproduction of the GAL1 or GAL3 protein causes galactose-independent activation of the GAL4 protein: evidence for a new model of induction for the yeast GAL/MEL regulon. *Mol Cell Biol*, 12(6):2701–2707, Jun 1992.
- [BLS01] H. J. Bussemaker, H. Li, and E. D. Siggia. Regulatory element detection using correlation with expression. *Nat Genet*, 27(2):167–171, Feb 2001.
- [BOH90] P. J. Bhat, D. Oh, and J. E. Hopper. Analysis of the GAL3 signal transduction pathway activating GAL4 protein-dependent transcription in *Saccharomyces cerevisiae*. *Genetics*, 125(2):281–291, Jun 1990.
- [BVT⁺08] Jan Christian Bryne, Eivind Valen, Man-Hung Eric Tang, Troels Marstrand, Ole Winther, Isabelle da Piedade, Anders Krogh, Boris Lenhard, and Albin Sandelin. JASPAR, the open access database of transcription factor-binding profiles: new content and tools in the 2008 update. *Nucleic Acids Res*, 36(Database issue):D102–D106, Jan 2008.
- [Car99] M. Carlson. Glucose repression in yeast. *Curr Opin Microbiol*, 2(2):202–207, Apr 1999.
- [Cha82] Tony F. Chan. An improved algorithm for computing the singular value decomposition. *ACM Trans. Math. Softw.*, 8(1):72–83, 1982.

- [Che] CherryPy HTTP framework – official website.
<http://www.cherrypy.org/> (accessed on June 1, 2008).
- [DDDM04] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communciation Pure Application*, LVII:1413–1457, 2004.
- [EHJT04] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Ann. Statist.*, 32(2):407–499, 2004.
- [GMA] G=MAT supplementary website.
<http://biit.cs.ut.ee/gmat> (as of June 1, 2008).
- [GS97] Charles M. Grinstead and J. Laurie Snell. *Introduction to Probability*. AMS, 1997.
- [GSK⁺00] A. P. Gasch, P. T. Spellman, C. M. Kao, O. Carmel-Harel, M. B. Eisen, G. Storz, D. Botstein, and P. O. Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Mol Biol Cell*, 11(12):4241–4257, Dec 2000.
- [Hae05] Maximilian Haeussler. Motif discovery on promotor sequences. Master’s thesis, University of Potsdam, 2005.
- [HS02] Christine E Horak and Michael Snyder. ChIP-chip: a genomic approach for identifying transcription factor binding sites. *Methods Enzymol*, 350:469–483, 2002.
- [HS03] Wolfgang Härdle and Leopold Simar. *Applied Multivariate Statistical Analysis*. Springer-Verlag, 2003. 486 pages.
- [Kid] Kid XML-based templating language – official website.
<http://www.kid-templating.org/> (accessed on June 1, 2008).
- [LROH05] Lin Lu, George G Roberts, Cynthia Oszust, and Alan P Hudson. The YJR127C/ZMS1 gene product is involved in glycerol-based respiratory growth of the yeast *Saccharomyces cerevisiae*. *Curr Genet*, 48(4):235–246, Oct 2005.
- [LVZ95] D. Lohr, P. Venkov, and J. Zlatanova. Transcriptional regulation in the yeast GAL gene family: a complex genetic network. *FASEB J*, 9(9):777–787, Jun 1995.
- [Mey01] Carl Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, 2001.

- [MFG⁺03] V. Matys, E. Fricke, R. Geffers, E. Gössling, M. Haubrock, R. Hehl, K. Hornischer, D. Karas, A. E. Kel, O. V. Kel-Margoulis, D-U. Kloos, S. Land, B. Lewicki-Potapov, H. Michael, R. Münch, I. Reuter, S. Rotert, H. Saxel, M. Scheer, S. Thiele, and E. Wingender. TRANSFAC: transcriptional regulation, from patterns to profiles. *Nucleic Acids Res*, 31(1):374–378, Jan 2003.
- [MKW⁺04] Manuel Middendorf, Anshul Kundaje, Chris Wiggins, Yoav Freund, and Christina Leslie. Predicting genetic regulatory response using classification. *Bioinformatics*, 20 Suppl 1:I232–I240, Aug 2004.
- [MS06] Keith E. Muller and Paul W. Stewart. *Linear Model Theory: Univariate, Multivariate and Mixed Models*. Wiley Series in Probability and Statistics. Wiley-Interscience, 2006.
- [Num] NumPy numerical routines library – official website.
<http://numpy.scipy.org/> (accessed on June 1, 2008).
- [PM90] U. M. Praekelt and P. A. Meacock. HSP12, a new small heat shock gene of *Saccharomyces cerevisiae*: analysis of structure, regulation and function. *Mol Gen Genet*, 223(1):97–106, Aug 1990.
- [PRHR00] A. Platt, H. C. Ross, S. Hankin, and R. J. Reece. The insertion of two amino acids into a transcriptional inducer converts it into a galactokinase. *Proc Natl Acad Sci U S A*, 97(7):3154–3159, Mar 2000.
- [PSC01] Y. Pilpel, P. Sudarsanam, and G. M. Church. Identifying regulatory networks by combinatorial analysis of promoter elements. *Nat Genet*, 29(2):153–9, Oct 2001.
- [Pyt] Python programming language – official website.
<http://www.python.org/> (accessed on June 1, 2008).
- [RKP⁺07] Jüri Reimand, Meelis Kull, Hedi Peterson, Jaanus Hansen, and Jaak Vilo. g:Profiler—a web-based toolset for functional profiling of gene lists from large-scale experiments. *Nucleic Acids Res*, 35(Web Server issue):W193–W200, Jul 2007.
- [RRG⁺99] M. Rep, V. Reiser, U. Gartner, J. M. Thevelein, S. Hohmann, G. Ammerer, and H. Ruis. Osmotic stress-induced gene expression in *Saccharomyces cerevisiae* requires Msn1p and the novel nuclear factor Hot1p. *Mol Cell Biol*, 19(8):5474–5485, Aug 1999.

- [RSOC89] P. R. Rhode, K. S. Sweder, K. F. Oegema, and J. L. Campbell. The gene encoding ARS-binding factor I is essential for the viability of yeast. *Genes Dev*, 3(12A):1926–1939, Dec 1989.
- [RZ06] Jianhua Ruan and Weixiong Zhang. A bi-dimensional regression tree approach to the modeling of gene expression regulation. *Bioinformatics*, 22(3):332–340, Feb 2006.
- [SGD] SGD Project. “Saccharomyces Genome Database”.
<http://www.yeastgenome.org/> (accessed on July 1, 2007).
- [SKB03] Lev A Soinov, Maria A Krestyaninova, and Alvis Brazma. Towards reconstruction of gene networks from expression data by supervised learning. *Genome Biol*, 4(1):R6, 2003.
- [Soi03] L. A. Soinov. Supervised classification for gene network reconstruction. *Biochem Soc Trans*, 31(Pt 6):1497–502, Dec 2003.
- [SQLa] SQLite library – official website.
<http://www.sqlite.org/> (accessed on June 1, 2008).
- [SQLb] SQLAlchemy library – official website.
<http://www.sqlalchemy.org/> (accessed on June 1, 2008).
- [SSR⁺03] Eran Segal, Michael Shapira, Aviv Regev, Dana Pe’er, David Botstein, Daphne Koller, and Nir Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat Genet*, 34(2):166–176, Jun 2003.
- [SSZ⁺98] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol Biol Cell*, 9(12):3273–3297, Dec 1998.
- [SSZ07] Dustin E Schones, Andrew D Smith, and Michael Q Zhang. Statistical significance of cis-regulatory modules. *BMC Bioinformatics*, 8:19, 2007.
- [TCS⁺01] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17(6):520–525, Jun 2001.

- [TRR02] David J Timson, Helen C Ross, and Richard J Reece. Gal3p and Gal1p interact with the transcriptional repressor Gal80p to form a complex of 1:1 stoichiometry. *Biochem J*, 363(Pt 3):515–520, May 2002.
- [Tur] TurboGears framework – official website.
<http://www.turbogears.org/> (accessed on June 1, 2008).
- [VBJ⁺00] J. Vilo, A. Brazma, I. Jonassen, A. Robinson, and E. Ukkonen. Mining for putative regulatory elements in the yeast genome using gene expression data. *Proc Int Conf Intell Syst Mol Biol*, 8:384–394, 2000.
- [WDKK96] E. Wingender, P. Dietze, H. Karas, and R. Knüppel. TRANSFAC: a database on transcription factors and their dna binding sites. *Nucleic Acids Res*, 24(1):238–241, Jan 1996.


Appendices

Appendix A. The G=MAT Web Tool

Appendix B. Program code (on a compact disc)

Appendix A

The G=MAT Web Tool



[Login](#)

View ls result for spellman-storm-800up

Top 10 associations

Motif	TF	Score
F\$GAL4_o1 GAL4	GAL1 Galactokinase, phosphorylates alpha-D-galactose to alpha-D-galactose-1-phosphate in the first step of galactose catabolism	0.29869
F\$GAL4_o1 GAL4	GAL3 Transcriptional regulator involved in activation of the GAL genes in response to galactose	0.26362
F\$GAL4_o1 GAL4	GAL80 Transcriptional regulator involved in the repression of GAL genes in the absence of galactose	0.17578
F\$MCM1_o2 MCM1+SFF	SFG1 Nuclear protein, putative transcription factor required for growth of superficial pseudohyphae (which do not invade the agar substrate) but not for invasive pseudohyphal growth	0.12133
F\$MCM1_o2 MCM1+SFF	ACE2 Transcription factor that activates expression of early G1-specific genes, localizes to daughter cell nuclei after cytokinesis and delays G1 progression in daughters, localization is regulated by phosphorylation	0.12132

Figure 6.1: GMAT-Web. Results page (fragment).

To conveniently perform G=MAT analyses and store the obtained results, we have created a web-based tool, GMAT-Web. In this appendix, we provide a brief overview of this tool.

The purpose of GMAT-Web is to make experiments, such as those described in the main text of the thesis, convenient to perform and manage. The tool supports several features that achieve this goal.

Multiple datasets. GMAT-Web can manage several G=MAT datasets. Each dataset is represented as a directory containing several files. The dataset description is stored in a text file `meta.txt`. The names and meta information about genes, microarrays, motifs and transcription factors are stored in tab-separated plaintext files `genes.meta`, `arrays.meta`, `motifs.meta` and `tfs.meta` respectively. The matrices **G**, **M** and **T** are stored in the tab-separated plaintext files

`g.tab`, `m.tab` and `t.tab` respectively. Finally, a binary validation matrix \mathbf{A} can be assigned to the dataset for result validation by storing it in the tab-separated plaintext file `a.validate.tab`. Figure 6.2 presents a screenshot of the page that shows a list of all datasets in the system. A new dataset can be added by uploading a properly formed directory to the server. The current version of the tool does not allow users to add datasets via the web interface.


Multiple analysis methods. Each dataset can be analyzed using the following G=MAT parameter estimation methods: least squares, ridge regression, z-score-based and positivity-based attribute selectors for ridge-regression, correlation-based estimate. We did not include the regularized least squares and the sparse regression because our iterative implementations of these methods are inefficient for online use. Figure 6.3 demonstrates the selection of analysis methods as they are presented to the user via the web interface.

Asynchronous computation and caching. The computation of G=MAT parameters can take considerable time if the dataset is large or when many iterations for the randomization algorithms are requested. To be able to run such computations from within a web interface we make them asynchronous. That is, when the user chooses to apply one of the analysis methods to a dataset, the computation is started in a separate process. The user can then observe the progress of this process via the web interface, yet she is not required to stay on the page until the results are ready. Once the result is ready it will be cached and the next time the user requests the same result it will not be recomputed again but retrieved from the cache immediately. Figure 6.4 demonstrates the progress screen of a running job.

Results display. Once the parameter matrix $\hat{\mathbf{A}}$ is estimated, GMAT-Web presents the 10 top-scoring and 10 lowest-scoring coefficients as pairs of motifs and transcription factors. If the dataset contains a validation matrix \mathbf{A} , the true positive rows in the result list are highlighted and some validation metrics are computed, such as the ROC AUC score of the fit. Figure 6.1 shows a part of the results screen.

Availability. The tool is available on all platforms, where Python 2.5 and its numerical extensions library NumPy can be run. This includes, in particular, Windows, MacOS and Linux on the PC-compatible or PowerPC processors. Although it is a web solution, well-suitable to be run on a server, it can be as easily started on any client machine. The current version of the software is available on the CD attached to the hardcopy as well as on the supplementary website [GMA]. It is free for non-commercial use.

Implementation details. The tool was developed using Python 2.5 [Pyt]. The web interface was constructed using the TurboGears framework [Tur] that includes the CherryPy standalone webserver [Che], SQLite3 embedded database engine [SQLa], SQLAlchemy object-relational mapping library [SQLb] and the Kid templating language [Kid]. The methods of linear algebra, required for the parameter estimation algorithms were available in the NumPy [Num] package.



G = MAT

[Login](#)

Select dataset


- gasch-geneclass:** A dataset similar to the one used in the GeneClass paper
- gasch-geneclass-alltfs:** The GeneClass dataset with the full set of regulators.
- sample-movies:** Mood-based movie preferences
- sample-movies-full:** Mood-based movie preferences (full dataset)
- spellman-7mers3995:** Spellman Expression Data + 3995 frequent 7mers
- spellman-go-200:** Spellman Expression Data + GO annotations
- spellman-storm-800up:** Spellman Expression Data + Transfac Yeast Motifs
- spellman-tamo-800up:** Spellman Expression Data + Harbison/MacIsaac Yeast Motifs

Copyright © 2008 Konstantin Trifunov

Figure 6.2: GMAT-Web. List of datasets.

Analyze:				
G-MAT _{Corr} [?]	<input type="button" value="Go!"/>			
G-MAT _{LS} [?]	<input type="button" value="Go!"/>	options		<input checked="" type="checkbox"/> center <input checked="" type="checkbox"/> normalize
G-MAT _{RR} [?]	<input type="button" value="Go!"/>	options		<input type="text" value="1.0"/> <small>Lambda M</small> <input type="text" value="1.0"/> <small>Lambda T</small> <input checked="" type="checkbox"/> center <input checked="" type="checkbox"/> normalize
G-MAT _{RR+Zscore} [?]	<input type="button" value="Go!"/>	options		<input type="text" value="1.0"/> <small>Lambda M</small> <input type="text" value="1.0"/> <small>Lambda T</small> <input checked="" type="checkbox"/> center <input checked="" type="checkbox"/> normalize <input type="text" value="100"/> <small>Num. iterations</small>
G-MAT _{RR+P(V > o)} [?]	<input type="button" value="Go!"/>	options		<input type="text" value="1.0"/> <small>Lambda M</small> <input type="text" value="1.0"/> <small>Lambda T</small> <input checked="" type="checkbox"/> center <input checked="" type="checkbox"/> normalize <input type="text" value="100"/> <small>Num. iterations</small> <input type="text" value="80"/> <small>Holdout (training) set size in %</small>

Figure 6.3: GMAT-Web. Different analysis methods.




G = MAT

[Login](#)
 1 job running

Please wait...

The computation is in progress


42% complete

Copyright © 2008 Konstantin Trifunov

Figure 6.4: GMAT-Web. Job progress screen.