Imperial College London

Final Report for MEng Individual Project

Bayesian Identification of Genetic Regulatory Networks

Author name:

Bence Mark Halpern

Supervisor:

Dr. Guy-Bart Stan

Submitted in partial fulfilment of the requirements for the award of MEng in Biomedical Engineering from Imperial College London

June 2018 Word Count: 5915

Feedback box for project markers:

WHAT I liked about the report:
WHAT could/should be improved:

Acknowledgements

The author would like to thank Dr. Zoltan Tuza for his guidance during the writing of the thesis, Dr. Wei Pan for the useful discussions, and finally the supervisor of the project, Dr. Guy-Bart Stan.

Notations and shorthands

The following notation and shorthands will be used throughout the thesis,

- k - the total number of samples
- b - the number of dictionary functions
- $f_a(\cdot) : \mathbb{R} \to \mathbb{R},\ a \in [1, 2, \dots, b]$ - dictionary functions used in our model
- $\mathcal{G}_a(\cdot) : \mathbb{R}^{m\times 1} \to \mathbb{R},\ a \in [1, 2, \dots, b]$ - the true dictionary functions
- $\theta_a \in \mathbb{R},\ a \in [1, 2, \dots, b]$ - the true mixing coefficients of the true dictionary functions
- $\mathcal{X} \in \mathbb{R}^{m\times k}$ - matrix containing all sampled concentrations of all genes
- l - the length of dictionary matrix
- n - the Hill coefficient
- m - number of genes in gene regulatory network
- X_i - gene or molecule
- x_i - concentration of X_i
- K_M - Michaelis-Menten constant
- k_i - maximum transcription rate of gene i
- d_i - degradation rate of gene i
- $\mathbf{x} \in \mathbb{R}^{m\times k}$ - all k samples of m genes
- $x_i(t)$ - concentration function of time for gene i
- $\mathbf{x}(t_i) \in \mathbb{R}^{m\times 1}$ - i^{th} samples of all m genes
- $\mathbf{\Phi} \in \mathbb{R}^{k\times l}$ - dictionary matrix of nonlinear descriptors
- $\mathbf{y} \in \mathbb{R}^{k\times 1}$ - a vector of measurements
- $\mathbf{w} \in \mathbb{R}^{l\times 1}$ - weight or mixing coefficient of nonlinear descriptors, w_i is the i^{th} row
- $\gamma \in \mathbb{R}^{l\times 1}$ - a vector containining all prior variances
- $\mathbf{\Gamma} \in \mathbb{R}^{l\times l}$ - the diagonal matrix from γ
- λ - the shrinkage parameter of a Regularised Least Squares algorithm
- GRN - Gene Regulatory Network

Probability theory

- $\hat{x}$ - estimate of a random variable x
- $\mathcal{M}$ - the identification model of a GRN
- $\mathcal{M}^*$ - the generative model of a GRN
- $p(\mathbf{x})$ - the probability of x

- $p(\mathbf{x}|\mathbf{y})$ - the probability of x given y
- $p(\mathbf{x}, \mathbf{y})$ - the joint probability of x and y
- $p(\mathbf{x}; \mathbf{y})$ - the probability of x for a fixed y
- Σ - covariance matrix (i.e. the covariance matrix for w is Σ_w)
- ϵ - random variable denoting an additive white Gaussian noise
- $\mathcal{N}(\mu, \Sigma)$ - normal distribution with mean μ and covariance Σ
- $\mathbb{E}[\mathbf{y}]$ - the expectation of a random variable $\mathbf{y}$
- MAP - Maximum *a Posteriori*
- ML - Maximum Likelihood

Optimisation

- $\arg\min_x \mathcal{L}(x)$ - the argument that minimises the loss function function
- $\min \mathcal{L}(\cdot)$ - the minimum value of loss function
- $\mathbf{z} \in \mathbb{R}^{l\times 1}$ - the penalty reweighting coefficient, measuring the sensitivity of the estimate on reweighting the parameters
- $\mathbf{z}^* \in \mathbb{R}^{l\times 1}$ - local minimiser of $\mathbf{z}$
- $\gamma^* \in \mathbb{R}^{1\times l}$ - local minimiser of γ
- $g(\mathbf{w}) : \mathbb{R}^{l\times 1} \to \mathbb{R}$ - part of the loss function depending on $\mathbf{w}$
- $|\cdot|$ - the determinant
- $||\cdot||_p$ - the L_p norm of a matrix where $p \in [0, \infty)$
- $\zeta(\cdot) : \mathbb{R} \to \mathbb{R}$ - a non-negative penalty function
- $s(\cdot) : \mathbb{R} \to \mathbb{R}$ - $-2\ln(\cdot)$ mapping of $\zeta(\cdot)$
- $\mathrm{tr}[A]$ - trace of a matrix
- $\mathrm{diag}[A]$ - diagonal matrix of A
- $h(\cdot)$ - concave conjugate
- SBL - Sparse Bayesian Learning

Contents

1 Abstract

Data scarcity hinders state of the art system identification methods, which is often tackled by regularisation. Sparse Bayesian Learning was developed in the field of signal processing to overcome data scarcity and provide sparse representation of signals. The mathematical formulation of SBL also allows us to identify dynamical systems such as gene regulatory networks. In this thesis, the mathematical background of SBL is summarised in order to develop a signal quality based measurement sampling technique. This sampling technique characterises the minimal amount of data required for successful system identification. The developed MATLAB software provides a versatile framework which can be used on different dynamical models.

2 Introduction

The aim of this project is to identify dynamic models of gene regulatory networks (GRNs) from noisy time series data.

Several identification methods already exist for GRNs in the literature [1], however even state-of-the-art techniques are only capable of identifying 43% of regulatory interactions from *Escherichia coli* [2]. An increasingly popular technique is to use regularised linear regression [3], which has a promising new direction called Sparse Bayesian Learning (SBL) [4]. Successful identification using SBL has been done previously on simulated GRN toplogies [5]. Unfortunately, it remains unclear how successful identification is linked with measurement quality and how many measurement points are needed for reconstruction of such topologies.

In abscence of such information, it is imperative to investigate the reconstruction quality of SBL in simulated GRN topologies. The aim of this thesis is to obtain empirical evidence about the minimal number of points (**lower bound**) needed for reconstruction by varying signal to noise ratio of artificially generated data.

It is important to see that low **measurement quality** and **scarcity of data** is a domain-specific problem of biologists [6], so determining such bounds is vital in a broader context than GRNs.

In non-regularised linear regression, identification of systems with more parameters than measurements leads to an ill-conditioned problem. Regularisation aims to address these problems by parameter shrinkage, however, the extent of the shrinkage is often ad-hoc, or it is based on cross-validation which has been proven to be an optimistic estimator of the generalisation capability of a model [7]. An identification framework will be introduced which directly relates **measurement quality** to regularisation in order to establish a disciplined choice of shrinkage parameters. An additional advantage of this framework is its ability to promote parameter sparsity.

The **scarcity of data** often cannot be alleviated, however, the minimal amount of required measurements for identification can be experimentally determined and linked with the signal to noise ratio. This is crucial for cost-efficient design and implementation of experiments.

The structure of the thesis is the following. First, in Section 3.1, methods will be shown on how to generate data from GRNs based on their dynamic models. At the end of that section, a discrete linear model will be introduced for data synthesis and identification.

In Section 3.2 the SBL framework will be introduced for identification which incorporates sparsity and signal quality information into the estimation process.

In order to determine the minimal amount of required measurements, it is first noted in Section 3.3 results from the field of Optimal Experiment Design will be used to impose an order on data samples based on their informativeness.

In Section 3.4, numerical studies are proposed for determining these lower bounds on required measurements and influence of the signal to noise ratio.

The main contributions of this thesis are,

- a MATLAB framework to construct and identify dynamical models GRNs using the Sparse Bayesian learning framework
- a measurement sampling technique which can be used to order samples by their informativeness
- a set of guidelines for identification of GRNs to biologists and bioengineers

3 Methods

3.1 Simulation Framework

3.1.1 Genetic Regulatory Networks (GRNs)

At the cellular level, proteins and messenger RNAs are the main workers maintaining homeostasis and performing the related signaling interactions [8]. To understand cellular processes, one must understand how proteins and RNAs interact with each other which can be described by GRNs.

GRNs have the capability to sense molecules, transmit signals and actuate by producing new molecules. They can be viewed as the biological counterparts of electrical circuits, and this view allows the use of similar engineering principles to design modern medicine [9].

Gene expression can be controlled mainly on two different levels, namely **transcription** and **translation**. In this thesis, the main focus is on **transcription** regulation [8]. In transcription, a molecule called a transcription factor (see Figure 1) binds to the binding site. This molecule either promotes (**activation**) or impedes (**repression**) binding of RNA-Polymerase, which is the molecular machinery responsible for the transcription of DNA to RNA. This mechanism can be schematically seen on Figure 1.

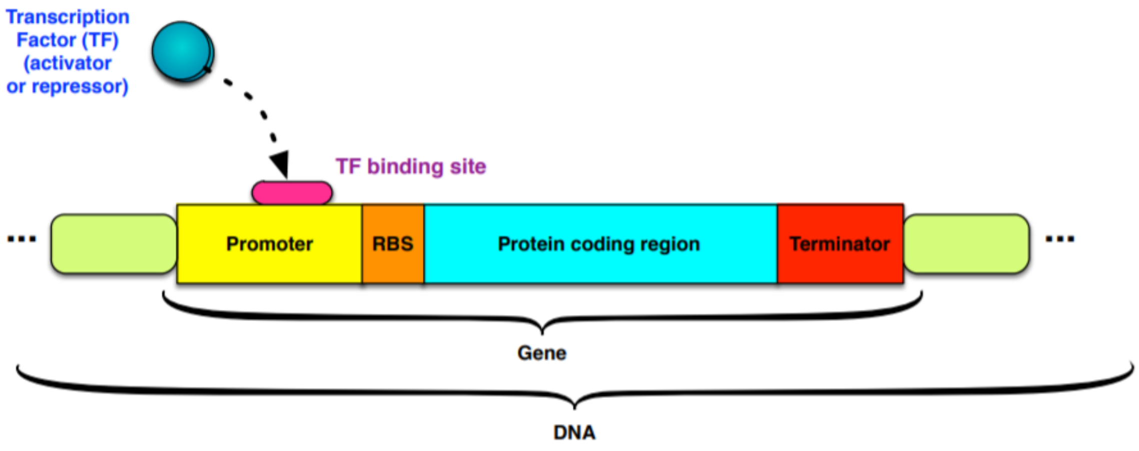

Figure 1: Transcription factor binds on a part of the promoter sequence called binding site. This either impedes or helps production of RNA. The produced protein depends on the instructions (order of DNA bases) in the coding sequence of the DNA (Image from [8]).

Therefore, **activation** and **repression** are the two main interactions in GRNs. However, it is also important that all produced molecules have a characteristic lifetime, this effect is described by **degradation**. These interactions realise complex behaviours such as the repressilator [10]. In order to understand these behaviours on the level of networks, a dynamic model is needed for GRNs.

3.1.2 Dynamic models of GRNs

In this section, a brief overview of GRNs will be given based on [8]. **Activation**, **repression** and **degradation** were briefly introduced in the previous section. In this section dynamical models are introduced which allow a simplified model of GRNs.

Activation and repression in transcription regulation

Transcription regulation can be modeled as an enzyme (E) catalysed substrate (S) product (P) conversion, where there are n molecules of substrates, which results in the chemical equilibrium reaction,

$$\mathrm{E} + \mathrm{nS} \underset{\mathrm{k_{-1}}}{\overset{\mathrm{k_1}}{\rightleftharpoons}} \mathrm{ES} \xrightarrow{\mathrm{k_2}} \mathrm{E} + \mathrm{P}$$

where k_1, k_{-1} and k_2 are non-negative real numbers representing the rates of the reactions.

The law of mass action and quasi-stationary assumptions can then be used to obtain a differential equation for gene expression, (for a full derivation, see [8])

$$\frac{d[P]}{dt} = V_{\max} \frac{[S]^n}{K_M + [S]^n}, \tag{1}$$

where $V_{\max} = nk_2[E]_0$ with $[E]_0$ being the initial concentration of the enzymes, and $K_M = \frac{k_{-1}}{k_2+k_1}$ is the Michaelis-Menten constant. The right hand side of the equation as a function of the substrate is called **Hill function**.

If the parameters above are known, it is possible to solve the differential equation to obtain the dynamics of the concentration $[P]$. In GRNs the enzyme is called the transcription factor and the substrate molecules are either proteins or messenger RNAs. In this thesis, the substrates will be assumed to be messenger RNAs.

A simplifying assumption of this model is that the binding of the n substrates happens simultaneously. It is possible that after one substrate is bound, the affinity for binding increases or decreases.

On Figure 2, the Hill function can be seen for different orders of n.

Due to the simplifying assumptions, the model is usually treated as an approximation which describes the switch-like phenomena observed in gene expression data [8]. A consequence of this approximation is that n is not necessarily an integer number. Thus, it is justified to use the symmetric counterpart of the function in Equation (1) as a phenomenological model for repression,

$$\frac{d[P]}{dt} = \frac{V_{\max}}{[S]^n + K_M}. \tag{2}$$

Definition 1. *The name **activating Hill function of order** n from now on will be used to refer to the function*

$$hillAct(n, x_i) = \frac{x_i^n}{1 + x_i^n}, \tag{3}$$

*where x_i denotes the concentration of **gene** X_i so that $x_i = [X_i]$.*

Definition 2. *Conversely, the reverse behaviour will be defined as the **repressing Hill function of order** n,*

$$hillRep(n, x_i) = \frac{1}{1 + x_i^n}, \tag{4}$$

*where x_i denotes the concentration of **gene** X_i so that $x_i = [X_i]$.*

Remark 1. *In the scope of the thesis, K_M is assumed to be 1 everywhere as you can see in Definition 1 and 2. Although, it is possible to introduce these functions so that they take K_M as an argument, this step will simplify the discussion of GRNs later on.*

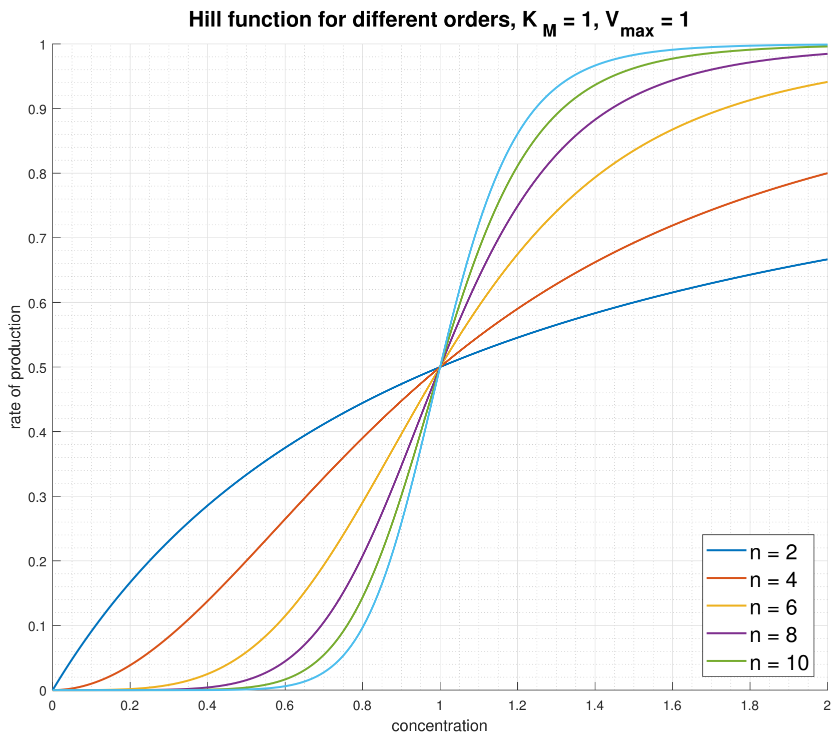

Figure 2: Notice that by increasing the order (n) in Equation (1), a sharper transition can be achieved.

Remark 2. *Note that the maximum and minimum of these functions are 1 and 0, respectively. This is consistent with the thermodynamic intuition so that there cannot be an infinite production of some species. It is also interesting to note, that the higher n is, the more switch-like the Hill-function becomes which can be seen on Figure 2.*

Degradation

Degradation is either the natural decay or dilution of molecular species due to cell growth. It will be modelled as a **first order reaction**, which means that the decay of the molecule is dependent on its concentration linearly.

Definition 3. *From now on the name degradation function will be used to refer to the function*

$$degradation(x_i) = -x_i, \tag{5}$$

where x_i denotes the concentration of ***gene*** *X_i so that $x_i = [X_i]$.*

Remark 3. *This definition of degradation considers only first order degradation, active degradation and second order degradation will not be considered in this thesis.*

Definition 4. *The notation $k_i \in \mathbb{R},\ k_i \geq 0$ will be used for the rate of maximum transcription.*

Definition 5. *The notation $d_i \in \mathbb{R},\ k_i \geq 0$ will be used for the rate of first order degradation for gene X_i.*

Graphical model of GRNs

It is helpful to introduce interactions graphically between gene regulatory networks at this point as this can build more intuitive understanding in GRNs. On Figure 3 you can see that activation is marked with an angled arrow while repression is marked with a slanted arrow.

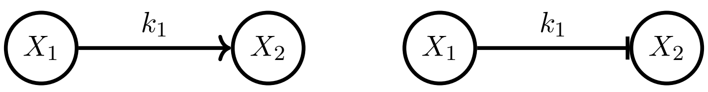

Figure 3: Graphical representation of activation and repression reactions

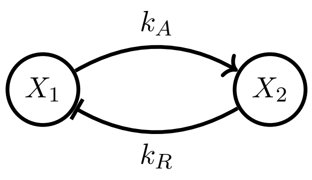

Figure 4: X_1 activates X_2, X_2 represses X_1

The power of such interpretation is better understood by considering the graphical model of a GRN on Figure 4, which can capture the dynamics of the following ODEs,

$$\begin{aligned}\frac{dx_1}{dt} &= k_R \frac{1}{1+x_2^n} - d_1 x_1 \\ \frac{dx_2}{dt} &= k_A \frac{x_1^n}{1+x_1^n} - d_2 x_2.\end{aligned} \tag{6}$$

By looking at Equation (6) it is not trivial what they represent, but with the graph it is easy to grasp the essence of that GRN. By using Definition 1-3, the differential equations above can be rewritten to yet another form,

$$\begin{aligned}\frac{dx_1}{dt} &= k_R \cdot \mathrm{hillRep}(x_2, n) + d_1 \cdot \mathrm{degradation}(x_1) \\ \frac{dx_2}{dt} &= k_A \cdot \mathrm{hillAct}(x_1, n) + d_2 \cdot \mathrm{degradation}(x_2).\end{aligned} \tag{7}$$

which can be used to define gene topologies in computer programs (see Appendix 6.1), and it is also a useful abstraction for separating the linear and nonlinear parameters.

Remark 4. *Notice that the derivative of concentration depend linearly on k_R, k_A, d_1 and d_2 but non-linearly on n. This will be important later in terms of the identification procedure.*

Remark 5. *The sign changes in Equation (7) compared to Equation (6) due the definition of degradation functions in Definition 3. Introducing the function this way enables to take all parameters positive later on.*

We have enough knowledge now to introduce a simple GRN, the repressilator.

3.1.3 GRN benchmark model: the Repressilator

The repressilator is a GRN which acts as a biological oscillator, which was the first artificial GRN [10]. As electrical oscillators vary the amplitude of voltage sinusoidally, the repressilator does the same with the concentrations of genes. Its stability properties are also well studied in the academic literature [10].

A three gene repressilator can also be represented using a graph as you can see on Figure 5. The governing differential equations for $n = 4$ are,

$$\begin{aligned}
\frac{dx_1}{dt} &= k_3 \cdot \text{hillRep}(x_3, 4) + d_1 \cdot \text{degradation}(x_1) \\
\frac{dx_2}{dt} &= k_1 \cdot \text{hillRep}(x_1, 4) + d_2 \cdot \text{degradation}(x_2) \\
\frac{dx_3}{dt} &= k_2 \cdot \text{hillRep}(x_2, 4) + d_3 \cdot \text{degradation}(x_3). \qquad (8)
\end{aligned}$$

In this thesis, our benchmark will be the repressilator model on Figure 5, where each repression is of order 4 with $k_i = 40$ and $d_i = -1$ where $i \in [1, 2, 3]$. The time course of this repressilator model can be seen in absence of measurement noise on Figure 6.

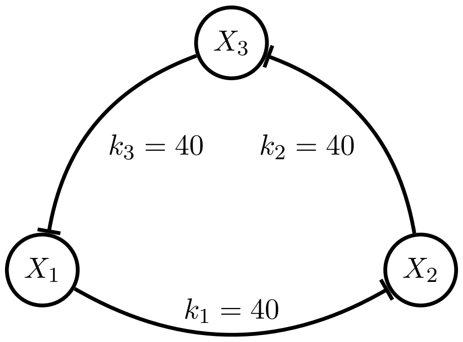

Figure 5: The graphical model of the repressilator. In our benchmark topology, all rates of transcriptions are taken $k_i = 40, \ i \in [1, 2, 3]$, and all degradation constants are taken $d_i = 1, \ i \in [1, 2, 3]$.

3.1.4 Matrix notation for GRNs in discrete time

In Section 3.1.2, a differential equation model for GRNs have been introduced. In this section a matrix notation for GRNs will be considered after discretisation of the problem.

Consider first the nonlinear differential equations for GRNs,

$$\frac{dx_i(t)}{dt} = \sum_{a=1}^{b} \theta_a \mathcal{G}_a(\mathbf{x}), \quad x_i(t_0) \in \mathbb{R}^{m \times 1}, \quad i \in [1, 2, \ldots, m], \qquad (9)$$

where $\mathcal{G}_a(\cdot) : \mathbb{R}^{m \times 1} \to \mathbb{R}$ will be called a dictionary function mapping the m continous functions $\mathbf{x} = [x_1(t), x_2(t), \ldots, x_m(t)] \in \mathbb{R}^{m \times 1}$ to a scalar, m is the number of genes, b is the total number of functions considered, $\theta_a \geq 0$ and $\theta_a \in \mathbb{R}$ is a coefficient weighting the influence of the a^{th} dictionary function, and $x(t_0)$ is the initial conditions of all gene concentrations.

In this thesis, the discussion will be limited to functions $f_a(\cdot) : \mathbb{R} \to \mathbb{R}, \quad a \in [1, \ldots, b]$ instead of $\mathcal{G}_a(\cdot)$ in Equation (9). This means that we assume that each function will take **only one gene concentration as an argument**.

Remark 6. *The functions considered are described in Section 3.1.2. The function $f_a(\cdot)$ can be either linear (i.e. degradation$(\cdot)$) or nonlinear (i.e. hillRep$(\cdot)$).*

First, the concentrations are discretised. The function $x_i(t)$ is discretised by considering a total of k samples (including the initial condition) at times $t_0, t_2, ..., t_{k-1}$. Then, a matrix containing all samples of all genes can be defined as,

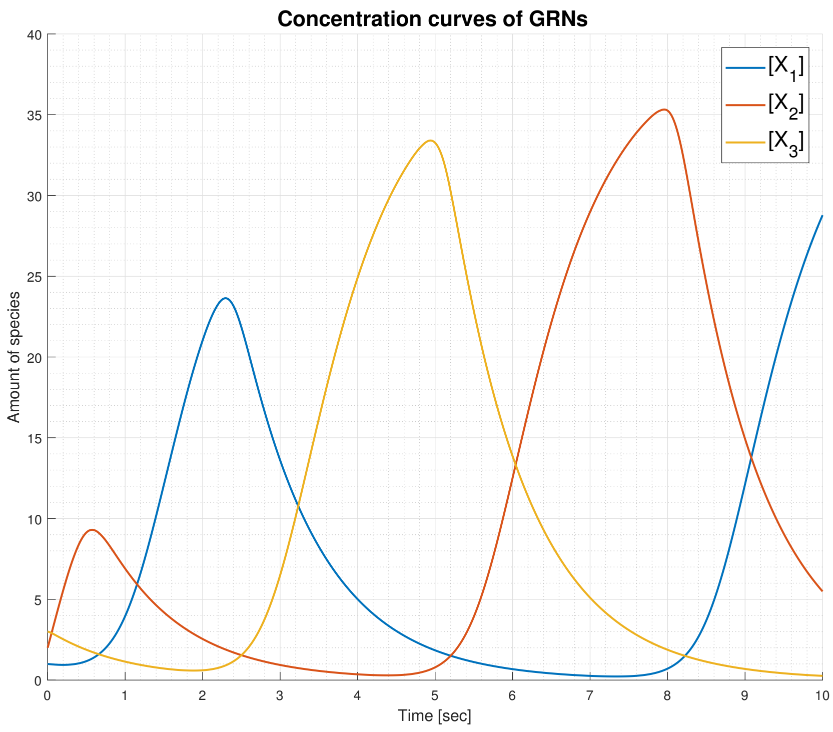

Figure 6: Time course simulation of the benchmark repressilator model. Each curve represents the concentration of a different gene as marked in the legend of the graph. The initial conditions were $x_1(t_0) = 1, x_2(t_0) = 2, x_3(t_0) = 3$.

$$\mathcal{X} = \begin{bmatrix} x_1(t_0) & x_1(t_1) & \dots & x_1(t_{k-1}) \\ x_2(t_0) & x_2(t_1) & \dots & x_2(t_{k-1}) \\ x_3(t_0) & x_3(t_1) & \dots & x_3(t_{k-1}) \\ \vdots & \vdots & \vdots & \vdots \\ x_m(t_0) & x_m(t_1) & \dots & x_m(t_{k-1}) \end{bmatrix} \in \mathbb{R}^{m\times k}. \tag{10}$$

It is assumed that the derivatives can be approximated from the concentrations at this point. This assumption and its limitations will be further discussed in Section 5.2. Here the derivative is approximated from the samples using the Euler formula $\dot{x}_i \approx \frac{x_i(t_{j+1})-x_j(t_j)}{\Delta t} \quad j \in [0, 1, \dots, k-1]$, taking $\Delta t = t_{j+1} - t_j$ sufficiently small. Then a vector can be defined as

$$\mathbf{y}^i = \dot{\mathbf{x}}^i = \begin{bmatrix} \frac{x_i(t_1)-x_i(t_0)}{\Delta t} \\ \frac{x_i(t_2)-x_i(t_1)}{\Delta t} \\ \frac{x_i(t_3)-x_i(t_2)}{\Delta t} \\ \vdots \\ \frac{x_i(t_k)-x_i(t_{k-1})}{\Delta t} \end{bmatrix} \in \mathbb{R}^{k\times 1}, \quad i \in [1, 2, \dots, m]. \tag{11}$$

A dictionary of nonlinear descriptors can be defined if we consider evaluation of these functions on each gene and all samples of $\mathcal{X}$,

$$\mathbf{\Phi}^i = \begin{bmatrix} f_1(x_1(t_0)) & f_1(x_2(t_0)) & \dots & f_1(x_m(t_0)) & f_2(x_1(t_0)) & \dots & f_b(x_m(t_0)) \\ f_1(x_1(t_1)) & f_1(x_2(t_1)) & \dots & f_1(x_m(t_1)) & f_2(x_1(t_1)) & \dots & f_b(x_m(t_2)) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f_1(x_1(t_{k-1})) & f_1(x_2(t_{k-1})) & \dots & f_1(x_m(t_{k-1})) & f_2(x_1(t_{k-1})) & \dots & f_b(x_m(t_{k-1})) \end{bmatrix} \in \mathbb{R}^{k\times l}, \tag{12}$$

where $l = b \cdot m$. We wish to approximate then Equation (9) by a linear sum of these functions by introducing the vector $\mathbf{w}^i \in \mathbb{R}^{l\times 1}$.

The dictionary of nonlinearities are independent of which gene we estimate, just the weights change, so $\mathbf{\Phi}^i = \mathbf{\Phi}$. From now on, the superscript i will be dropped from $\mathbf{w}^i$ and $\mathbf{y}^i$, as the remainder will apply for each gene i.

This can be formalised by the following definition:

Definition 6. *A particular gene X_i in a GRN is described uniquely by the differential equation model $\mathcal{M}^*$*

$$\mathcal{M}^* : \mathbf{y} = \mathbf{\Phi}\mathbf{w} + \epsilon, \tag{13}$$

*where $\mathbf{\Phi} \in \mathbb{R}^{k\times l}$ is a dictionary defined as in Equation (12). The vector $\mathbf{y} \in \mathbb{R}^{k\times 1}$ are the derivative series which are defined as in Equation (11) and $\mathbf{w} \in \mathbb{R}^{l\times 1}$ are the mixing coefficients of the descriptors. The notation ϵ refers to additive white Gaussian noise $\mathcal{N}(0, \lambda\mathbf{I})$, where $\lambda \geq 0$ is the variance. The notation $\mathcal{M}^*$ will be used to refer to this as the **generative model**.*

Worked example 1. *An example for a difference equation for a gene in a GRN is,*

$$\frac{x_1(t_{j+1}) - x_1(t_j)}{\Delta t} = k_1 \frac{1}{1 + x_2(t_j)^4} - d_1 x_1(t_j), \quad j \in [0, 1, \dots, k-1]. \tag{14}$$

The discretised matrix notation is sought for $k = 3$ samples. The $b = 2$ dictionary functions are $f_1(x_i) = hillRep(x_i, 4) = \frac{1}{1+x_i(t)^4}$, and $f_2(x_i) = degradation(x_i) = -x_i$. Evaluating for each gene $m = 2$ results in $l = 4$ dictionary functions,

$$\mathbf{\Phi} = \begin{bmatrix} \frac{1}{1+x_1(t_0)^4} & \frac{1}{1+x_1(t_1)^4} & \frac{1}{1+x_1(t_2)^4} \\ \frac{1}{1+x_2(t_0)^4} & \frac{1}{1+x_2(t_1)^4} & \frac{1}{1+x_2(t_2)^4} \\ -x_1(t_0) & -x_1(t_1) & -x_1(t_2) \\ -x_2(t_0) & -x_2(t_1) & -x_2(t_2) \end{bmatrix}^T \quad \mathbf{w} = \begin{bmatrix} 0 \\ k_1 \\ d_1 \\ 0 \end{bmatrix} \quad \epsilon = 0. \tag{15}$$

By solving the difference equation it is now possible to synthesise measurements from GRNs, by appropriate selection of functions and coefficients as in the worked example above. In addition, by knowledge of the synthesised data and the numerical approximation of the derivatives of the data, it is possible to define the identification model.

Definition 7. *A dynamic behaviour of a particular gene X_i in a GRN can be identified with the linear model $\mathcal{M}$*

$$\mathcal{M} : \mathbf{y} = \mathbf{\Phi}\mathbf{w} + \epsilon, \tag{16}$$

*where $\mathbf{\Phi} \in \mathbb{R}^{k\times l}$ is the dictionary defined as in Equation (12). The vector $\mathbf{y} \in \mathbb{R}^{k\times 1}$ are the derivative series which are defined as in Equation (11) and $\mathbf{w} \in \mathbb{R}^{l\times 1}$ are **unknown coefficients**. The notation ϵ refers to additive white Gaussian noise $\mathcal{N}(0, \lambda\mathbf{I})$, where $\lambda \geq 0$ is the variance. The notation $\mathcal{M}$ will be used to refer to this **identification model**.*

The above problem can be solved by **linear regression**, however this has some challenges.

3.2 Sparse Bayesian Learning

3.2.1 Motivation for Bayesian Regularisation

The identification problem introduced in the previous section is a linear regression problem which are easy to solve using a Least Squares algorithm. However, in our case, this is problematic due to two reasons.

1. Underdetermined case

The first reason is that the length of the dictionary l might be a smaller than the the number of samples k. The Least Squares estimation can be done using the formula

$$\hat{\mathbf{w}} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{y}. \tag{17}$$

In order to invert $\mathbf{\Phi}^T\mathbf{\Phi}$, it has to be full rank. This is not the case if $l < k$. A remedy for this is to calculate $\hat{\mathbf{w}}_\lambda = (\mathbf{\Phi}^T\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^T\mathbf{y}$, which is called Ridge **regularisation** [11]. (see Figure 3.2.1)

In regularisation the choice of λ is often based on cross-validation [12]. Nevertheless, one would wish to know an optimal selection of λ, which is independent from an arbitrary validation set.

2. Overfitting

In the identification model, the dictionary functions $f_a(\cdot)$, $a \in [1, 2, \ldots, b]$ might not be orthogonal, meaning there could be multiple linear combinations of functions explaining a datum point. This means that for a limited amount of noisy measurements there could be an estimate with lower variance than the true model. Calculating $\hat{\mathbf{w}}$ using Equation (17), one typically finds that each dictionary function is assigned some weight to fit the noisy derivative $\mathbf{y}$.

A solution would be to fit some **subset** of the dictionary and choose the model which achieves an optimal trade-off in the number of parameters and performance. The field of Model Selection offers various ways to do so, such as AIC, BIC or cross-validation [12]. But, this is a tedious task to do for dictionaries where l is large, because it results in

$$c = \binom{l}{1} + \binom{l}{2} + \ldots + \binom{l}{l}, \tag{18}$$

models to fit and test. An ideal solution would be to incorporate some penalty on the number of parameters in the optimisation objective, which could induce **sparse** solutions.

The optimisation objective of regularisation does exactly that by solving the least squares normal, while simultaneously minimising the L_p norm on the weights (see Figure 3.2.1). The two most popular penalties are Ridge [11] (see (A) on Figure 3.2.1) and LASSO [7] (see (B) on Figure 3.2.1),

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} ||\mathbf{\Phi}\mathbf{w} - \mathbf{y}||_2^2 + \lambda||\mathbf{w}||_2 \quad \text{(Ridge)} \tag{19}$$
$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} ||\mathbf{\Phi}\mathbf{w} - \mathbf{y}||_2^2 + \lambda||\mathbf{w}||_1. \quad \text{(LASSO)} \tag{20}$$

While giving sparse solutions, the equations above also introduce a new hyperparameter λ. Also, it does not solve the combinatorial search problem in Equation (18). The best would be, if we could find the best columns of $\mathbf{\Phi}$ that fits our model. The notion of the best columns is characterised by the L_0 norm, which penalises for each nonzero rows equally. This process is called **best subset selection** [7].

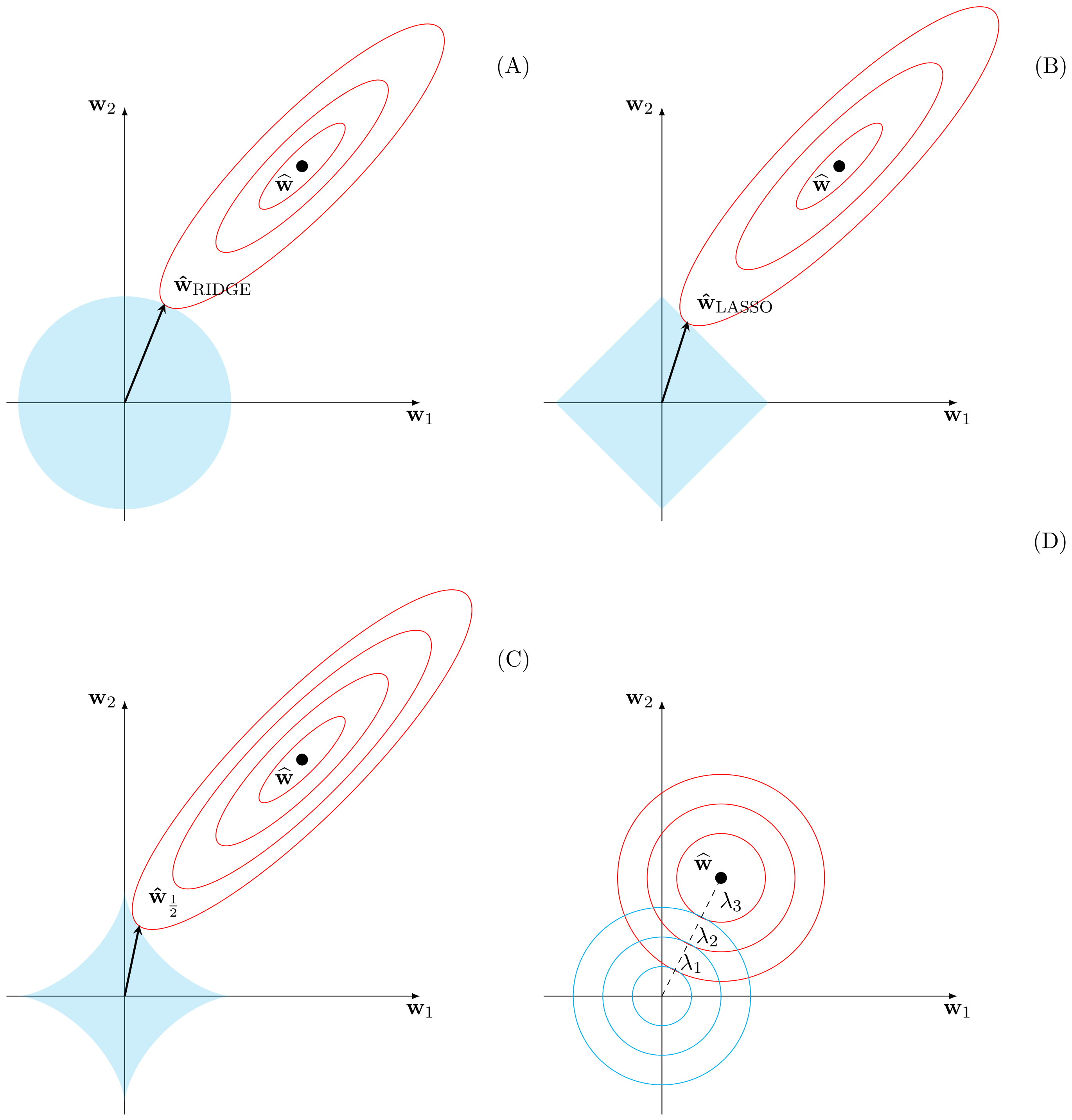

Figure 7: A geometric interpretation of regularisation can be considered by recognising that the Least Squares is an ellipsoid centered at the solution $\hat{\mathbf{w}}$, while the L_2 is an equation of a circle centered at the origin (see (A)). For this reason, the L_2 norm is also called a norm ball [13], and other norms also have geometric meaning, the L_1 is a diamond (B), and the even sparser $L_{\frac{1}{2}}$ norm is a shape bounded by hyperboles on its circumference (C). Notice the parameter shrinkage due to the increasingly sparser norms. On (D), observe how the choice of $\lambda_1 > \lambda_2 > \lambda_3$ affects Ridge solution. We can see that $\hat{\mathbf{w}}_{\text{RIDGE}}$ is a point which lies on the line between $\hat{\mathbf{w}}_{\text{OLS}}$ and the origin. The trade-off parameter λ controls the exact place on the line, a larger λ meaning the new center of the circle will be closer to the origin. (Figure based on [7])

In the Bayesian interpretation, Equation (19) and (20) can be obtained as the maximum *a posteriori* objective, given a prior probability on the distribution of the weights $p(\mathbf{w})$. A derivation for the Ridge function can be seen in the Appendix 6.3. The key takeaway note from the Ridge proof is that the penalty parameter becomes interpretable, however choosing the variance of the weight prior is subjective. In order to obtain the weight prior objectively, Tipping in [14] proposed to obtain this parameter so that it maximises the evidence $p(\mathbf{y})$, which is called **Bayesian interpolation**. The SBL in the next section provides an alternative technique for choosing λ by choosing the variance of a sparsity-inducing weight prior $p(\mathbf{w})$.

3.2.2 Sparse Bayesian Learning Algorithm

In this section, the Sparse Bayesian Learning (SBL) framework will be introduced for the estimation of the weight vector. The aim of the section is to provide a brief outline of how the SBL algorithm works, and how is it possible to relate the penalty parameter λ to the process noise. The following introduction is based originally on [15] and the description in [16] greatly helped the understanding of the author.

It is easy to lose context in the introdution of this framework, so the following list contains all the major stepping stones.

- formulation of the linear regression problem as a MAP optimisation problem and showing how λ is related to the variance
- derivation of a cost function based on evidence maximisatio where we conclude that it is not convex
- obtaining the dual problems which is **convex**
- obtain the formula for sparse prior hyperparameter γ and reweighting parameter $\mathbf{z}$
- description of the computer algorithm

Formulation of the linear regression problem as a MAP optimisation problem

Suppose that we wish to estimate a stochastic process using a linear model. The linear model $\mathcal{M}$ has the form

$$\mathbf{y} = \mathbf{\Phi}\mathbf{w} + \epsilon, \tag{21}$$

where $\mathbf{\Phi} \in \mathbb{R}^{k\times l}$, $\mathbf{w} \in \mathbb{R}^{l\times 1}$, $\mathbf{y} \in \mathbb{R}^{k\times 1}$, and ϵ is additive white Gaussian noise, $\mathcal{N}(0, \lambda\mathbf{I})$. The goal of the estimation problem is to find an estimate of $\mathbf{w}$, called $\hat{\mathbf{w}}$, which is sparse.

The motivation for the introduction of the Bayesian framework is to obtain an interpretation for the sparse penalty parameter λ. Compare the equations below and notice that λ can be divided through to the variance,

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} ||\mathbf{\Phi}\mathbf{w} - \mathbf{y}||_2^2 + \lambda||\mathbf{w}||_0 \tag{22}$$

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} p(\mathbf{y}|\mathbf{w})p(\mathbf{w}). \tag{23}$$

The first term in Equation (22) and (23) will be maximal when the model fits the available data well and the second term in (22) and (23) will be maximal when the weights' distribution are a plausible sample from the prior distribution. Thus to achieve an approximation to the problem above, the aim is to find a $p(\mathbf{w})$ which represents $||\mathbf{w}||_0$ well.

A variational distribution of priors are considered for this, $p(w_i) = \max_{\gamma_i \geq 0} \mathcal{N}(w_i; 0, \gamma_i)\zeta(\gamma_i)$, where $\zeta(\cdot) : \mathbb{R} \to \mathbb{R}$ is a non-negative penalty function [17]. This distribution is able to take a variety of priors depending on the hyperparameter $\gamma \in \mathbb{R}^{l \times 1}$. The variational distribution of priors can then be written as $p(\mathbf{w}) = \mathcal{N}(0, \mathbf{\Gamma}) \prod_{i=1}^{l} \zeta(\gamma_i)$, where $\mathbf{\Gamma} = \operatorname{diag}(\gamma)$. The hyperparameter γ can be chosen so that it maximises the evidence $p(\mathbf{y})$ in a Bayesian interpolation process [14]. An optimisation is then sought to maximise the evidence.

General cost function derivation

The evidence can be maximised by minimising the misalignment between the “true” prior and the variational representation of the prior. The “true” prior is not known, but it is a quantity which is at best the same as the prior. Practically, it is a relaxation which works at the places where $p(\mathbf{y}|\mathbf{w})$ is high. The penalty function can be factored out because it is independent of the weights. This makes the following scheme a **separable** penalty scheme [18].

$$\begin{aligned}
\hat{\gamma} &= \arg\min_{\gamma} \int p(\mathbf{y}|\mathbf{w})[p(\mathbf{w}) - p(\gamma; \mathbf{w})]\mathbf{dw} \\
\hat{\gamma} &= \arg\max_{\gamma} \int p(\mathbf{y}|\mathbf{w}) p(\gamma; \mathbf{w})\mathbf{dw} \\
\hat{\gamma} &= \arg\max_{\gamma} \prod_{i=1}^{l} \zeta(\gamma_i) \int p(\mathbf{y}|\mathbf{w}) \mathcal{N}(0, \mathbf{\Gamma})\mathbf{dw}. \qquad (24)
\end{aligned}$$

It can be proven that for a Gaussian marginal distribution on $p(\mathbf{w})$ and a Gaussian conditional distriution $p(\mathbf{y}|\mathbf{w})$, the marginal for $p(\mathbf{y})$ will be Gaussian with formula $\mathcal{N}(0, \lambda\mathbf{I} + \mathbf{\Phi\Gamma\Phi}^T)$, see page 93 in [12].

Using a $-2\ln(\cdot)$ transformation, it is possible to deal with the exponent of the Gaussian and arrive at the loss function,

$$\mathcal{L}(\gamma) = \mathbf{y}^T \mathbf{\Sigma_y}^{-1} \mathbf{y} + \ln(|\mathbf{\Sigma_y}|) + \sum_{i=1}^{l} s(\gamma_i), \qquad (25)$$

where $\mathbf{\Sigma_y} = \lambda\mathbf{I} + \mathbf{\Phi\Gamma\Phi}^T$ and $s(\gamma_i) = -2\ln(\zeta(\gamma_i))$ mapping of the penalty function. The desired γ minimises the loss function above. Note, that the function is **not convex**, thus in this form it is not amenable to convex optimisation techniques.

Dual representation

We seek a convex representation of the above function in an iterative scheme.

Notice that the data-dependent term $\mathbf{y}^T \mathbf{\Sigma_y}^{-1} \mathbf{y}$ is a Ridge optimisation problem on $\mathbf{w}$ if we consider $\mathbf{\Gamma}$ fixed, because we marginalised Equation (24), with a normal prior on the weights. This point is further elaborated in Section 6.3. This observation results in,

$$\mathbf{y}^T \mathbf{\Sigma}_{\mathbf{y}}^{-\mathbf{1}} \mathbf{y} = \min_{\mathbf{w}} \frac{1}{\lambda} ||\mathbf{y} - \mathbf{\Phi w}||_2^2 + \sum_{i=1}^{l} \frac{w_i^2}{\gamma_i}. \qquad (26)$$

This hints that decoupling and successively optimising for γ and $\mathbf{w}$ can be written as a convex problem. Substituting into the loss function, a joint optimisation scheme is obtained,

$$\mathcal{L}(\gamma, \mathbf{w}) = \min_{\mathbf{w}} \frac{1}{\lambda} ||\mathbf{y} - \mathbf{\Phi w}||_2^2 + \sum_{i=1}^{l} \frac{w_i^2}{\gamma_i} + \ln(|\mathbf{\Sigma_y}|) + \sum_{i=1}^{l} s(\gamma_i). \qquad (27)$$

At this point, the function of the priors will be assumed to be zero. Different penalty functions would lead to different separable penalty schemes, these are described in [18]. The problem can be interpreted as the joint minimisation of a Ridge problem (first two terms in Equation (27)) and the volume of a confidence ellipsoid (third term) corresponding to the confidence in the evidence function. Note that in some sense this algorithm performs optimal experiment design with the third term γ depending on the measurements. A function $g(\cdot) : \mathbb{R}^{l\times 1} \to \mathbb{R}$ is defined which collects the penalties on the weight magnitude and on the confidence in the evidence,

$$g(\mathbf{w}) = \min_{\gamma\geq 0} \sum_{i=1}^{l} \frac{w_i^2}{\gamma_i} + \ln|\mathbf{\Sigma_y}|. \tag{28}$$

Notice, that $\ln|\mathbf{\Sigma_y}|$ is a concave, non-decreasing function in γ, which is still not amenable for convex optimisation in this form. The dual problem is considered to obtain a convex function [13].

Consider the equation for transforming $\ln|\mathbf{\Sigma_y}|$ into the dual form. Consider a free variable $\mathbf{z} \in \mathbb{R}^{l\times 1}$

$$\ln(\mathbf{\Sigma_y}) = \min_{\mathbf{z}\geq 0} \{\mathbf{z}^T\gamma - h^*(\mathbf{z})\}. \tag{29}$$

Geometrically, the term $\mathbf{z}^T\gamma$ means that in z-space a normal vector of a plane is sought. The term $h^*(\mathbf{z})$ refers to the concave conjugate of $\ln(\mathbf{\Sigma_y})$ which can be expressed as

$$h^*(\mathbf{z}) = \min_{\gamma\geq 0} \{\mathbf{z}^T\gamma - \ln(\mathbf{\Sigma_y})\}. \tag{30}$$

which means that for a fixed normal vector we are looking for a plane which has minimum distance from the logarithm function [13]. That means that the dual problem means nothing else than for a fixed γ it is possible to find a supporting hyperplane which is amenable to convex optimisation.

This leads to the function

$$g(\mathbf{w}) = \min_{\gamma,\mathbf{z}\geq 0} \mathbf{w}\mathbf{\Gamma}^{-1}\mathbf{w} + \mathbf{z}^T\gamma - h^*(\mathbf{z}). \tag{31}$$

Now, the problem is casted into a convex optimisation problem. The interested reader is lead to the articles [19] and [20] which discuss these decoupling methods and the dual problem more deeply.

Formula for γ and w

The aim is to succesively optimise over γ , $\mathbf{z}$ and $\mathbf{w}$. For γ, we take the other two fixed. The notation $\gamma^* \in \mathbb{R}^{l\times 1}$ and $\mathbf{z}^* \in \mathbb{R}^{l\times 1}$ will be used to denote the local minimisers,

$$0 = \frac{\partial g}{\partial \gamma_i} = \frac{-w_i^2}{\gamma_i^{*2}} + z_i \qquad\qquad \gamma_i^* = ||w_i||_1 z_i^{-1/2}. \tag{32}$$

This can be resubstituted to obtain $g(\mathbf{w}) = \min_{\mathbf{z}\geq 0} \sum_{i=1}^{l} 2z_i^{-1/2}||w_i||_1$.

For $\mathbf{z}^*$, we fix γ. Notice by re-substituting the conjugate in $g(\mathbf{w})$, that this is just equivalent to the vector derivative of the $\ln|\mathbf{\Sigma_y}|$. This is not a trivial derivation, but it is known that $\nabla_\gamma \ln \mathbf{\Sigma_y} = \mathbf{tr}[\mathbf{\Sigma_y}^{-1}\frac{\partial \mathbf{\Sigma_y}}{\partial \mathbf{\Gamma}}]$ which is equivalent to $\mathbf{z}^* = \text{diag}[\mathbf{\Phi}\mathbf{\Sigma_y}^{-1}\mathbf{\Phi}^T]$.

Iterative algorithm for SBL

An algorithm can now be constructed for the SBL algorithm, which can be seen on Algorithm 1.

Algorithm 1: Sparse Bayesian Learning Algorithm

```
Result: $\hat{\mathbf{w}}$ which is a sparse estimate of $\hat{\mathbf{w}}$
Input: Nonlinear decriptors constructed from time series $\mathbf{\Phi}$, derivative series $\mathbf{y}$, noise power $\lambda$ ;
Initialise $\mathbf{z}^* = [1, 1, 1, ....1]^T$;
repeat
    Find $\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} ||\mathbf{y} - \mathbf{\Phi w}||_2^2 + 2\lambda \sum_{i=1}^{l} z_i^{-1/2} ||w_i||_1$;
    Calculate $\gamma_i^* = z_i^{-1/2} ||\hat{w}_i||_1$ ;
    Calculate $\mathbf{\Gamma} = \text{diag}[\gamma_i^*]$ ;
    Calculate $\mathbf{z}^* = \text{diag}[\mathbf{\Phi}^T (\lambda \mathbf{I} + \mathbf{\Phi \Gamma \Phi^T})^{-1} \mathbf{\Phi}]$ ;
until until convergence of $\gamma^*$;
```

3.3 Optimal Measurement Data Sampling

3.3.1 Relation to Optimal Experiment Design

In order to establish bounds on the minimal amount of measurements needed for identification, we have to order our measurements based on some informativeness metric. Trivially, an infinite amount of measurements with zero input do not help to identify the model parameters related to the input (i.e. an input gain parameter).

This idea is expressed in the field of System Identification as the problem of optimal experiment design [21].

Definition 8. *The problem of optimal experiment design is*

$$\mathbf{\Phi} = \arg\min_{\mathbf{\Phi}} \alpha(\mathbf{\Sigma}_{\mathbf{w}}(\mathbf{\Phi})), \tag{33}$$

where $\mathbf{\Sigma}_{\mathbf{w}} \in \mathbb{R}^{l \times l}$ is the covariance matrix of the weights and $\alpha(\cdot) : \mathbb{R}^{l \times l} \to \mathbb{R}$ is a function which maps a square matrix to a scalar value, and k is the total number of input samples to use. $\mathbf{\Phi}$ is a $\mathbb{R}^{k \times l}$ matrix containing the l predictors of a model $\mathcal{M}$.

Remark 7. *The optimisation objective of optimal experiment design and maximum likelihood estimation is identical, but here the optimisation is subject to the input, whereas in maximum likelihood, it is subject to the parameters.*

The problem of optimal experiment design is parallel with the optimal measurement data sampling algorithm that is sought. It is a relaxation of the optimal input design problem, where a set of inputs are already given and the aim is to choose the best ones.

Definition 9. *An optimal measurement sampling algorithm is*

$$\mathbf{\Phi}_p = \arg\min_{\mathbf{\Phi}_p} \alpha(\mathbf{\Sigma}_{\mathbf{w}}(\mathbf{\Phi}_p)]), \tag{34}$$

where $\mathbf{\Sigma}_{\mathbf{w}} \in \mathbb{R}^{l \times l}$ is the error covariance matrix of the weights, where the error is $\epsilon = \hat{\mathbf{w}} - \mathbf{w}$ and $\alpha(\cdot) : \mathbb{R}^{l \times l} \to \mathbb{R}$ is a function which maps any square matrix with $p \in \mathbb{Z}^{+}, \quad p \leq k$ rows and columns to a scalar. $\mathbf{\Phi}_p$ is a $\mathbb{R}^{p \times l}$ matrix containing the l predictors of a model $\mathcal{M}$, and p is the total number of samples to use.

This optimisation objective above is only sensible if the model used is unbiased [21]. The conditions of unbiasedness will be discussed in the next section.

3.3.2 Bias-variance trade-off

The bias-variance trade-off is a central result in estimation theory [7], namely that the variance of an estimator has three components:

- the irreducible power, the part that is inherently stochastic $\mathrm{Var}[\mathbf{y}]$
- the square of the bias of the estimator, which are due to the simplicity of our model $(\mathbf{y} - \mathbb{E}[\hat{\mathbf{y}}])^2$
- the variance of the the estimator, which is often a target for minimisation $\mathrm{Var}[\hat{\mathbf{y}}]$

When the error in an estimator is mostly due to the square of the bias of the estimator, the problem is often called **underfitting**, because there are not enough parameters or nonlinear functions to fit the data well. When the variance of the estimator is the problem, the problem is called **overfitting**, because the noise is fitted, which greatly reduces generalisation capability.

In Section 3.1 a set of dynamic models have been introduced that were used for generating data from GRNs. In the learning framework that was introduced in Section 3.2 the model used for fitting the data contains the same dictionary functions $\mathbf{\Phi}$. Because there exists a linear combination of $\mathbf{\Phi}$ which results in the generative model, ent of the models that have been used for fitting.

Assuming that the SBL algorithm performs optimal subset selection, the estimate is unbiased. Thus the error in estimation is due to overfitting. The aim is to reduce overfitting by optimal data sampling then.

3.3.3 A solution to the optimal sampling problem

Now, it is possible to solve the minimisation problem in Definition 9 for the SBL algorithm. The quantity is the covariance of the weights [16],

$$\mathbf{\Sigma_w} = (\mathbf{\Gamma}^{-1} + \mathbf{\Phi^T\Phi}(\lambda\mathbf{I})^{-1})^{-1}. \tag{35}$$

The problem is that $\mathbf{\Gamma}$ is not known before running the algorithm, making the optimal sampling problem intractable before the actual identification.

At the start of the initialisation $\mathbf{\Gamma} = \text{diag}[1, 1, \ldots, 1] = \mathbf{I}$, in this case, that would mean all inputs are considered equally important in the sampling problem. In that case, the optimisation objective is the minimisation of

$$\mathbf{\Sigma_w^{\text{MAP}}} = (\mathbf{I} + \mathbf{\Phi^T\Phi}(\lambda\mathbf{I})^{-1})^{-1}. \tag{36}$$

The mapping $\alpha(\cdot)$ (see Section 3.3) used will be the determinant, because that will avoid inversion of the matrix, by exploiting the determinant identity $\frac{1}{|\mathbf{A}|} = |\mathbf{A}^{-1}|$. That means that the minimisation problem will be recasted to the maximisation of

$$\mathcal{L}(\mathbf{\Phi}) = \max_{\mathbf{\Phi}} |(\lambda\mathbf{I} + \mathbf{\Phi^T\Phi})|, \tag{37}$$

by noticing that multiplying and taking the determinant is an affine transformation and does not change the objective. This can be used to stabilise the determinant of the matrix.

An algorithm can be constructed that maximises the quantity in Equation (37). The key idea is to solve the optimal selection problem by recursively finding the maximally informative element. The full algorithm can be seen on Algorithm 2.

Algorithm 2: Inverse MAP covariance based sampling

```
Result: p row dictionary $\mathbf{\Phi}_p$ with maximal $\mathcal{L}(\mathbf{\Phi}_p)$
Solve for p-1 recursively ;
for each measurement 1:t do
    Concatenate with p-1 solution;
    Calculate $\mathcal{L}(\mathbf{\Phi}) = |\lambda\mathbf{I} + \mathbf{\Phi}_p^T\mathbf{\Phi}_p|$
end
Perform maximum search for $\mathcal{L}(\mathbf{\Phi}_p)$;
Return $\mathbf{\Phi}_p$ ;
```

3.4 Simulation studies

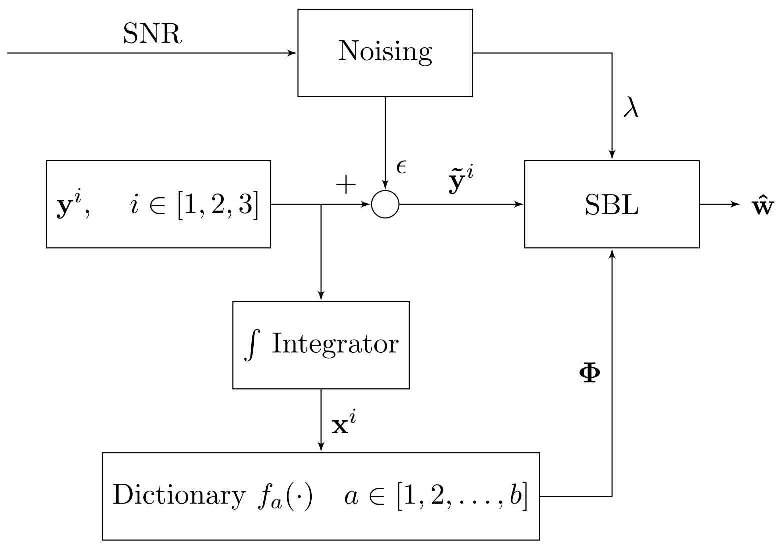

Figure 8: Experimental setup for testing the influence of noise

In this section, a list of numerical simulations will be executed to test hypotheses regarding the identification lower bounds of the SBL algorithm in a biological setting.

3.4.1 Influence of the signal to noise ratio

To the test influence of noise, a benchmark repressilator topology (see Section 3.1.3) is identified using the experimental setup on Figure 8. The main parts are now explained in detail.

Derivatives

A benchmark repressilator topology was modelled as described in Section 3.1.3. The analytical formulas were evaluated by substitution after integration of the formulas.

Integrator

For integration, `ode45` were used. The initial conditions were $x_1(t_0) = 1, x_2(t_0) = 2, x_3(t_0) = 3$. The time interval of simulation is 10 seconds, with a sample each 10^{-3} miliseconds, which resulted in $k = 1001$ samples.

Dictionary

The nonlinear descriptors used were degradation functions along with activating and repressing Hill functions up to order 4. The dictionary was generated using Equation (12). The total number of descriptors were $l = 27$, the dictionary used was,

$$\mathbf{\Phi} = \begin{bmatrix} \text{degradation}(x_1(t_1)) & \text{degradation}(x_1(t_1)) & \dots & \text{degradation}(x_1(t_k)) \\ \text{degradation}(x_2(t_1)) & \text{degradation}(x_2(t_2)) & \dots & \text{degradation}(x_2(t_k)) \\ \text{degradation}(x_3(t_1)) & \text{degradation}(x_3(t_2)) & \dots & \text{degradation}(x_3(t_k)) \\ \text{hillAct}(1, x_1(t_1)) & \text{hillAct}(1, x_1(t_2)) & \dots & \text{hillAct}(1, x_1(t_k)) \\ \text{hillAct}(1, x_2(t_1)) & \text{hillAct}(1, x_2(t_2)) & \dots & \text{hillAct}(1, x_2(t_k)) \\ \text{hillAct}(1, x_3(t_1)) & \text{hillAct}(1, x_3(t_2)) & \dots & \text{hillAct}(1, x_2(t_k)) \\ \text{hillRep}(1, x_1(t_1)) & \text{hillRep}(1, x_1(t_2)) & \dots & \text{hillRep}(1, x_1(t_k)) \\ \text{hillRep}(1, x_2(t_1))) & \text{hillRep}(1, x_2(t_2)) & \dots & \text{hillRep}(1, x_2(t_k)) \\ \text{hillRep}(1, x_3(t_1)) & \text{hillRep}(1, x_3(t_2)) & \dots & \text{hillRep}(1, x_3(t_k)) \\ \text{hillAct}(2, x_1(t_1)) & \dots & \dots & \dots \\ \text{hillAct}(2, x_2(t_1)) & \dots & \dots & \dots \\ \text{hillAct}(2, x_3(t_1)) & \dots & \dots & \dots \\ \text{hillRep}(2, x_3(t_1)) & \dots & \dots & \dots \\ \text{hillRep}(2, x_3(t_1)) & \dots & \dots & \dots \\ \text{hillRep}(2, x_3(t_1)) & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots \\ \text{hillAct}(4, x_1(t_1)) & \dots & \dots & \dots \\ \text{hillAct}(4, x_2(t_1)) & \dots & \dots & \dots \\ \text{hillAct}(4, x_3(t_1)) & \dots & \dots & \dots \\ \text{hillRep}(4, x_1(t_1)) & \dots & \dots & \dots \\ \text{hillRep}(4, x_2(t_1)) & \dots & \dots & \dots \\ \text{hillRep}(4, x_3(t_1)) & \dots & \dots & \dots \end{bmatrix}^T \in \mathbb{R}^{1001\times 27} \tag{38}$$

Noising scenarios

The noise was added to the derivative approximation $\mathbf{y}$ to obtain the noised derivative signal $\tilde{\mathbf{y}} \in \mathbb{R}^{k\times 1}$,

$$\tilde{\mathbf{y}} = \mathbf{y} + \epsilon, \tag{39}$$

where $\epsilon = \mathcal{N}(0, \lambda\mathbf{I})$. The value of λ used was determined using,

$$\text{SNR} = \frac{P_{\mathbf{y}^i}}{\lambda}, \tag{40}$$

where $\mathbf{P}_{y^i}$ is the signal power calculated as,

$$\mathbf{P}_{\mathbf{y}^i} = \frac{\sum_{j=1}^{k} ||\mathbf{y}^i(t_j)||_1^2}{k}, \quad i \in [1, 2, 3], \tag{41}$$

and SNR $\in [0.1, 1, 10, 100, 1000]$. The noise was added to the derivative approximation 200 times for each signal to noise ratio.

Sampling

The dictionary rows were sampled using the maximisation of inverse MAP covariance described in Algorithm 2. For each added measurements the value of $p \in [1, 2, \dots, 50]$ increased by one for a total of 50 measurements.

Identification using SBL

The identification was performed as described in Algorithm 1.

Error metric

The root normalised mean square error metric were used for measuring the success of identification. The formula of the RNMSE metric is

$$\text{RNMSE} = \frac{||\mathbf{w} - \hat{\mathbf{w}}||_2}{||\mathbf{w}||_2}. \tag{42}$$

This metric was chosen so that it reflects the error of identification compared to the size of the true weights. For all zero weights, this metric gives 1, which means that an RNMSE larger than 1 is poor performance because an all-zero estimate would perform better on this metric, see Appendix 6.2.

Oracle estimator lower bound

If the SBL were to perform best subset selection with minimal variance estimation, then it would perform a linear regression on the true subset. To compare the SBL, the true subset is solved with a Least Squares estimation, however the Least Squares is underdetermined when $k < l$ as mentioned in Section 3.2.1. To amend this problem the Least Square estimation is solved using,

$$\hat{\mathbf{w}} = \left(\mathbf{\Phi}^T \mathbf{\Phi}\right)^{\dagger} \mathbf{\Phi}^T \mathbf{y}, \tag{43}$$

where the superscript † denotes the Moore-Penrose pseudoinverse.

Software environment

Code for the simulations was written in MATLAB. The iterative optimisation for the SBL algorithm was written using the CVX toolbox [13]. The simulations were done in MATLAB R2017b (9.3.0.713579) 64-bit (glnxa64), using operating system Ubuntu 14.04.5 LTS (GNU/Linux 3.13.0-147-generic x86_64).

4 Results

In the following sections, all results of the numerical simulations were conducted as described in Section 3.4.

4.1 RNMSE results for different SNR values

Simulation results for different signal to noise ratios can be seen on Figure 9-11, where the number of optimally sampled measurements were plotted against the RNMSE metric. It can be seen that with increased number of measurements the SBL algorithm is able to obtain better estimates in general.

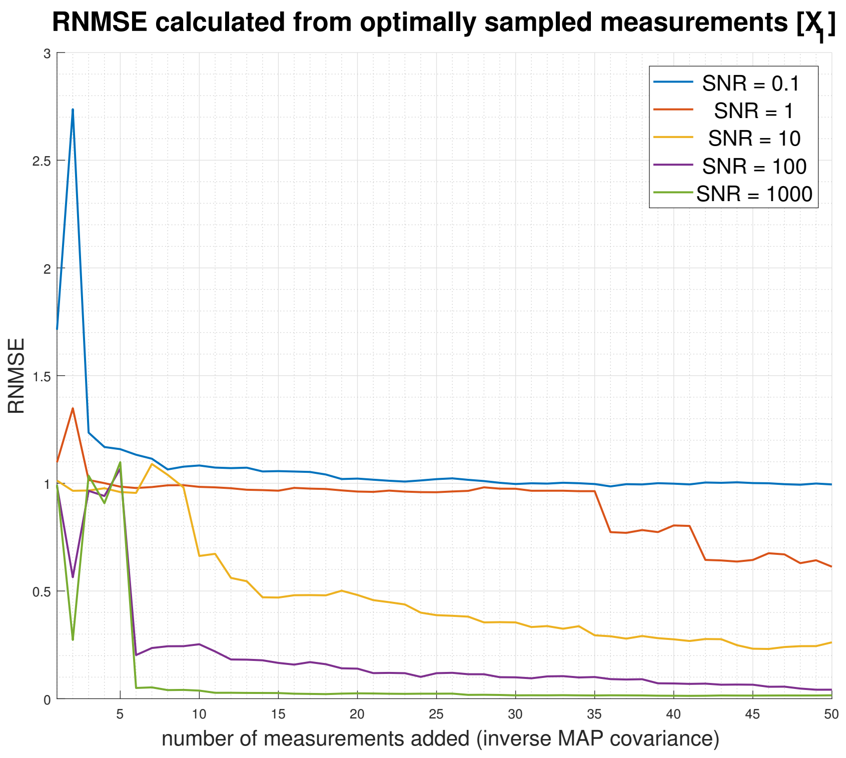

Figure 9: RNMSE results for Gene X_1 with different signal to noise ratios. Observe that an increased SNR leads to better reconstrution. The graphs exhibit cliff-like (see **Cliff and decreasing trends** decreases in reconstruction error. It seems that the place of these cliffs are closer to the origin when the signal to noise ratio is increased.

It is possible to observe two distinct regions on the graphs.

Early region with spikes

The first region is characterised by random jumps, i.e. a large increase in error (i.e. SNR = 0.1 on Figure 9) or large decrease (i.e. SNR = 1000 on Figure 9).

In this region the algorithm is attempting to fit a sparse model on few measurements points. Given noiseless samples, and a sparsity constraint, two measurements are theoretically enough to fit a model. The pit for SNR = 100 and SNR = 1000 indicates, that this is indeed the case, for SNR = 1000 an RMSE of 0.3 is shown on Figure 9.

Cliff and decreasing trends

The second region has either estimates improving in a slowly decreasing fashion or suddenly which we call **cliffs**. Cliffs might be related to the algorithm setting weights to 0, while the slowly decreasing trend might be related to getting increasingly better estimates of the weights.

Observe that the first curve on Figure 9, corresponding to SNR $= 0.01$ is higher than one, which indicates that the estimates are very bad quality because the zero model would overperform in this metric (see Appendix 6.2). It does not mean that the SBL is unable to reconstruct the system with this SNR value, more likely is that it needs substantially more measurements.

Last interesting thing to note on these graphs is that for all genes, the curves are very similar, and the jumps are also at similar places, which might indicate rule of thumbs for how many optimal measurements are needed for reconstruction.

Cliff lower bound

Knowing that the signal with SNR $= 1000$ is practically noiseless, it is interesting that the number of measurements needed for reconstruction are 4 in Figure 11, even though that the number of parameters needed to be identified are two. This indicates that there is a gap in achieving this theoretical lower bound which might be because the algorithm is an approximation to the L_0 penalty.

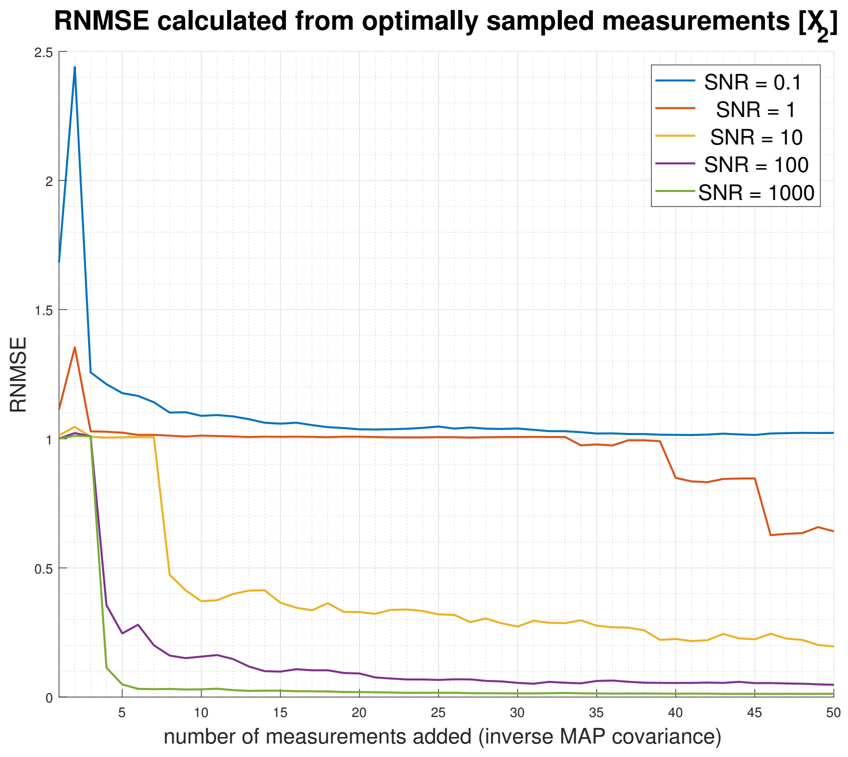

Figure 10: Observe that the cliff-like behaviours happen at similiar points as Figure 9, i.e. the SNR=1 curves decreaes at $\mathbf{y}_{41}$, while here it is $\mathbf{y}_{40}$.

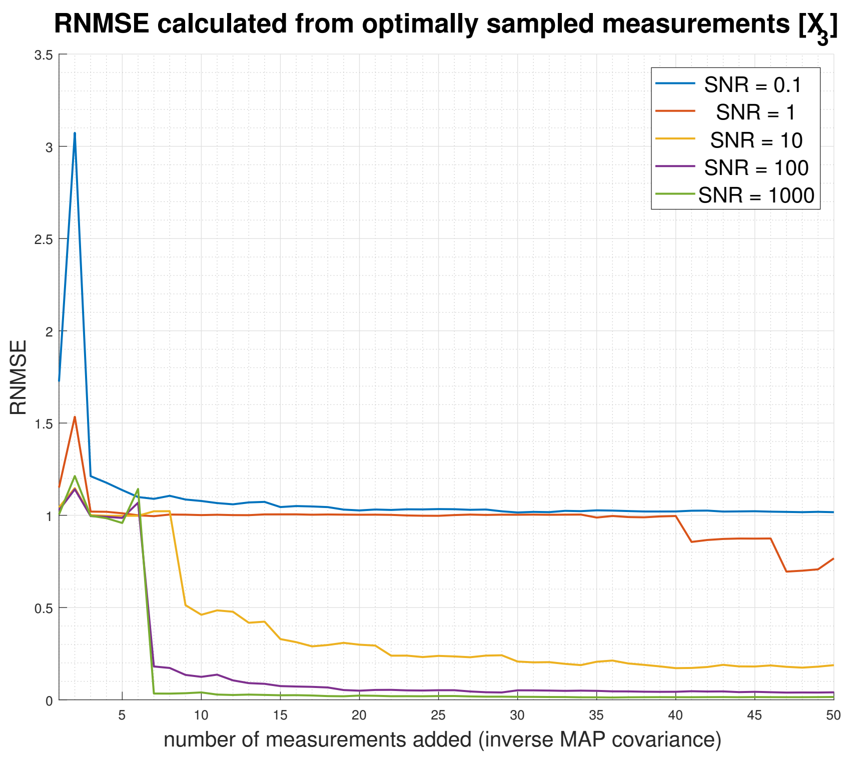

Figure 11: In agreement with Figure 9 and 10, similar error curve is obtained for all genes, with nearly the same points of decrease.

4.2 Evolution of weights in the dictionaries

On Figures 12-15, the weights for the dictionary functions are plotted as a function of measurements. As the reader goes down in this plot, more measurements are added and each column represents one dictionary function. The white colour indicates that those weights are 0. The values are the absolute values and logarithms of the actual weight values, which compresses the order of magnitudes for better visibility.

Higher signal to noise ratio results in sparser solution

Notice that the signal to noise ratio's effect can be observed here too. Effectively, a better signal to noise ratio enforces the dictionary to be sparser and prunes the unrelated weights more succesfully, which we can see graphically as the disapperance of the squares.

Cliff-like drops correspond to subset selection or deselection

Also, notice that the estimates are improving over measurements. For example on Figure 12, gene 3 prunes out the activator dictionary function at column 25. This pruning is also associated with the cliff-like drop of the RNMSE curve on Figure 11. This indicates that the cliff-like drops correspond to the selection of the right descriptors, or deselection of the wrong descriptors from the dictionary.

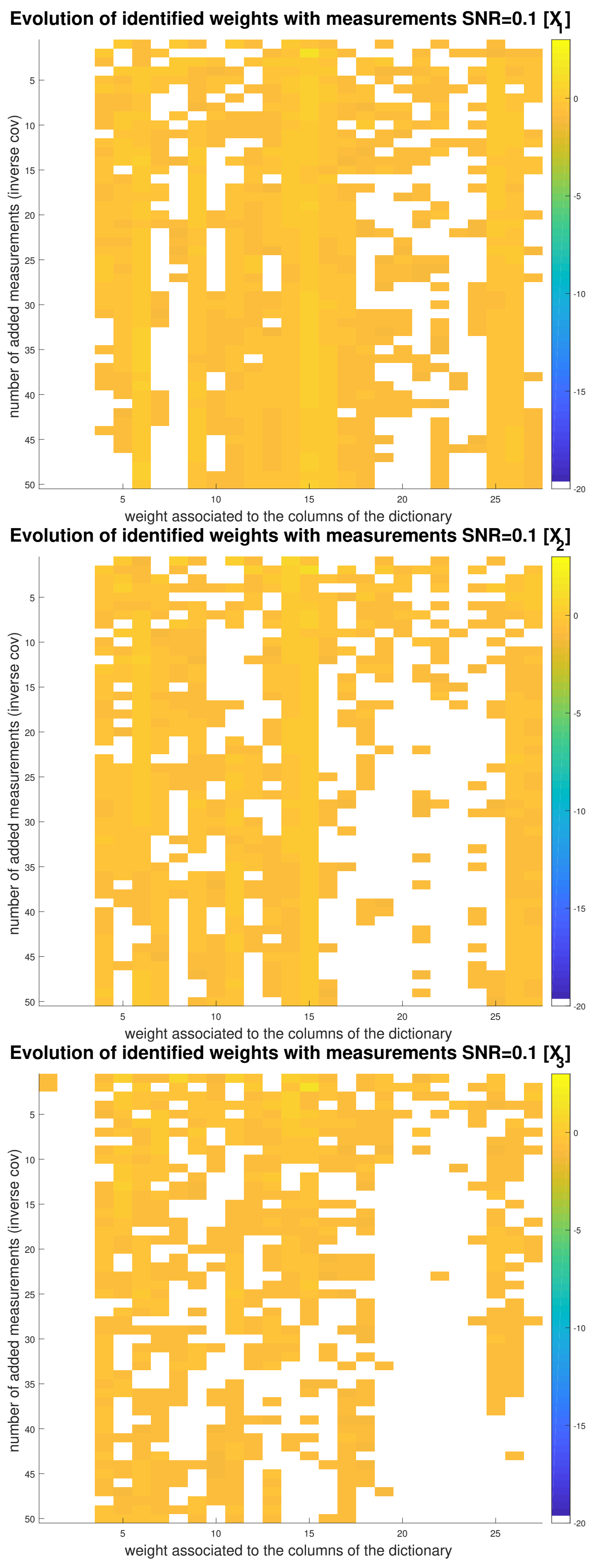

Figure 12: Dictionary of weights with SNR=0.1

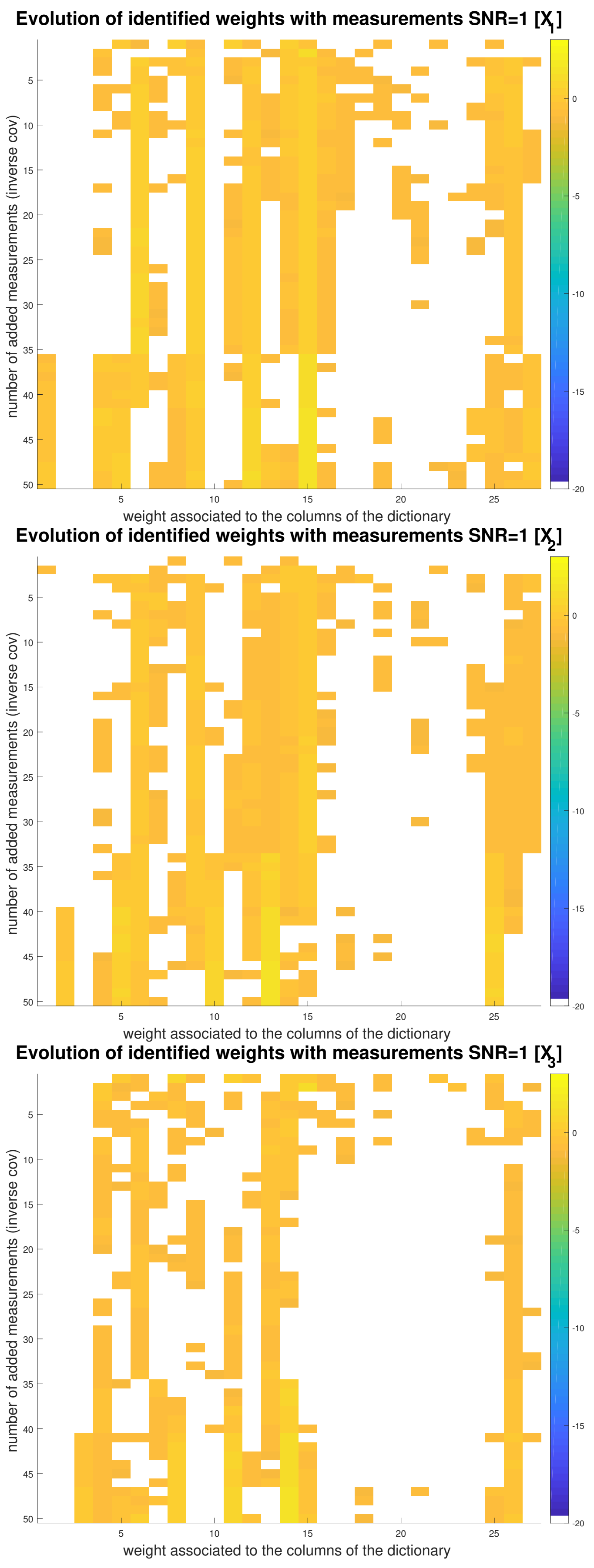

Figure 13: Dictionary of weights with SNR=1

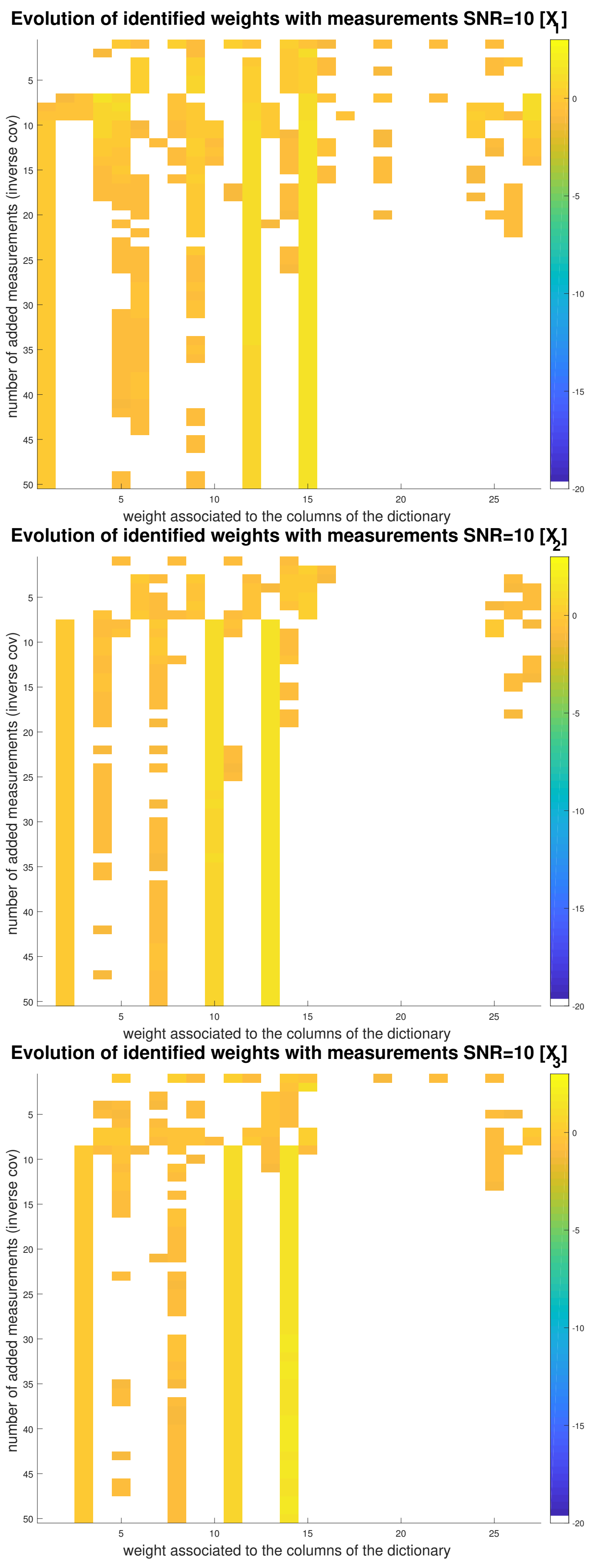

Figure 14: Dictionary of weights with SNR=10

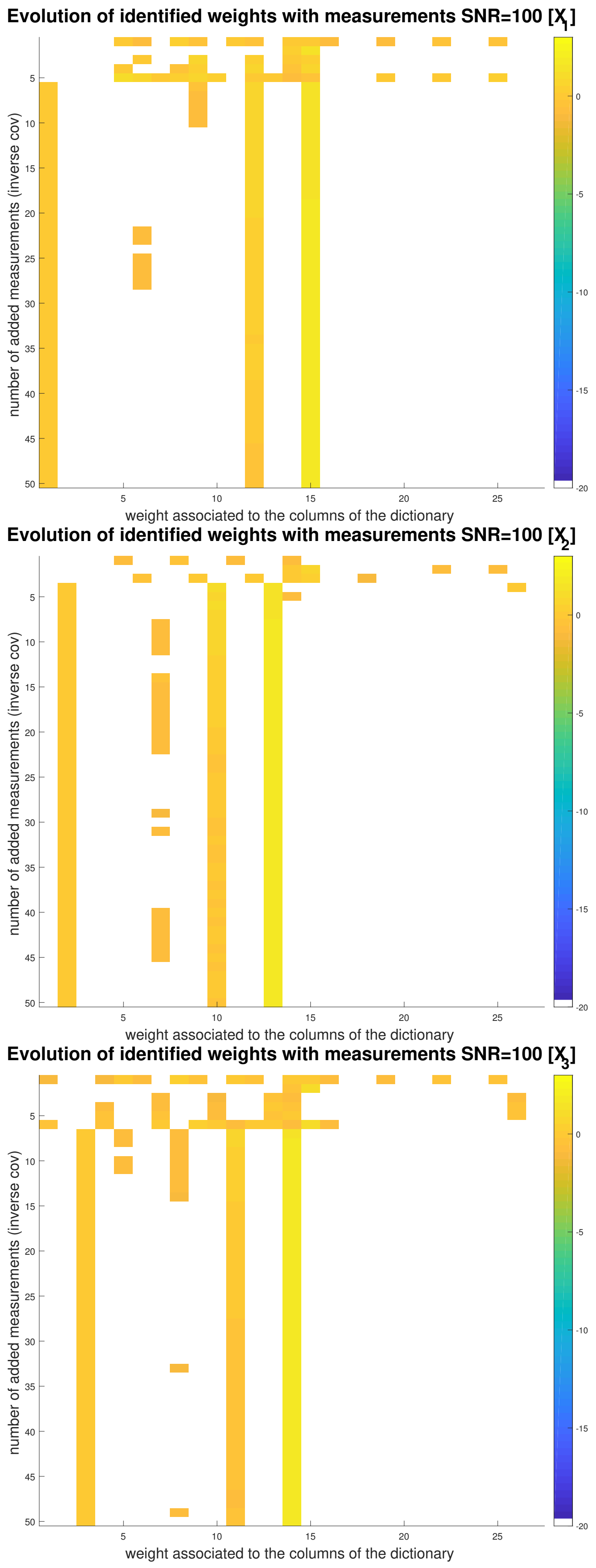

Figure 15: Dictionary of weights with SNR=100

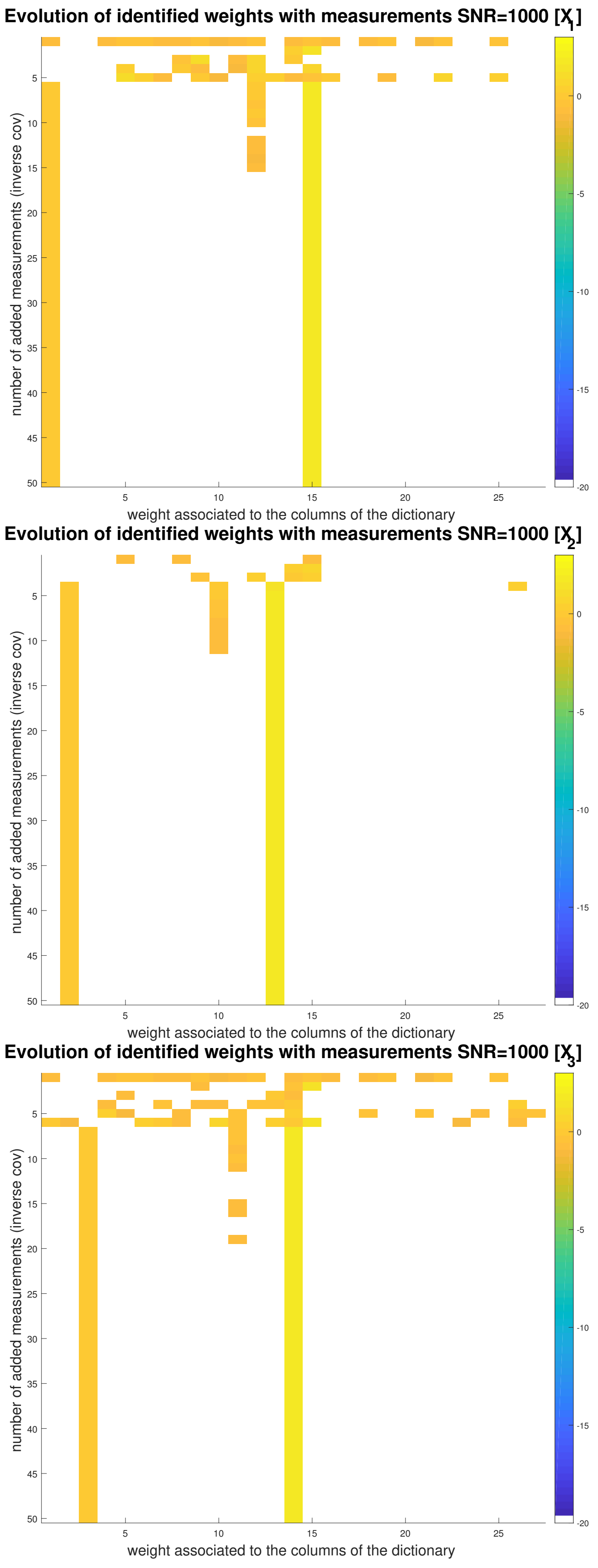

Figure 16: Dictionary of weights with SNR=100

4.3 Oracle estimator comparison

The oracle estimator is of considerable interest because it tells us given the right subset of the dictionary what is the minimum variance estimate.

The SBL is able to leverage a priori noise information in estimation

It was expected that the Oracle estimator will give a lower bound, surprisingly, this is not always true. For example, on Figure 17, it is easy to see that this does not hold. This is because the SBL assigns values more cautiously, because it is not confident in the estimate, while the RNMSE just tries to minimise the variance on very noisy samples and adjusts the parameters to fit the noise.

However, it is important to note that an RNMSE larger than 1 indicates a very bad model (See Section 6.2). Being still close to 1 is achieved by the SBL, by using a myriad of small coefficients to fit the data.

Intersection of the SBL and Oracle curves might indicate an identifiability criterion

On Figure 18, notice the intersection of the oracle estimator's and the SBL algorithm's error curve. It seems to be that the RNMSE cannot get lower than 1 before this intersection. In this case the oracle doesn't behave as a lower bound, because it blindly fits noisy measurements. So what we can see here is rather the fact, that when the RNMSE is higher than one it is meaningless to compare the models.

Least Squares after subset selection by SBL might give better results sometimes

On Figure 19-21, the oracle estimator is a clear lower bound. Take a moment now to look at Figure 16 and 21, and observe that the SBL converged to the true subset. This indicates that when subset selection is successful, it might be advisable to use that subset in a Least Squares, because it will give better results.

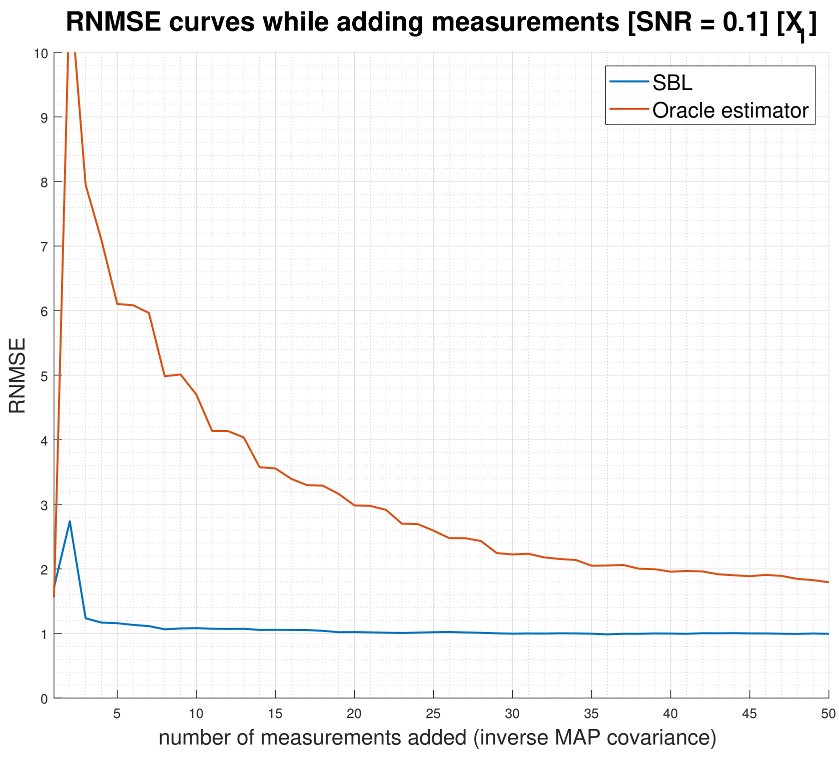

Figure 17: Oracle estimator comparison with SNR=0.1

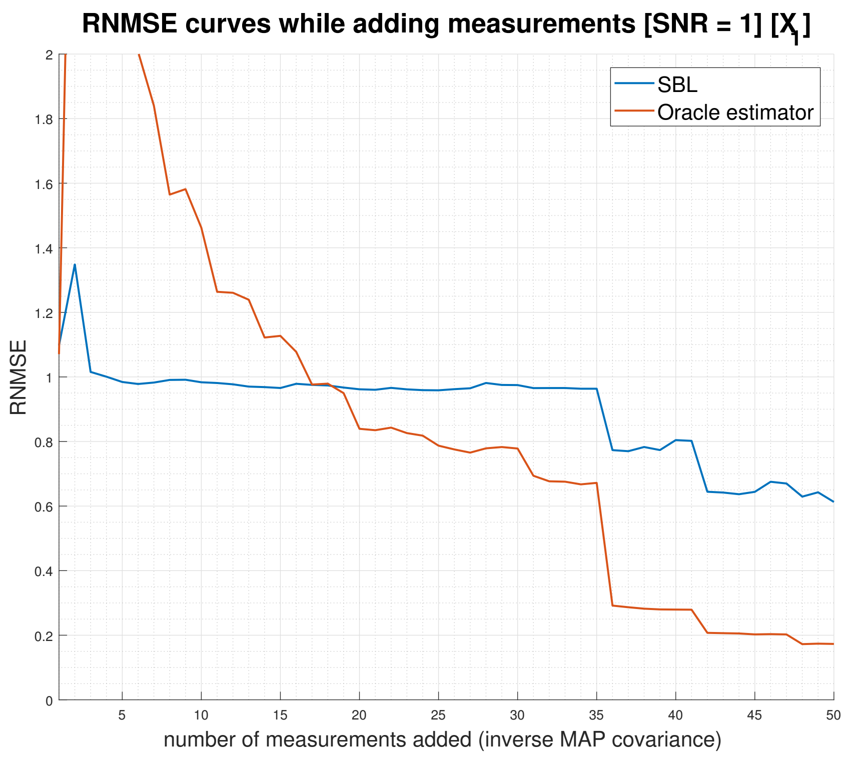

Figure 18: Oracle estimator comparison with SNR=1

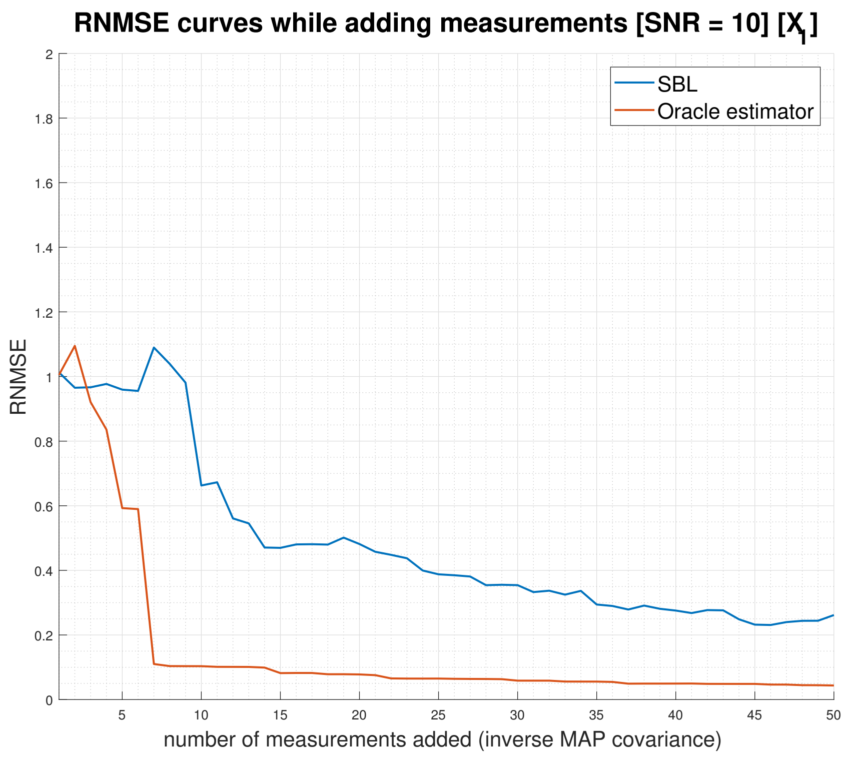

Figure 19: Oracle estimator comparison with SNR=10

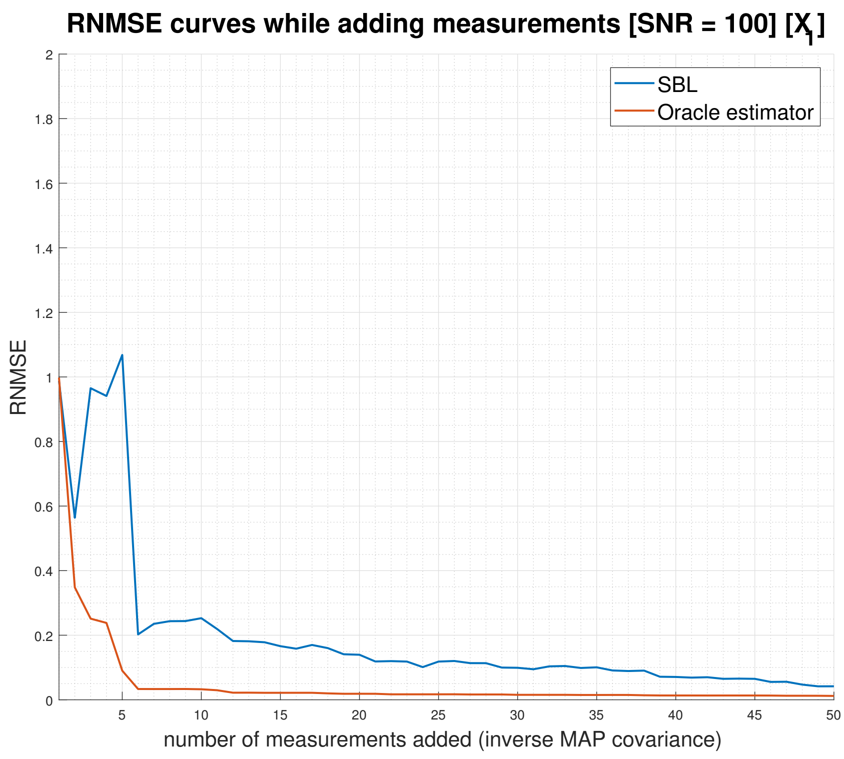

Figure 20: Oracle estimator comparison with SNR=100

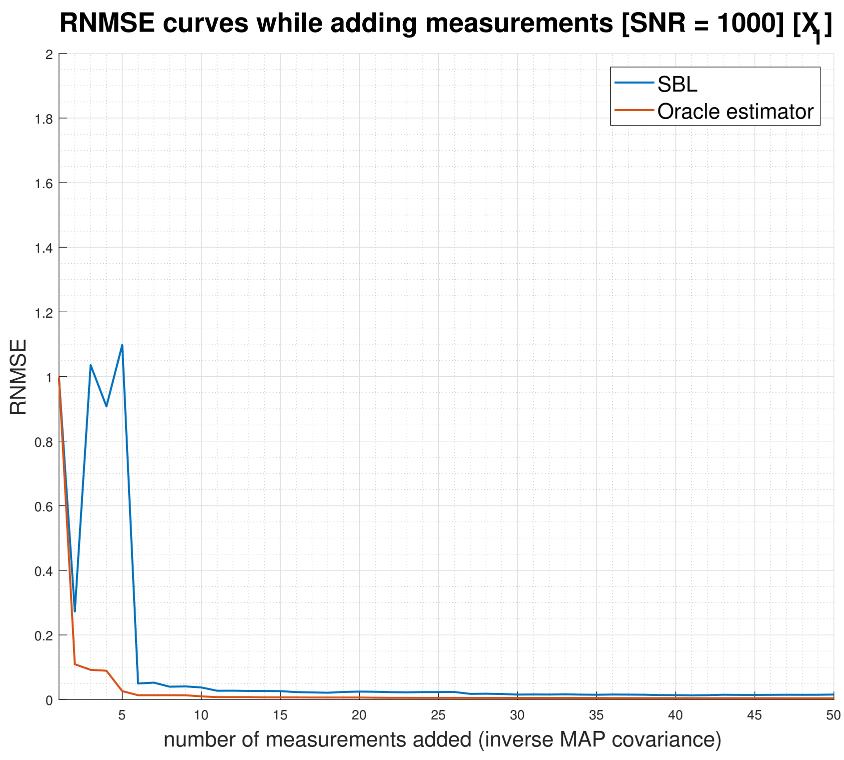

Figure 21: Oracle estimator comparison with SNR=1000E

4.4 Inverse MAP covariance values

On Figure 22, observe that the inverse covariance maximisation curves (see Section 3.3.3) indicate the usefulness of the data by having a substantial increase at high SNR, while only a moderate increase at low SNR. This is due to how we effectively regularise the dictionaries with adding $\lambda\mathbf{I}$, which diminishes the difference in the measurements. The curves on the top are the noisiest because the largest $\lambda\mathbf{I}$ are added to the noisiest curve.

One one hand, this is desired because the noisier the data, the more indifferent should the algorithm become about ordering samples based on measurements. On the other hand, with a smaller λ, we could have still ordered the samples, however such ordering would remain without interpretation.

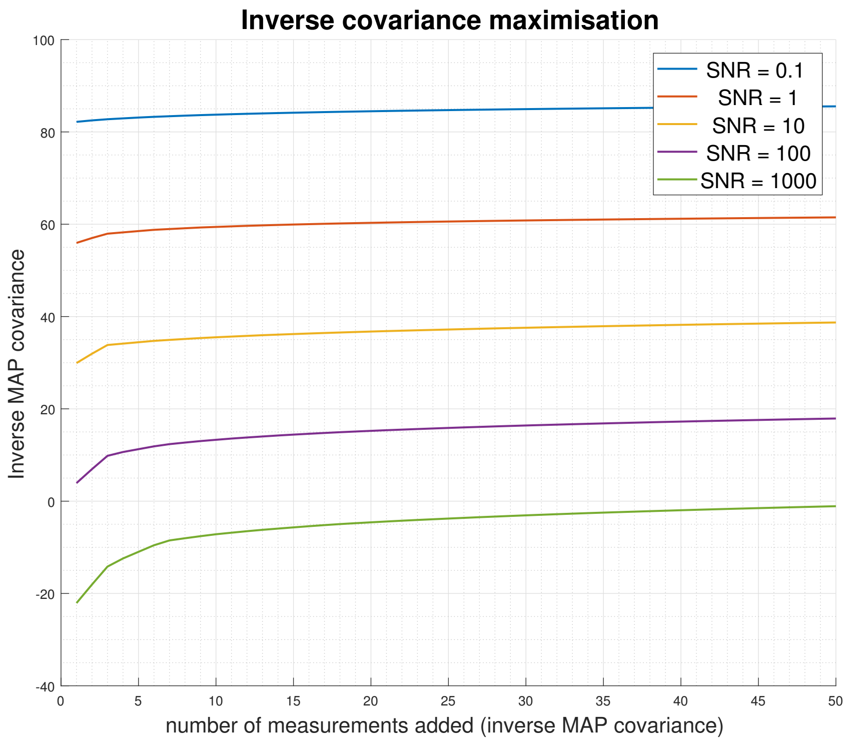

Figure 22: The logarithm of inverse MAP covariance plotted against the number of measurements. As the noise content increases the curves flatten.

5 Discussion

5.1 Guidelines for biologists using the toolbox

Based on these results it is possible to construct a set of guidelines for biologists using the toolbox.

When using the SBL algorithm, the biologist will get back a subset selection from the algorithm. That subset selection is the most useful part of this algorithm. Based on the Oracle estimators, it is recommended to run a Least Squares algorithm in the subsets of weights, because it might give back better results. This is only guaranteed if the SBL algorithm correctly identifies the subset of dictionary functions used.

The other conclusion from the results that the algorithm usually gets substantially better after some number of points are sampled, due to this cliff behaviour. We recommend using at least 10 optimally values for $\text{SNR} = 100, 1000$ and using at least 50 values for lower SNRs.

We are currently working on calculating the positions of these cliffs at different signal to noise ratios to obtain some rule of thumb values.

5.2 Future work: Estimation of derivatives, nonlinear behaviour of noise

A key assumption that we have made using this model is that the derivatives can be estimated to a satisfactory extent. There are signal processing techniques discussed in the PhD thesis of Dr. Pan [5] to achieve this, but it is also part of the work to be done with this toolbox.

Another matter that will need additional considerations is **measurement noise**. In Definition 7, $\mathbf{x}$ was considered unaffected by noise, however this is not the case. The dictionary $\mathbf{\Phi}$ has to be considered as a mapping of the measurements corrupted with white Gaussian noise η.

$$\mathbf{\Phi} = \begin{bmatrix} f_1(x_1(t_0)+\eta) & f_1(x_2(t_0)+\eta) & \dots & f_1(x_m(t_0)+\eta) & \dots & f_b(x_m(t_0)+\eta) \\ f_1(x_1(t_1)+\eta) & f_1(x_2(t_1)+\eta) & \dots & f_1(x_m(t_1)+\eta) & \dots & f_b(x_m(t_1)+\eta) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f_1(x_1(t_{k-1})+\eta) & f_1(x_2(t_{k-1})+\eta) & \dots & f_1(x_m(t_{k-1})+\eta) & \dots & f_b(x_m(t_{k-1})+\eta) \end{bmatrix} \in \mathbb{R}^{k\times l} \tag{44}$$

A possible solution would be to use the SBL algorithm to first clean the signal using an orthogonal basis. Succesful use of the algorithm for cleaning signals from MEG was already demonstrated [22].

References

[1] J. T. L. Wang, "Inferring Gene Regulatory Networks: Challenges and Opportunities," *Journal of Data Mining in Genomics & Proteomics*, 2015.

[2] D. Marbach, J. C. Costello, R. Küffner, N. M. Vega, R. J. Prill, D. M. Camacho, K. R. Allison, M. Kellis, J. J. Collins, A. Aderhold, G. Stolovitzky, R. Bonneau, Y. Chen, F. Cordero, M. Crane, F. Dondelinger, M. Drton, R. Esposito, R. Foygel, A. De La Fuente, J. Gertheiss, P. Geurts, A. Greenfield, M. Grzegorczyk, A. C. Haury, B. Holmes, T. Hothorn, D. Husmeier, V. A. Huynh-Thu, A. Irrthum, G. Karlebach, S. Lèbre, V. De Leo, A. Madar, S. Mani, F. Mordelet, H. Ostrer, Z. Ouyang, R. Pandya, T. Petri, A. Pinna, C. S. Poultney, S. Rezny, H. J. Ruskin, Y. Saeys, R. Shamir, A. Sîrbu, M. Song, N. Soranzo, A. Statnikov, N. Vega, P. Vera-Licona, J. P. Vert, A. Visconti, H. Wang, L. Wehenkel, L. Windhager, Y. Zhang, and R. Zimmer, "Wisdom of crowds for robust gene network inference," *Nature Methods*, vol. 9, no. 8, pp. 796–804, 2012.

[3] K. D. Dahlquist, B. G. Fitzpatrick, E. T. Camacho, S. D. Entzminger, and N. C. Wanner, "Parameter Estimation for Gene Regulatory Networks from Microarray Data: Cold Shock Response in Saccharomyces cerevisiae," *Bulletin of Mathematical Biology*, vol. 77, pp. 1457–1492, aug 2015.

[4] D. Wipf and S. Nagarajan, "A New View of Automatic Relevance Determination," *Compute*, 2008.

[5] W. Pan, *Bayesian Learning for Nonlinear System Identification.* PhD thesis, 2016.

[6] M. Drakakis, "Biomedical Instrumentation Op-Amp Filters,"

[7] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning," *Springer New York USA HuberW*, p. 37, 2009.

[8] G.-B. Stan, "Modelling in Biology (Lecture Notes)," 2018.

[9] R. M. Murray and D. Vecchio, "Biomolecular Feedback Systems," *Bernoulli*, 2010.

[10] M. B. Elowitz and S. Leibier, "A synthetic oscillatory network of transcriptional regulators," *Nature*, 2000.

[11] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, 1970.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning*, vol. 4. 2006.

[13] S. Boyd and L. Vandenberghe, *Convex Optimization.* 2004.

[14] D. J. C. MacKay, "Bayesian Interpolation," *Neural Computation*, 1992.

[15] D. Wipf and S. Nagarajan, "Iterative reweighted 1and 2methods for finding sparse solutions," *IEEE Journal on Selected Topics in Signal Processing*, 2010.

[16] Z. Tuza and G.-B. Stan, "Characterization of Biologically Relevant Network Structures from Time Series Data (Unpublished work)," *Characterization of Biologically Relevant Network Structures form Time Series Data.*

[17] J. Palmer, D. Wipf, K. Kreutz-Delgado, and B. Rao, "Variational EM algorithms for non-Gaussian latent variable models," *Advances in Neural Information Processing Systems*, 2006.

[18] D. Wipf, J. Palmer, B. Rao, and K. Kreutz-Delgado, "Performance evaluation of latent variable models with sparse priors," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2007.

[19] M. Seeger and D. Wipf, “Variational Bayesian Inference Techniques,” *Signal Processing Magazine, IEEE*, 2010.

[20] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, “Introduction to variational methods for graphical models,” *Machine Learning*, 1999.

[21] L. Ljung, “Ljung L System Identification Theory for User,” 1987.

[22] D. P. Wipf, B. D. Rao, and S. Nagarajan, “Latent variable Bayesian models for promoting sparsity,” *IEEE Transactions on Information Theory*, 2011.

[23] S. Haykin, *Adaptive Filter Theory.* 2002.

6 Appendix

6.1 Simulation Toolbox

During the timespan of the project, it was quickly realised that there is a need for flexible generation of GRNs.

For this purpose, part of the code created in MATLAB was aimed at writing a library which is capable of generating GRN with any order of Hill functions and perform numerical integration of networks in Section 3.1.4.

This automated framework is capable of:

- defining a GRN with n genes
- defining repression, degradation, activation kind of interactions between two genes
- visualising graphs of these simple network, so the user can be confident that he constructed the desired GRN
- solving these for any desired timespan using either Runge-Kutta or Euler solvers

The following section will contain an introduction to these and hopefully, by the end of this section, the reader will be encouraged to use this framework to prototype GRNs.

Below, you see an example code for constructing an incoherent feedforward loop. The user has to prespecify the number of genes in the GRN by the function `geneGraph(geneNumber)`. An interaction can be specified by `sim.activation(activatorGene,activatedGene,transcriptionRate, hillOrder`.

Listing 1: Example of constructing an incoherent feedforward loop

```
% Incoherent feed-forward loop

sim = geneGraph(3);

% Constructing basic interactions
sim = sim.activation(1, 2, 0.4, 4);
sim = sim.activation(2, 3, 0.4, 4);
sim = sim.repression(1, 3, 0.4, 4);

% Addding degradation terms
sim = sim.degradation(1, 0.1);
sim = sim.degradation(2, 0.1);
sim = sim.degradation(3, 0.1);

% Showing gene graph
sim.showGraph();
```

This is an important tool in our simulation framework because it greatly simplifies creating GRNs. With many nodes, these can be tedious to follow, for this reason, `sim.showGraph()` shows the graph below on Figure 6.1. This function is good for prototyping GRNs.

An additional nice feature of this framework is that it can be used very elegantly with conventional programming techniques to provide rapid generation of designs with certain patterns. Consider the following code for generation of an n gene repressilator.

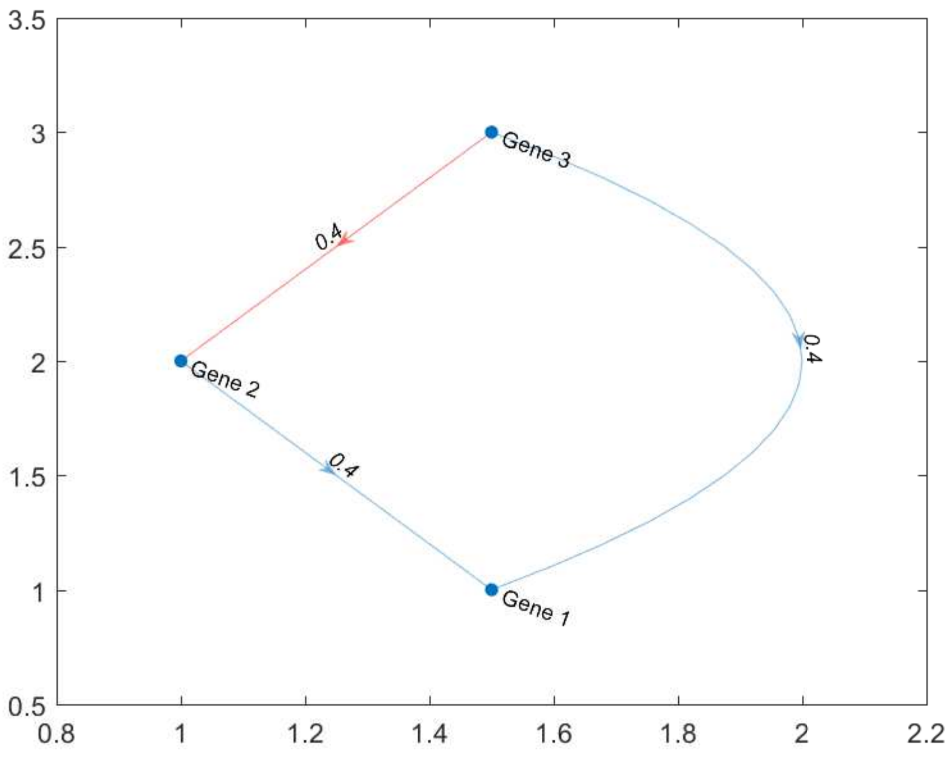

Figure 23: The incoherent feedforward loop generated by `sim.showGraph()`

Listing 2: An example of constructing an n gene repressilator

```
% Repressilator

% Housekeeping
clc;
clear;
close all;

node = 5;
sim = geneGraph(node);

k = 40;
d = -1;
sim = sim.repression(1, node, k, 4);
sim = sim.degradation(1,d);

for i=2:node
    sim = sim.repression(i, i-1, k, 4);
    sim = sim.degradation(i,d);
end



% Let's simulate the gene regulatory network
initialConditions = 1:1:node;
simulationInterval = 0:0.01:100;
```

```
processNoise = 0.01;

[derivativeSeries, ...,
    timeSeries] = ...,
    sim.runRungeKutta(initialConditions, processNoise, simulationInterval
        );

figure;
plot(simulationInterval, timeSeries, 'LineWidth', 1.5)
run('figureFormatter');
```

This toolbox can then be used to draw conclusions by varying some parameters while keeping the topology of the GRNs exact same. The most interesting part, of course, is that given some initial conditions, the library will solve you the ODEs' of these GRNs without writing the actual code for it.

6.2 Characterisation experiments with the RNMSE

The RNMSE metric quantifies the error between the ground truth and the estimated weights.

A few properties of this metric can be demonstrated mathematically.

Everything higher than RMSE of 1 is useless Recall the definition of RNMSE,

$$\mathrm{RNMSE} = \frac{||\mathbf{w} - \hat{\mathbf{w}}||_2}{||\mathbf{w}||_2}. \tag{45}$$

Consider the case when $\hat{\mathbf{w}} = [0, 0, \ldots, 0]^T$, by substitution it is easy to see that

$$\mathrm{RNMSE} = \frac{||\mathbf{w}||_2}{||\mathbf{w}||_2} = 1. \tag{46}$$

Meaning that if all weight estimates are zero then the RNMSE is 1.

6.3 Remarks on the similarities of Ridge regression and SBL

Consider the linear model,

$$\mathbf{y} = \mathbf{\Phi}\mathbf{w} + \epsilon, \tag{47}$$

where ϵ is $\mathcal{N}(0, \Lambda\mathbf{I})$. Consider the calculation of the maximum *a posteriori* distribution $p(\mathbf{w}|\mathbf{y})$. Using the Bayes theorem first, we obtain

$$p(\mathbf{w}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{y})}. \tag{48}$$

The maximum a posteriori objective can be written as $\mathbf{w}$ being the dependent variable, and simplified because $p(\mathbf{y})$ is independent of the dependent variable,

$$\arg\max_{\mathbf{w}} p(\mathbf{y}|\mathbf{w})p(\mathbf{w}) \tag{49}$$

In convex optimisation, minimisation problems are preferred, and applying the logarithm won't change the place of minimas and maximas. Using this, we obtain

$$\arg\min_{\mathbf{w}} - \ln p(\mathbf{y}|\mathbf{w}) \tag{50}$$

Now, we assume that the prior is distributed with a fixed $\mathcal{N}(0, \mathbf{\Gamma})$, where $\mathbf{\Gamma}$ is **fixed**.

$$\arg\min_{\mathbf{w}}(\mathbf{y} - \mathbf{\Phi w})^T(\Lambda \mathbf{I})^{-1}(\mathbf{y} - \mathbf{\Phi w}) + \mathbf{w}(\mathbf{\Gamma})^{-1}\mathbf{w} \tag{51}$$

It is easy to see that λ in the MAP optimisation is the combination of the variance of the prior and the variance of the measurements. Thus, if there is a formula for finding the variance of the prior, the measurement noise can be used to tune λ.

In Ridge regression, the optimisation is performed to maximise the posterior value for a fixed variance $\mathbf{\Gamma}$. In deriving the solution of the MAP problem, thus the denominator is a constant so the maximum value need not be computed.

This means that the following equation in the Ridge regression framework would not make sense, as Γ is fixed.

$$\hat{\gamma} = \arg\max_{\gamma} \zeta(\gamma_j) \int p(\mathbf{y}|\mathbf{w})\mathcal{N}(0, \mathbf{\Gamma})\mathbf{dw} \tag{52}$$

In the process of Bayesian interpolation [14], this term is forced to make sense by considering the variance Γ variable. In the maximum *a posteriori* framework, thus the maximisation of the evidence is simply the maximum a posteriori estimate itself, as in

$$p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{w})p(\mathbf{w})\mathbf{dw} \tag{53}$$

only the MAP estimate can be maximised as the prior is fixed.

Thus, when considering $\mathbf{\Gamma}$ variable and also a subject to optimisation, the reweighting of the Ridge coefficients appear in the cost function, because it is the minimisation of the evidence function subject to this new variable, which can be decoupled to solving the MAP problem and solving for the $\mathbf{\Gamma}$ dependent terms,

$$\mathcal{L}(\gamma, \mathbf{w}) = \min_{\mathbf{w}} \frac{1}{\lambda}||\mathbf{y} - \mathbf{\Phi w}||_2^2 + \sum_{i=1}^{n} \frac{w_i^2}{\gamma_i}. \tag{54}$$