Technische Universität München

Fakultät für Mathematik

HelmholtzZentrum münchen
Deutsches Forschungszentrum für Gesundheit und Umwelt

ICB Institute of Computational Biology

From deterministic Boolean networks to stochastic continuous models

Master Thesis

by

Sebastian Winkler

Technische Universität München (TUM)

Department of Mathematics

M.sc Mathematics in Bioscience

Supervisor: Prof. Dr. Dr. Fabian Theis

Advisor: Dr. Christiane Fuchs

Submission date: 28. August 2014

Technische Universität München

Fakultät für Mathematik

Titel der Masterarbeit, englisch:

From discrete Boolean networks to stochastic continuous models

Titel der Masterarbeit, deutsch:

Von diskreten Boolschen Netzwerken zu stochastischen stetigen Modellen

Verfasser: **Sebastian Winkler**
(Matrikelnummer 03608811)

Aufgabensteller: Prof. Dr. Dr. Fabian Theis
Betreuerin: Dr. Christiane Fuchs

Abgabedatum: 28.08.2014

Ich erkläre hiermit, dass ich die vorliegende Masterarbeit selbstständig und nur mit den angegebenen Hilfsmitteln angefertigt habe.

München, den 28. August 2014, _____

Sebastian Winkler

Danksagung

Ich möchte folgenden Personen, Lebewesen und/oder sonstigen Entitäten
meinen Dank aussprechen:

1. Meiner Familie und auch allen anderen, insbesondere aber meinen Hunden

   Schnitzlon von Schnitzly (R.I.P.)
   und
   Zwirni (Zwirniratzis) Zwirnodopoulos.

2. Frau Dr. Christiane Fuchs für die angenehme und hilfreiche Betreuung.

3. Herrn Prof. Dr. Dr. Theis und Frau Dr. Fuchs für die Möglichkeit diese Arbeit am ICB im Rahmen der dort vorliegenden Gesamtsituation verfassen zu können.

Abstract

The overall topic of this thesis is the relationship between various models for biochemical systems in general and for gene regulatory networks in particular. According to classical dipolar characteristics like continuity vs. discreteness of the state space, continuity vs. discreteness of time, spatial homogeneity vs. heterogeneity or determinism vs. stochasticity, different respectively suitable model classes can be applied in order to model systems which are thought to be best modeled with a particular modeling framework. Boolean network models are chosen as a point of departure for the exploration of the outlined issue and are thus covered in slightly more detail than other modeling approaches. In particular, part of the main part of the thesis addresses the question of parameter inference in specific Boolean models. Another focus lies on methods which extend and enrich the basic state-discrete, time-discrete, spatially homogeneous and non-stochastic framework of Boolean networks. Hence the title of the thesis: *From discrete Boolean networks to stochastic continuous models*.

While the first chapter provides some biological background and outlines basic graph-based modeling approaches, the second chapter deals with two classical modeling approaches: deterministic models based on ordinary differential equations on the one hand and stochastic chemical kinetics and its extensions and approximations on the other hand. The third chapter then contains a description of Boolean modeling including the indicated extensions with respect to the respective nature of states, time, space and stochasticity where a particular focus in laid on so called generalized kinetic models (GKL). Chapter 4 proposes a simple estimation procedure for GKL networks with exponentially distributed time delays. While these are mathematically convenient it is argued that exponential distributions are not particularly well-suited for the situations modeled with GKL networks and hence Weibull distributed time delays are motivated and implemented. Chapter 5 deals with models which incorporate discrete and continuous characteristics into a single modeling framework. These are generically called hybrid models. Chapter 5 especially considers stochastic hybrid models and explores a simple example of this model class by means of example.

Zusammenfassung

Die vorliegende Arbeit beschäftigt sich mit dem Zusammenhang zwischen verschiedenen Ansätzen zur Modellierung biochemischer Netzwerke im Allgemeinen und von Genregulationsnetzwerken im Besonderen. Modellierungsansätze für biochemische Netzwerke können gemäß klassischer dipolarer Merkmale wie etwa Stetigkeit vs. Diskretheit des Zustandsraumes, Stetigkeit vs. Diskretheit der Zeit, räumliche Homogenität vs. Heterogenität oder Determinismus vs. Stochastizität klassifiziert werden und haben jeweils spezifische Anwendungen abhängig von der angenommenen Adäquatheit eines jeweiligen Ansatzes für ein konkretes biologisches System. Boolsche Netzwerke bilden im skizzierten Gesamtzusammenhang einen weiteren Schwerpunkt der Arbeit und werden daher ein wenig ausführlicher behandelt als andere Ansätze. Ein Teil der Arbeit beschäftigt sich mit der Parameterschätzung in bestimmten Boolschen Modellen, während ein weiterer Teil sich detailliert mit verschiedenen Ansätzen befasst, die zum Ziel haben, die klassischerweise Zustands-diskreten, Zeit-diskreten, Raum-homogenen und deterministischen Boolschen Netzwerk Modelle entsprechend zu erweitern und idealerweise zu bereichern. Daher der Titel der Arbeit: Von diskreten Boolschen Netzwerken zu stochastischen stetigen Modellen.

Nachdem im ersten Kapitel einige biologische Hintergrundaspekte und grundlegende Graphen-basierte Modellierungsansätze besprochen wurden, behandelt Kapitel zwei die klassischen Ansätze der Modellierung mit gewöhnlichen Differentialgleichungen einerseits und jenen der stochastischen chemischen Kinetik andererseits, Erweiterungen und Annährungen zu letzterer eingeschlossen. Das dritte Kapitel schließlich behandelt Boolsche Netzwerke und die angedeuteten Erweiterungen hinsichtlich Zustandsraum, Charakter der Zeit, Räumlichkeit und Stochastizität. Einen speziellen Fokus innerhalb dieses Kapitels erfahren die sogenannten verallgemeinerten Modelle der logischen Kinetik (engl.: generalized kinetic logic, abgekürzt GKL). Daran anknüpfend, studiert Kapitel vier eine einfache Methode zur Parameterschätzung in GKL-Modellen mit exponential-verteilten Verzögerungszeiten. Dieser Ansatz ist zwar mathematisch bequem, jedoch erscheinen seine Annahmen für die gegebene zu modellierende Situation eines biochemischen Netzwerkes inadäquat und es wird demgemäß eine Modifikation vorgeschlagen und implementiert, welche letztlich durch Weibull-verteilte Verzögerungszeiten charakterisiert ist. Kapitel 5 untersucht anhand von eines speziellen Beispiels das Zusammenspiel von diskreten und stetigen Variablen im allgemeineren Gesamtzusammenhang der stochastischen Hybridmodelle.

Contents

Notation

The notation adopted is either standard mathematical notation or else (hopefully) defined properly. Some notational "bottlenecks" are nevertheless outlined here:

$\mathbb{N} := \{1, 2, 3, 4, 5, ...\}$ is the set of positive integers

$\mathbb{N}_0 := \mathbb{N} \cup \{0\}$

For $n \in \mathbb{N}$ define $[n] := \{1, 2, ..., n-1, n\}$

$\mathbb{R}_{\geq 0} := [0, \infty)$

$|A|$ denotes the cardinality of a set A

For sets $A_1, ..., A_n$, $n \in \mathbb{N}$: $\prod_{i=1}^{n} A_n := A_1 \times \cdots \times A_n$

1 Biological background: biochemical networks

Generally speaking a biochemical "network" can be understood as a collection of biochemical species (different kinds of molecules) which through their respective interactions (chemical reactions, for example) form a (in general) very complex system. As a first intuitive approach one indentifies chemical species with the nodes (vertices) of a network (mathematically, a graph, see Subsection 1.5.1) and the (so far unspecified) interactions between these species as connections (arcs, edges) between these nodes. This (up to now) highly qualitative and even imprecise model is intuitively very pleasing and frequently used throughout all of molecular biology [Bornholdt 2008].

One can identify certain types of biochemical networks according to the types of molecules involved, the type of molecular interactions taking place, the function which is accomplished by the network or the time-scales on which the network acts and so on. Because of these differences the subdivision of different kinds of biochemical networks makes, as a first approximation, conceptual sense and is usually adopted throughout the literature. But it leaves of course open the question what kind of problems could arise when one tries to combine models for different sorts of networks what would certainly be asked for if the ultimate goal of Systems Biology ([Klipp et al. 2009], [Alon 2007], [Walhout et al. (ed.) 2013]) namely the understanding of whole organisms as a correspondence to the interactions of their parts, is to be accomplished in the future. See Section 1.4 for some illustrations.

One very important category of a biochemical network is given by gene regulatory networks (GRNs) which are networks where the nodes are genes or proteins (transcription factors) encoded by their respective genes which in turn have some influence on the expression of other genes whose gene products again may influence some other genes and so on and so forth [Alon 2007], [De Jong 2002]. GRNs are the main focus of the thesis and a more detailed description and some examples of GRNs are given in Section 1.3 and Subsection 1.5.1.

First I will take a brief and superficial look, based on characteristic examples, at two further important sorts of biochemical networks, namely signal transduction networks (or signalling networks) in section 1.1 and metabolic networks in section 1.2.

Finally section 1.4 shortly discusses the crucial interplay of the discussed three main types of biochemical interaction networks in living cells by means of example and section 1.5 formally introduces 'the' graph representation of biochemical networks and makes some general remarks on different modelling approaches in systems biology.

1.1 Signal transduction networks

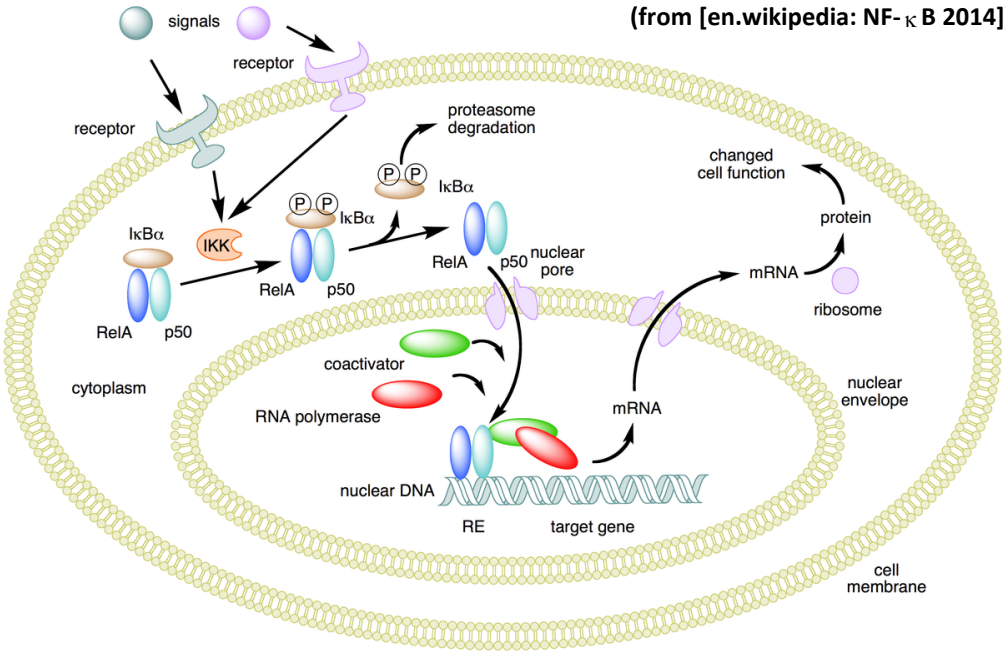
Signalling networks do not work via the direct alteration of gene expression (as GRNs, see section 1.3), but instead only rely on successive minor modifications (phosphorylation, methylation, etc.) to some existing molecules or multimerizations depending on some input

signal (usually the presence or absence of some molecules registered or not registered by some kind of molecular receptor)[Purves et al. 2006: chapter 15], [Klipp et al. 2009: chapter 3.2]. Signalling networks, since the elaborate and relatively time-consuming process of gene expression is not directly involved, usually act on much faster timescales than GRNs. Of course, in real biological systems Signalling and Gene Regulatory networks are highly interwoven such that it is often necessary to give up the to some extent (though not totally) artificial separation between signalling networks and GRNs. One typical situation usually involves some signalling network sensing some alteration in the environment which is then transmitted through the network to alter some GRN in order to change gene expression.

The NF-κB signalling pathway

One simple paradigmatic example of a signalling pathway is the so called NF-κB pathway which is involved in 'stress response' ('stress': ultraviolet radiation, bacterial or viral antigens etc., 'stress response': inflammation, proliferation of immune cells, etc.) in many animals [en.wikipedia: NF-κB; 2014]. By example of a (tiny) part of this pathway depicted in figure 1.1 below we try to exemplify the signalling paradigm. NF-κB stands for 'nuclear factor kappa-light-chain-enhancer of activated B cells' and is the shortcut for a protein complex which in the figure below is constituted as a dimer between a protein called RelA and another protein called p50. In addition some further protein, called IκBα is bound to that dimer and under normal, i.e. stress-free conditions prevents the complex to assert a certain action on some specific region of the DNA in the cell nucleus. Now, if some 'stress factors' are sensed by the receptors in charge, some modifications of signalling molecules (not shown) take place such that ultimately a protein called IKK gets 'activated' (via some chemical modification). Activated IKK then phosphorylates the protein IκBα which is bound to the RelA-p50 complex.

Fig. 1.1. Conceptual Logic of the NF-κB pathway
(from [en.wikipedia: NF-κB 2014])



This phosphorylation leads to the dissociation of IκBα and the complex and while IκBα is degraded (by a machinery called proteasome) the RelA-p50 complex is transferred to the cell nucleus where it binds to some very specific region of the DNA and somehow 'recruits' further proteins (RNA polymerase for example) which are necessary for the transcription of so called target genes. This eventually leads to the alteration of gene expression in response to the stress signal.

I should remark that the just described conceptual example underscores the complexity of the whole NF-κB signalling pathway which involves many ways of stimulation (i.e. activation by certain inputs), different context-dependent ways of reacting to these inputs, various output possibilities via NF-κB variants and finally the interaction with other signalling pathways. A more complete picture of the NF-κB signalling pathway can be anticipated in the figure below (figure 1.2).

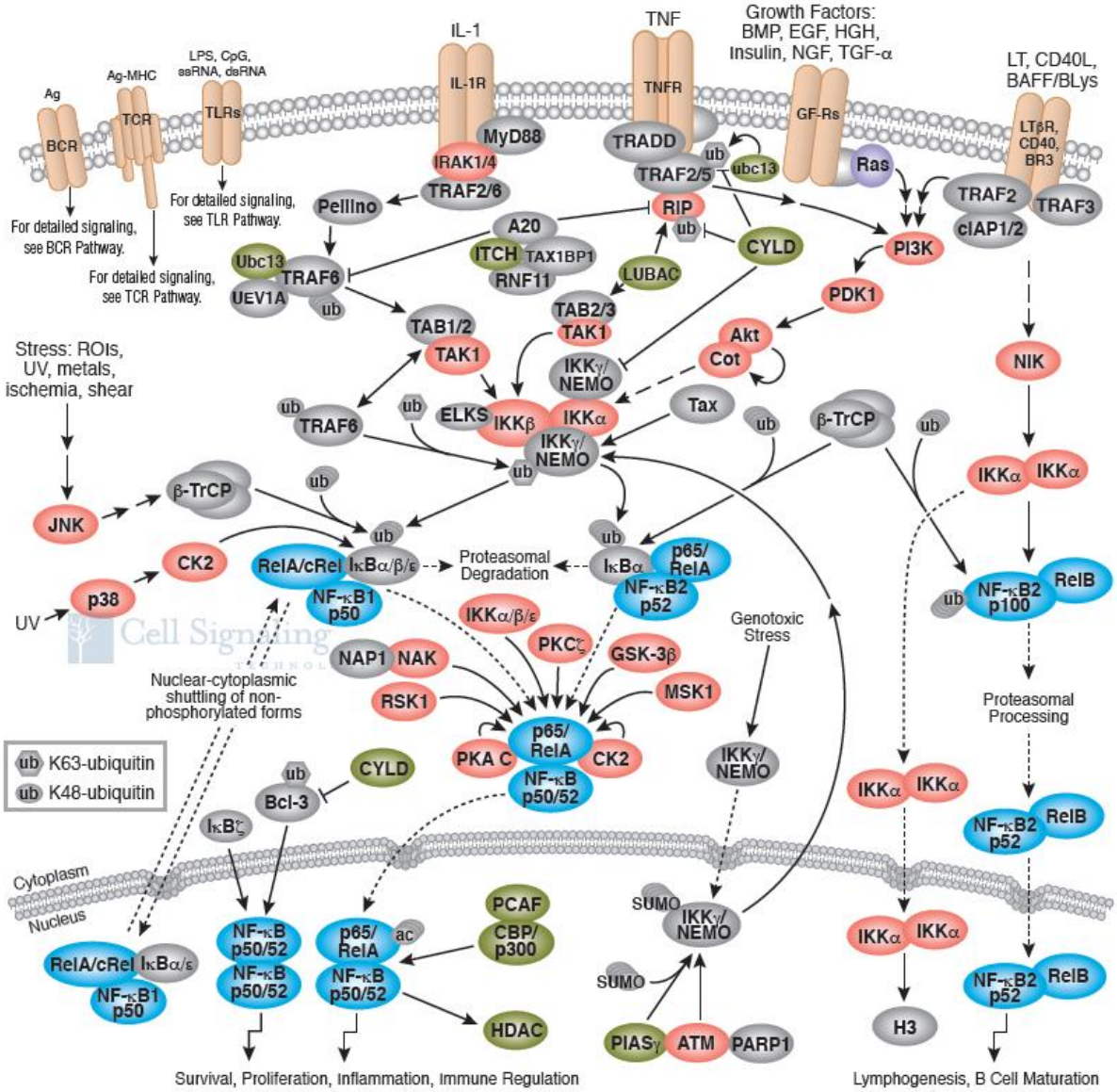


Fig. 1.2. Detailed NF-κB pathway **(from [CellSignalingTechnology NF-κB 2014])**

In figure 1.2 above one can see that there are several trans-membrane receptors, for example IL-1R or TNFR, which respond to specific signalling molecules like IL-1 or TNF and induce a downstream cascade of signalling events which ultimately culminate in the activation of the NFκB-complex which is then transferred to the cell nucleus where it positively influences genes involved in stress response. One can also see that there are other signalling pathways (upper left) which also feed into the NFκB pathway and in contrast to the external signals like TNF there can also be cell-internal signals like JNK on the middle left for example which responds to stress like toxic metals or UV radiation and then triggers the NFκB pathway via its specific route of activation. The exact legend of the scheme in figure 1.2 and further explanations can be looked up at [CellSignalingTechnology NFκB 2014], i.e. the exact meaning of the different colours of the diverse chemical species (indicating a certain type of molecule, like protein kinases which phosphorylate other proteins for example) or the meaning of the different interaction arrows. In general [CellSignalingTechnology 2014] is a place to find good depictions of signalling pathways. More detailed and annotated data on biochemical networks (and hence potentially more useful data) can of course be found via the various (often non-commercial) databases available on the internet [en.wikipedia: List of BioDatabases 2014] like for example the *Reactome database* for human signalling pathways [Joshi-Tope et al. 2004].

Circadian cycles in cyanobacteria

In the example of NFκB signalling the involved signal pathway(s) finally leads to a change in gene expression. There are however signalling processes which do not depend on some alteration of gene expression for their function in the sense that the output of the signalling pathway invariably induces some change in gene expression which then ultimately can bring about the necessary changes in order to cope with the environmental or cell-internal conditions sensed as the input to the signalling pathway. Instead, some signalling pathways just use the logic of modifications of some target molecules which then bring about the desired adaptation to the arriving signals. For example there is evidence that circadian cycles of certain cyanobacteria function only through the use of signalling modifications (phosphor-rylation in this case) but without the use of some process-specific change of gene expression [Nakajima et al.; 2005], [Dong, Golden; 2008]. (Of course, the involved proteins have to be produced at some point via gene expression, but the adaptations brought about by the signalling pathway only rely on the existing proteins. Maybe there are differences in the functionality (reaction time or response strength for example) of the pathway depending on the state of gene expression of the involved proteins but the adaptation to the environmental signal itself does not rely on some sort of alteration in gene expression.)

For a review on the systems biology of signal transduction pathways one can consult [Klipp et al. 2009: chapter 3.2] and [Mariottini, Iyengar 2013]. For a more experimentally oriented review on the systems biology of signalling pathways I refer to [Kulkarni, Perrimon 2013].

1.2 Metabolic pathways

The second types of biochemical network I want to discuss are metabolic networks. They are responsible for the process by which organisms utilize chemicals (nutrients) to gain energy and useful molecules which are again used by biosynthetic metabolic pathways to build up cell- and organism-specific bio molecules. Metabolic pathways mainly consist in linear, cyclic and branched cascades of successive enzyme-catalyzed reactions which constitute the to some extent already iconic pictures of biochemical pathways [Michal, Schomburg 2012].

Glycolysis

To get a feeling for the flavour of a typical metabolic pathway I provide a highly superficial look at one of the most important ones in life: glycolysis [en.wikipedia: Glycolysis; 2014], [Munk et al. 2008: chapter 7]. Glycolysis is highly conserved throughout the kingdom of life and almost every organism uses the same pathway for glucose utilization which indicates that the pathway is one of the most ancient biochemical pathways in all of evolutionary history. Of course there are minor modifications in some organisms like for example the Entner-Doudoroff pathway which is a variant of glycolysis adopted by some prokaryotes [en.wikipedia: Entner-Doudoroff 2014]. The 'real' glycolysis (i.e. the most frequent one) on the other hand is sometimes also called Embden-Meyerhof-Parnas pathway (EMP pathway). In figure 1.3 on the next page one can see the enzyme-catalyzed reaction cascade by which the sugar glucose gets transformed into pyruvate with the ultimate result of energy production in the form of ATP (adenosintriphosphate) later on. ATP is the most important energy carrier in all life forms. Some details on the process of glycolysis depicted in figure 1.3 are explained below the figure.

Metabolic modelling

Metabolic reactions and pathways have been subject to mathematical modelling since the early days of enzyme kinetics [Michaelis, Menten 1913] and there exists a huge literature with diverse approaches on the topic. [Cornish-Bowden 2012] is a good introduction to classical enzyme kinetics and also to more modern topics. Almost every book on biochemical modelling or systems biology will usually also contain sections on enzyme kinetics and metabolic models [Klipp et al. 2009: chapters 2.1 and 3.1], [Kremling 2012: chapter 6]. A more mathematically detailed starting point is [Heinrich, Schuster 1996]. Approaches specifically designed for metabolic pathways are for example *biochemical systems theory* (BST) [Savageau 2009 (1976)] or *metabolic control analysis* (MCA) [Fell 1997], [Klipp et al. 2009: chapter 2.3], [Kremling 2012: chapter 10.3]. So called *stochiometric and constraint-based modelling*, for example *Flux Balance Analysis* (FBA), is also very closely associated to the modelling of metabolic pathways [Klamt, Stelling 2006], [Klipp et al. 2009: chapters 2.2 and 9.1], [Kremling 2012: chapter 13], [Haggart et al. 2011] and pathway optimization methods in metabolic engineering based thereon apply various mathematical optimization techniques [Torres, Voit 2002], [Klipp et al.: chapters 9.1 and 9.2] from classical non-linear optimization to linear and integer-linear programming. The so called *whole-genome*

reconstruction of metabolic pathways, i.e. the model-based reconstruction of the entire metabolic map of an organism and the elucidation of the associated 'implementation' of that map in the genome of the organism is reviewed for example in [Hefzi et al. 2013] and [Haggart et al. 2011].

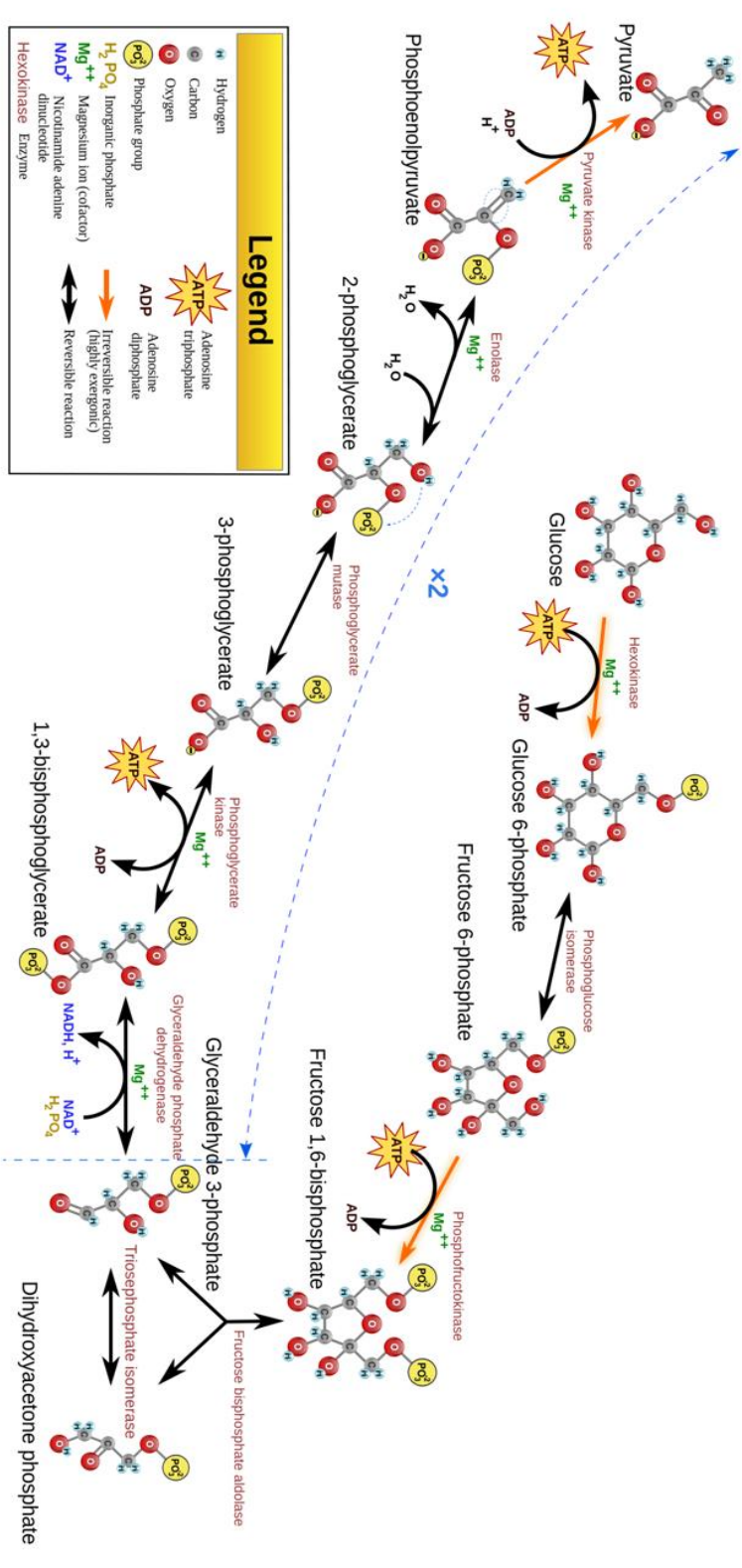


Figure 1.3. The Glycolytic pathway (from [en.wikipedia: Glycolysis; 2014]). Ignoring almost every detail depicted, I point out that first glucose is transformed to glucose-6-phosphate in a reaction which is catalyzed by an enzyme called hexokinase and in which ATP is transformed to ADP while a phosphate group is added to the C6-atom of glucose (which therefore become glucose-6-phosphate). Then in a next reaction step, glucose-6-phosphate is converted to Fructose-6-phosphate in a reaction catalyzed by an enzyme called Phosphoglucoseisomerase and so on and so forth until through successive enzyme-catalyzed conversions glucose is finally produced and energy (in the form of ATP and NADH) gained. The dashed blue line separating the lower reactions from the upper ones with the indication "2x" means that the lower reactions actually proceed two time because on the right there are two Glyceraldehyde-3-phosphate molecules produced for every glucose molecule. So in the end of the glycolytic pathway one molecule of glucose is converted into two molecules of pyruvate which is then further processed via the citric acid cycle (Krebs cycle) (not shown), oxidative phosphorylation and the electron transport chain which provide even more energy in the form of ATP [Munk et al. 2008]. The green Mg^{2+} sign over a reaction arrow means that the enzyme responsible needs enough magnesium ions in order to be able to catalyze the corresponding reaction.

1.3 Gene Regulatory Networks (GRNs)

As outlined gene regulatory networks (GRNs) are the biological focus of this thesis and therefore the description of this particular type of biochemical network is a little bit more detailed than the ones in the preceding sections. First we will encounter the basics of gene expression and its regulation in Subsection 1.3.1 while afterwards we will look more specifically at the two ingredients of GRNs, namely transcription factors (TFs) and cis-regulatory elements (CREs) in Subsection 1.3.2. Then in Section 1.3.3 we explore the interactions between TFs and CREs while Section 1.3.4 is concerned with the combinatorial action of TFs at CREs via so called gene regulation functions. Then, finally, in the last two subsections specific examples of GRNs are introduced: the infection of the bacterium Escherichia coli by λ-phage in Subsection 1.3.5 and the synthetic repressilator in Subsection 1.3.6.

General references for GRNs include [Bulyk, Walhout 2013], [Bolouri 2008], [Klipp et al. 2009: chapter 6], [Alon 2007] and [Davidson 2006]. For some basic material on gene expression it is referred to [Purves et al. 2006: Chapters 12, 13 and 14].

1.3.1 Gene expression: the central dogma (and beyond)

This subsection concerns the basic mechanism by which the genes of an organism exert their action: *transcription* and *translation.* Together (with some additional features indicated below) these processes are also called *gene expression*. Most information provided in this subsection is generally extracted from [Purves: Chapters 12, 13, 14], [Lewin 2008], [Klipp et al. 2009: Section 6.1] and [Orphanides, Reinberg 2002], all other sources are indicated in the respective places.

Gene expression is immensely complex. The following survey tries to give a basic overview but is certainly incomplete. For example, I spare out microRNAs [Klipp et al. 2009: Subsection 6.1.4], just to mention one omission and I am sure that there a many more I am not even aware of. For the purposes of this thesis, a double-stranded *DNA* (deoxyribonucleic acid) molecule is a pair of sequences $\mathrm{DNA} = \left(\mathrm{str}^{(1)}, \mathrm{str}^{(2)}\right) \in \left(\{\mathrm{A},\mathrm{T},\mathrm{C},\mathrm{G}\}^{\ell}\right)^2$ with $\mathrm{str}_i^{(1)} = \overline{\mathrm{str}_i^{(2)}}$ such that $\bar{\mathrm{A}} = \mathrm{T}$, $\bar{\mathrm{T}} = \mathrm{A}$, $\bar{\mathrm{G}} = \mathrm{C}$ and $\bar{\mathrm{C}} = \mathrm{G}$. $\ell \in \mathbb{N}$ denotes the length of the DNA molecule. Of course, A, T, C and G code for the bases adenine, thymine, cytosine and guanine which (together with so called desoxyribose and phosphate molecules) define DNA and form pairs (so called *base pairs*) according to the rule defined above (*complementary base pairing*). A and T as well as C and G are called *complementary bases* respectively. The structure of DNA is the famous double helix model introduced by James Watson and Francis Crick based on work of Rosalind Franklin and others in 1953 [en.wikipedia DoubleHelix 2014]. DNA molecules have a defined directionality as shown in the following figure 1.4 on the next page.

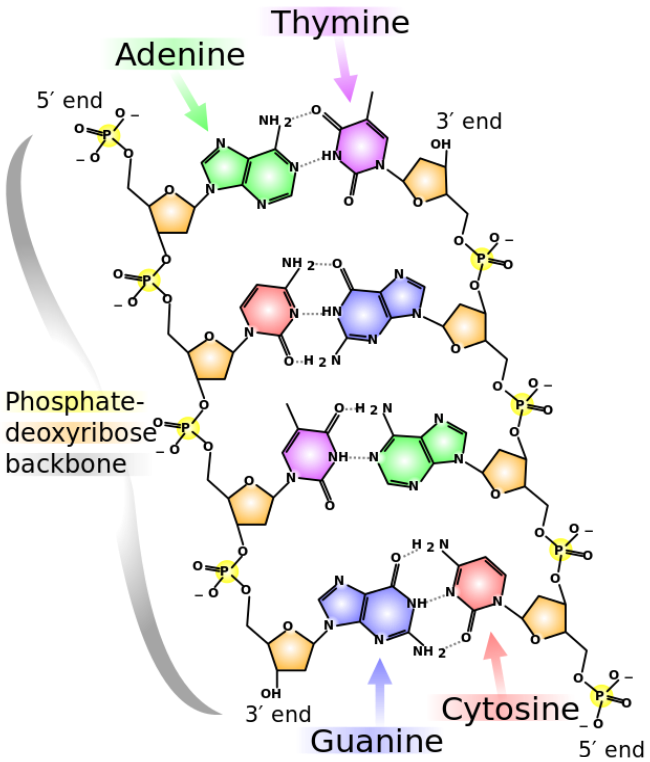


Fig. 1.4: **DNA directionality**. See main text for explanations. From [en.wikipedia DNA 2014]

On the left you can see a schematic picture of four base pairs of a DNA molecule. The bases pair according to the rules of complementary base pairing and the bases of one strand are connected by the so called *phosphate-deoxyribose backbone* (sugar-phosphate backbone). This backbone is a linear chain of alternating deoxyribose and phosphate molecules such that the bases are respectively attached to the sugar molecules. But as you can see the backbone is not symmetric in the sense that one end possesses a phosphate and the other does not and that this phenomenon is reversed in the two strands of the DNA molecule (the so called *antiparallelity* of the DNA strands).This of course has well-defined biochemical reason which I ignore.

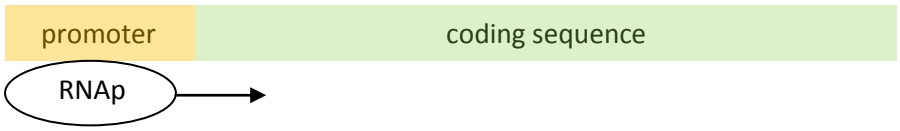
What is important here, is the directionality thus defined: a single strand of DNA has a direction defined by its respective so called 3' and 5' ends which are defined as the sugar ends and the phosphate ends of the sugar-phosphate backbone respectively.

DNA is the carrier of genetic information, the information being encoded in the sequence of base pairs. DNA sequence information is transformed (transcribed) into so called *mRNA* (messenger RNA) by a process called *transcription*. RNA (ribonucleic acid) molecules have multiple roles in the processes of life and are single linear chains of bases connected by a sugar-phosphate backbone (with thymine substituted by uracil (U) and the sugar being ribose instead of deoxyribose). The mRNA molecules transport the genetic information to the so called ribosomes (complex biomolecular machines) where the information is transformed (translated) to respective protein sequences by a process called translation according to the rules of the genetic code. This is the basic form of the central dogma of molecular biology as formulated by Francis Crick:

$$\text{DNA} \xrightarrow{\text{transcription}} \text{RNA} \xrightarrow{\text{translation}} \text{protein}$$

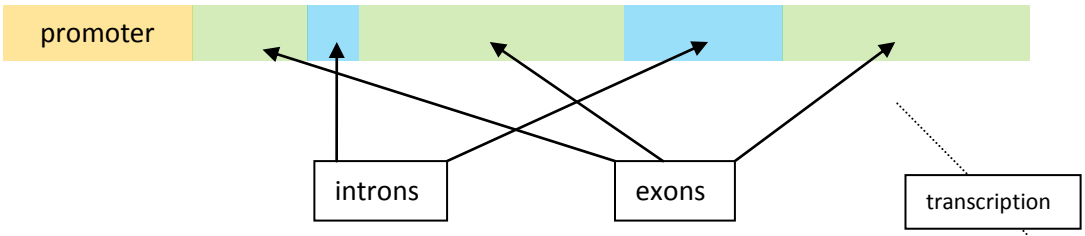
The proteins then fulfil the various tasks defining a living organism. This is nothing but the simplest summary of the process of gene expression capturing only the fundamental logic and there are a lot more subtleties involved, especially when it comes to the regulation of gene expression as explained in the following subsections. Figure 1.5 on page 19 shows the process of (eukaryotic) gene expression in more detail. For further explanations you can consult the subtext of this figure (maybe after you had a look at Subsections 1.3.2 and 1.3.3.

Before we proceed to aspects concerning the regulation of gene expression in the next subsections we have a closer look on how a gene is represented on/in the DNA. Actually, what is a gene? For the purposes of this thesis g*enes* are segments of DNA which are defined by a particular structure of gene defining DNA segments. This somewhat circular "definition" will become clearer in a moment. A gene (in its simplest form) looks like follows (where the boxes represent DNA segments on one of the DNA strands):

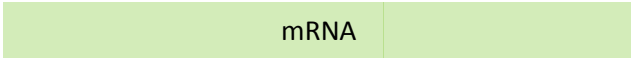


The so called *coding sequence* is the DNA segment of the gene which carries the actual information with respect to the protein the gene encodes. The segment called *promoter* is decisive in the initiation of transcription. The promoter is bound by *RNA polymerase* (RNAp) and other proteins which ultimately will trigger transcription by their characteristic interactions. RNA polymerase is then released and transcribes the coding sequence to mRNA. RNA is also based on bases and hence this just happens by complementary pairing.

The kingdom of life can be roughly divided into *eukaryotes* and *prokaryotes* (and archea). Eukaryotes are the "complicated" life forms like humans and yeast while prokaryotes are mainly represented by bacteria [Purves 2006: Part V], [Munk et al. 2008 : Section 1.1]. Eukaryotes possess a cell nucleus and cell organelles (mitochondria for example) while prokaryotes simply possess "one room for everything", i.e. they have no nucleus and no organelles. There are many more differences, but the important one here is that there are huge differences concerning the complexity of gene expression. It turns out that eukaryotic genes most of the time do not possess only one uninterrupted coding sequence but that the coding sequence is actually "scattered". The segments dividing the different coding segments (called *introns*) are called *exons*. After transcription the exons are removed by a process called *splicing*. So, the structure of a eukaryotic gene roughly looks like follows:



Conceptionally we thus arrive at the following mRNA transcript:



These are now some of the basics of gene expression. The next subsections deal with the regulation of gene expression by means of transcription factors.

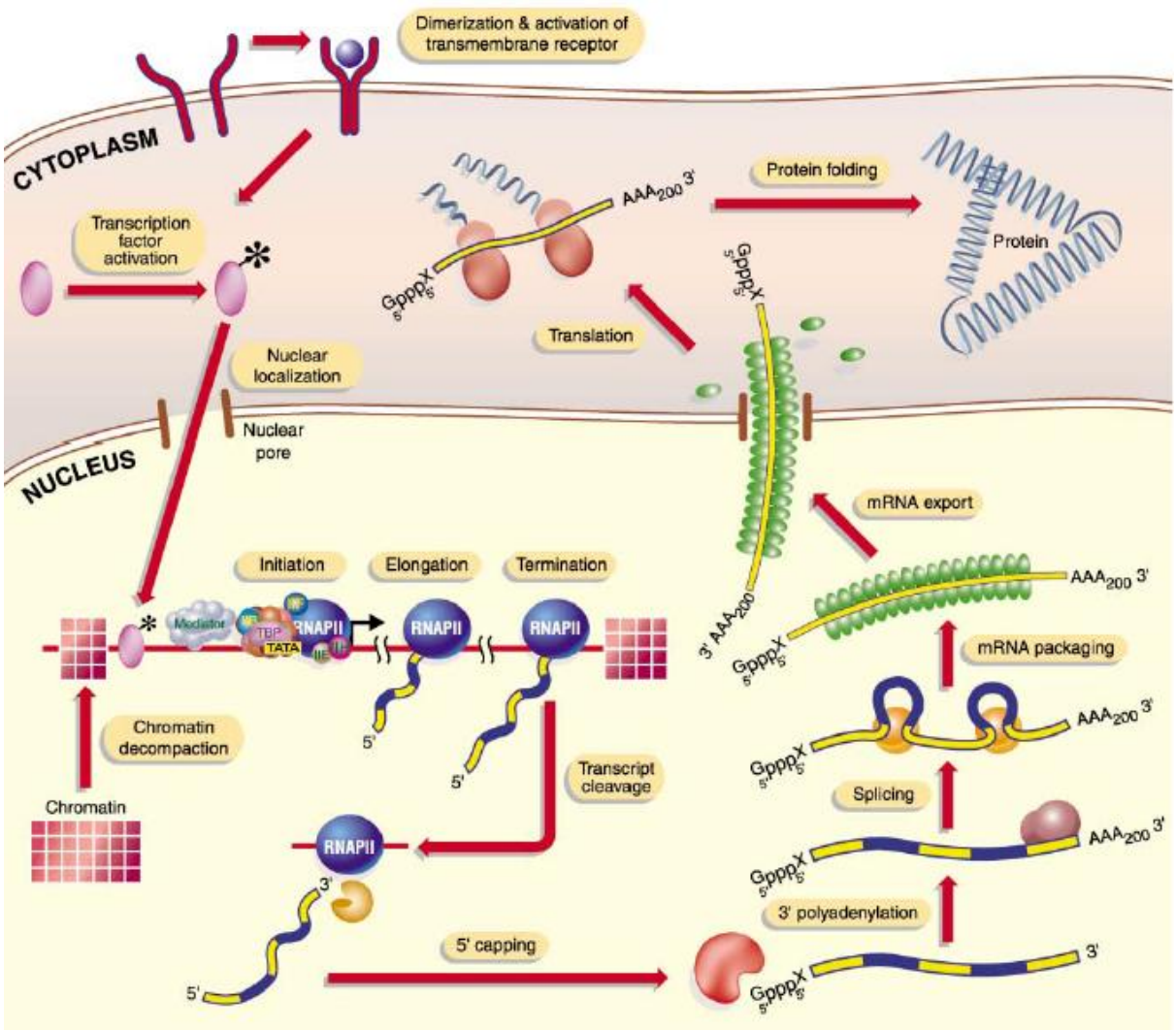


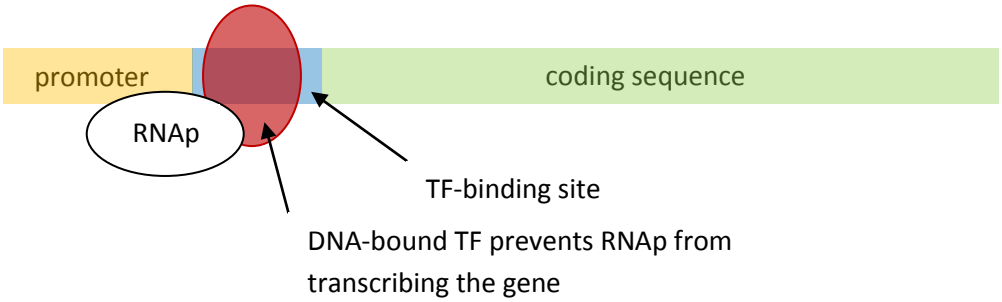
Fig. 1.5: **Gene expression.** From [Orphanides, Reinberg 2002]. The story starts on the bottom left: "Chromatin". ***Chromatin***, a conglomerate of proteins that is attached to the DNA, has to decompactify in order to enable gene transcription. When the chromatin is in its compact form, genes are so to speak not accessible by the transcription machinery. Chromatin decompaction is controlled by various ways which constitute an active area of research today [Dekker, van Steensel 2013]. Once the gene of interest is accessible the ***basal transcription machinery*** can assemble at the ***promoter*** (maybe assisted by some activating transcription factors and mediators, see Subsection 1.3.2): ***RNA polymerase II***, so called ***general transcription factors*** like TFIIB, TFIIE, TFIIF or TFIIH, the TATA-box binding protein (TBP) (and many more). This process is called ***transcription initiation***. After initiation the RNA polymerase is released from the initiation complex and starts to transcribe the DNA into RNA, a process known as ***elongation***. After termination the produced ***pre-mRNA*** is released. The processes of ***pre-mRNA processing*** involve the addition of a so called 5'-cap at the 5' end of the RNA (***5' capping***) and of a so called polyA tail at the 3' end (***3' polyadenylation***). These are mainly signals for transport and protection appendages. Then ***splicing*** takes place and the RNA is now simply called ***mRNA***, see the main text. After the mRNA is further packaged into a protecting protein coat it is exported to the cytoplasm where it is ***translated at the ribosomes***. The emerging polypeptide chains then fold to functional proteins. Some of these proteins are so called transcription factors (TFs) which can be activated for example by signalling cascades triggered by some external ligand molecule binding some respective transmembrane receptor. The activated TF is then "brought" to the nucleus through nuclear pores where it affects the expression of its target genes, see Subsection 1.3.2 and 1.3.3.

1.3.2 Transcription factors (TFs) and TF-binding sites

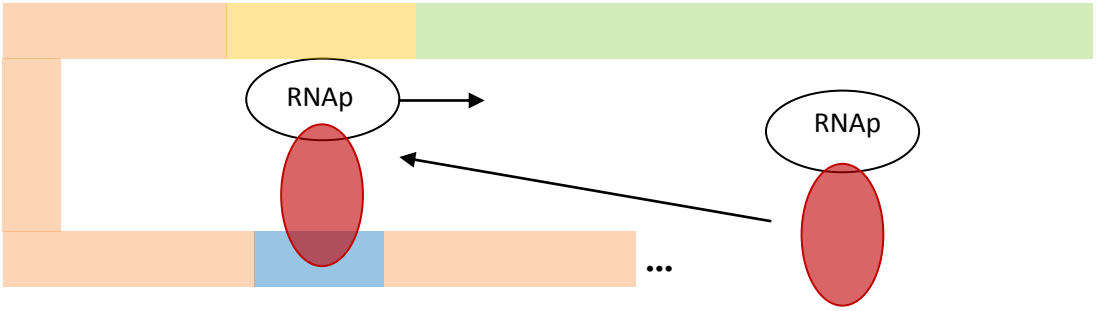
This subsection is based on [Bulyk, Walhout 2013], [Lewin 2008: Chapters 24 and 25] and [Klipp et al. 2009: Section 6.1].

The main ingredients of GRNs are proteins called transcription factors (TFs) and their respective DNA binding sites. As proteins, TFs are expressed like any other proteins by the already described transcription-translation path. But in contrast, for example, to proteins that act as signal transmitters or are part of the cell as building material, transcription factors can directly (!) influence the expression of genes (their target genes) by binding at specific places in the DNA called TF-binding sites. These binding sites are often located in front of the promoter of a gene and are often also called operators. Segments in front of the promoter which contain many TF-binding sites (usually for many different TFs) are also called *cis-regulatory elements or modules* (CREs/CRMs). "cis" refers to the direction of the gene defined by the directionality of DNA indicated in the preceding section. For our purposes "cis" simply refers to the location "in front" of a promoter.

When TFs bind to DNA they can have diverse effects on the regulated gene. In principle these can be activation or inhibition but the precise ways of influence are very diverse and so I just indicate two typical examples. TFs can repress (inhibit) transcription simply by physically blocking the RNA polymerase from transcribing the gene:



A typical situation of activation would be that a TF "recruits" RNAp to the promoter (usually with help from some so called *mediator*):



In the scheme on the last page, the TF "directs" the RNAp to the promoter and hence the transcription rate goes up. As indicated complex DNA loopings can also play a role in gene regulation, see for example [Vilar, Leibler 2003].

There a many more variations and subtleties but I leave it at that.

1.3.3 TF-DNA interactions: GRNs

In summary, we have genes which may code for TFs or some other protein. If they code for TFs, these influence the expression of other genes and hence one can draw a map (graph) where genes are connected if one encodes a TF which influences the other gene. This map (graph) then represents the notion of a gene regulatory network (GRN).

For the purposes of this thesis a GRN is assumed to be given, but in the real world it is very involved to figure out which TF binds to an operator of which gene and how different TFs effectively integrate their inputs if they are bound at the same time to operators influencing one and the same gene. See the next subsection for an indication of this particular topic. Furthermore things are complicated because TF-encoding genes can be alternatively spliced[1] or form variable multimers with different regulatory functions respectively [Bulyk, Walhout 2013]. So, the simple picture of one TF-encoding gene corresponding to one and only one TF is already a simplification.

More precisely, a GRN could be defined as a directed bipartite graph [Bulyk, Walhout 2013] with genes (or their regulatory sequences) as one vertex set and the corresponding proteins as the other vertex set while edges go from genes to proteins if the respective protein is encoded by the gene and edges go from proteins to genes if the respective protein has a regulatory influence on the particular gene. In this setting, proteins which do not function as TFs but fulfil some other role in the organism, for example in signalling or in bringing about the physical structure of the organism as building material, have only incoming edges but no outgoing edges. The question how to proceed with proteins which only act as heterotypic multimers (i.e. as assemblies of different protein subunits)[2] can in principle be settled by only considering the functional monomers as vertices and encode their multimerity by several ingoing edges which originate from the respective genes which code for the protein subunits. Analogously alternative splicing would result in several edges originating from the same gene vertex. For some possible exact definitions, see Subsection 1.5.1.

There exist various experimental and computational techniques to assess TF-DNA binding (or protein-DNA binding more generally) which are reviewed for example in [Bulyk, Walhout 2013].

[1] Alternative splicing refers to the fact that in the process of splicing (see Section 1.3.1) exons may be variably included or excluded into the final mRNA transcript [Lewin 2008: Chapter 26].

[2] A homotypic multimer consists of several numbers of the same subunits and would just be connected by a directed edge to its gene in the usual way from gene to protein.

1.3.4 Gene regulation functions

The gene regulation function of a gene ideally describes the transcription rate of a gene as function of all species which influence the expression of the gene [Klipp et al. 2009: Section 6.2], [Alon 2007: Subsection 2.3.5]. Theoretically, one can try to determine the functional form of gene regulation functions from purely thermodynamical considerations by making some assumptions, for example equilibrium binding of the various regulators of the gene to their DNA binding sites [Klipp et al. 2009: Section 6.2], [Bintu et al. 2005a], [Bintu et al. 2005b]. These functions can then be parameterized based on experimental measurements of the actual gene regulation function. Both was done for example in [Setty et al. 2003] where the promoter activity of the so called lac gene of E.coli was measured by a reporter gene construct in response to two of its regulators, cAMP (cyclic adeninemonophosphate) and IPTG (isopropyl-β-D-thiogalactopyranosid). The resulting gene regulation function is shown in Figure 1.6. [Kaplan et al. 2008] measured many two-dimensional gene regulation functions involved in the sugar utilization of E.coli cells by the same methodological rationale.

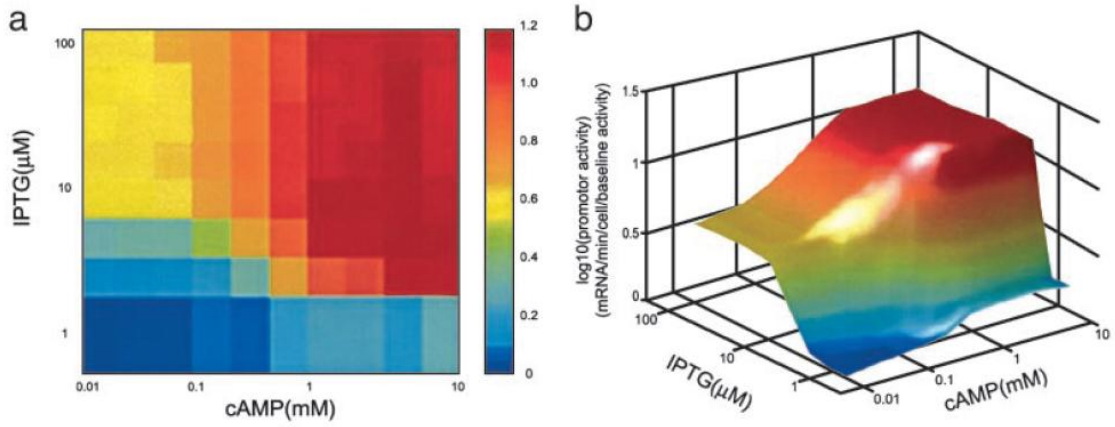


Fig. 1.6. From [Setty et al. 2003]. **A typical two-dimensional gene regulation function.**

1.3.5 Example 1: λ-phage infection of E.coli

This subsection indicates a concrete example of a GRN. The example concerns the gene regulatory network of so called λ-phages, viruses which infect E.coli bacteria. The following condensed description is based on [Ptashne 2004].

λ-phages consist of a protein coat and DNA which is enclosed by the coat. They attach to E.coli cells and insert their DNA (the protein coat remains outside) and usually this leads to the occupation of the host cell's gene expression machinery to express genes of the injected virus DNA in such a way that new virus DNA and coat proteins are produced until finally the new synthesized virus particles are released from the host. This process is referred to as lysis or lytic pathway. But in some situations a different scenario shows up. It is possible that

instead of pursuing lysis of the cell one of the virus DNA molecules gets integrated into the E.coli genome and establishes some kind of dormant virus state while equipping the cell with resistance to lysis. This is referred to as the lysogenic pathway. Upon UV radiation exposure for example the lysogenic phage DNA flip to a lysis pathway which is different from the one described first because the initial situation is different (the DNA has to be excised from the host's genome again for example) but which ultimately follows the same regulatory rules until enough new virus particles are manufactured and the new phages lyse the cell.

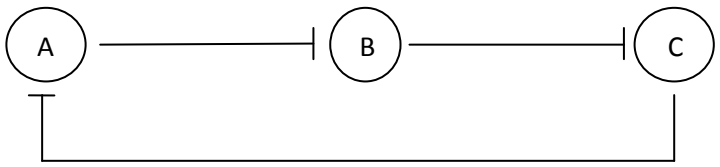
In terms of gene regulation, the lytic and the lysogenic pathways are mainly determined by two TFs encoded in the phage genome, named cI and cro. In the lysogenic state the situation with respect to the phage genes is such that the cI TF (also called λ repressor) is the only phage gene which is transcribed thereby activating its own expression and effectively inhibiting all other phage genes including the cro gene.

cI also prevents other phage DNA which enters the cell from inducing lysis, just by the same mechanisms of repression which are applied to the integrated phage genes. In the case of lysis on the other hand, cro is expressed and inhibits cI. In addition to inhibiting cI the cro TF also activates the (more complex regulatory pathway) which ultimately results in lysis.

As always, there a many more subtleties involved in λ-phage gene regulation. For example it is very interesting how the initial decision whether to take the lytic or lysogenic pathway is actually brought about in the first place. For this and many other thrilling issues one can have a look at [Ptashne 2004] and [Oppenheim et al. 2005].

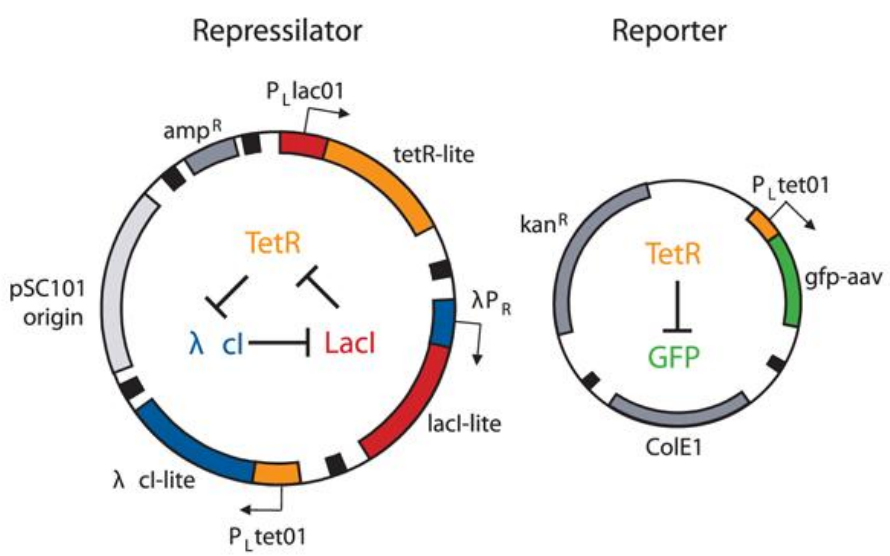
1.3.6 Example 2: the synthetic repressilator

[Elowitz, Leibler 2000] designed an artificial GRN, measured its functionality and examined a mathematical model of their constructed system. The system they constructed is known as the *repressilator*. It is built out of three species, A, B and C say, and its fundamental regulatory logic can be depicted as follows:



As shown above, the logic comes down to: A represses B, B represses C and C represses A. The repressilator examined by [Elowitz, Leibler 2000] was a synthetic one which was implemented using fundamental tool from genetic engineering. Many regulatory sequences of DNA are exchangeable in the sense that for example the promoter of gene x can drive the expression of gene y if located at the right place in front of y. [Elowitz, Leibler 2000] used

three different genes from different organisms and implemented them on a so called plasmid. Plasmids are relatively small rings of DNA which are the working horses of genetic engineering. The repressilator was constructed by placing a promoter which is negatively regulated by gene A in front of gene B, a promoter which is negatively regulated by gene B in front of C and finally a promoter which is negatively regulated by C in front of A. The dynamical behaviour of the repressilator system was tested in E.coli hosts by introducing another plasmid which carried a promoter which was repressed by gene A and controlled the gene encoding for a protein called green fluorescent protein (GFP) which can be detected with fluorescence microscopy if present in the system and hence could be used to investigate the system behaviour over time. As can somehow be expected from the logic shown in the diagram above, the system is able to show oscillations. Figure 1.7 shows the implementation of the system in the form of plasmids while Figure 1.8 depicts the oscillations of the GFP fluorescence over time. Both figures are from [Elowitz, Leibler 2000].



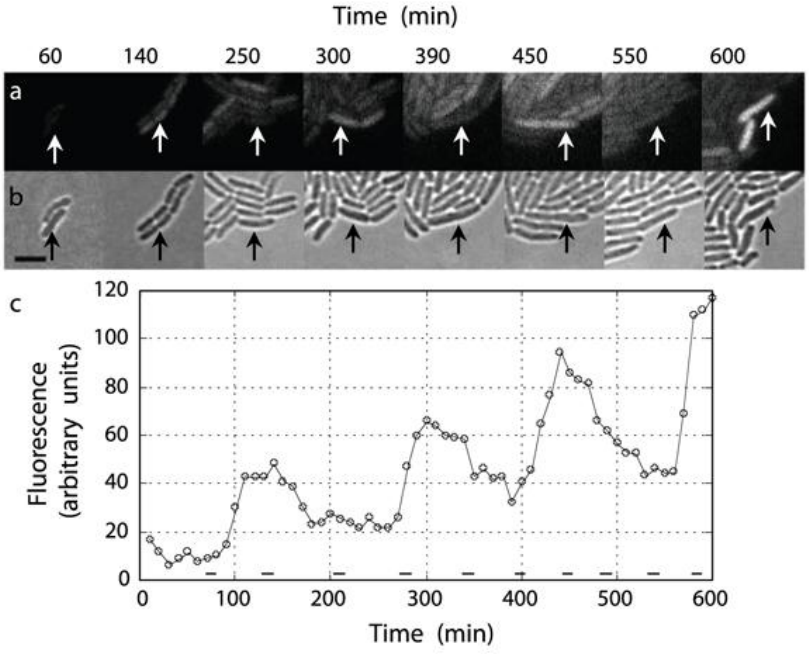


Fig. 1.7: Plasmid implement-tation of the repressilator and the reporter system.

[Elowitz, Leibler 2000].

TetR-lite, λ cI-lite and LacI-lite are the three ***genes*** consti-tuting the repressilator. gfp-aav is the GFP gene which is controlles by LacI. $P_L$lac01, $P_L$tet01 and λP_R are the ***promoters*** which are regulated by the products of the re-specttive genes. amp^R and kan^R are so called ***selection markers*** which are needed for reliable successful cloning. pSC101 and ColE1 are so called ***replication origins*** which are needed in order that the plasmids can be replicated in their host.

Fig. 1.8: Oscillations of the repressilator.

[Elowitz, Leibler 2000].

The first row of pictures taken over time is the fluorescence microscopical evaluation while the second row is the same spot pictured under normal conditions.

Figure c finally diagrams the measured GFP fluorescence. Oscillations are clearly visible. The linear increase in overall fluorescence could be due to the growth of the population. (Plasmids are also replicated…)

1.4 Interconnections between network types

As indicated in Section 1.1 on signalling pathways, different pathways can interact with each other, for example the so called BCR-, TCR- and TLR-pathways all interact with the NFκB-pathway as indicated on the upper left side in Figure 1.2 (although the exact way of interaction is not shown). These pathways are all signalling pathways. There are also many interactions between metabolic pathways and GRNs, metabolic pathways and signalling pathways or GRNs and signalling pathways (as already seen in the NFκB example since there, the activation of the NFκB complex, which is a transcription factor, by the signalling cascade ultimately leads to the regulation of some genes (responsible for the 'stress response') what clearly belongs to the realm of GRNs). One could say that living organisms are networks of different network types and the distinction between GRNs, metabolic and signalling networks is mainly motivated by the types of entities (genes and their CREs together with TFs vs. metabolites and enzymes vs. signalling proteins like kinases), the types of interactions (TF-DNA binding, transcription and translation vs. enzyme-catalyzed metabolite conversions vs. protein-protein interactions and minor modifications like phosphorylation or methylation) and their respective functions (gene regulation vs. energy production and biosynthesis vs. sensing of the environment). The three types of networks also typically act on different time-scales. For example, it can take hours for gene expression to be effectively altered while signalling pathways usually act on the scale of seconds [Alon 2007: chapter 1]. But in order to be able to understand the structure and function of living organisms it is necessary to cope with their various interconnections. For an overview on modelling approaches addressing this issue see [Covert et al. 2008] and [Gonçalves et al. 2013].

We now give two interesting examples which exemplify the interconnection issue further. The first one deals with the timed gene regulatory control of the arginine biosynthesis pathway (i.e. we have a GRN-metabolic interaction) while the second example deals with the so called Warburg effect related to cancer biology representing an interesting example for the interactions of metabolism with signalling.

Gene regulatory control of the E.coli arginine biosynthesis pathway

Arginine is one of twenty-three *amino acids* needed as building blocks of proteins [Munk et al. 2008: chapter 4]. *Amino acid biosynthesis pathways* are metabolic pathways that can build up amino acids given some initial substrates, often other amino acids. For example, in humans there are nine amino acids which cannot be synthesized de novo (i.e. humans posses no biosynthesis pathway for these amino acids) and these are called *essential amino acids* and have to be consumed via nutrition [en.wikipedia EssentialAA 2014].

In E.coli bacteria there exists an arginine biosynthesis pathway [Zaslaver et al. 2004] (in humans too) which is based on the three amino acid substrates glutamate (glutamic acid), glutamine and aspartate. Bacteria which encounter an arginine-free medium have to synthesize arginine themselves via this pathway. Bacteria which are grown in an arginine-rich

medium on the other hand shut down their arginine biosynthesis since it is much cheaper energetically for them to just pick up the arginine present in the medium. What it basically means to shut down a metabolic pathway is that the enzymes associated to that pathway are no longer produced by the transcription-translation machinery of the bacterium, i.e. the respective genes are no longer expressed and therefore after some time (assuming plenty of arginine in the medium) none of these enzymes will be present in the cell anymore (because of enzyme degradation).

[Zaslaver et al. 2004] have grown E.coli in arginine-rich medium (so that the arginine biosynthesis pathway was not active and its enzymes not expressed) and then transferred them to arginine-free medium making sure that the substrates for arginine biosynthesis glutamate, glutamine and aspartate were present. Then the gene expression of the enzymes responsible for the arginine pathway was measured via *reporter gene constructs* and it was found that the expression proceeded in a timed fashion in the sense that enzymes which are needed first are produced first. This is shown in Figure 1.9 below (from [Zaslaver et al. 2004]). A mathematical model of this timed transcriptional activation of a metabolic pathway was also developed by [Zaslaver et al. 2004].

So the regulation of the arginine biosynthesis pathway constitutes a nice example of GRN-metabolic network interconnection. See also [Alon 2007: Chapter 5] for an introduction to the described arginine biosynthesis example.

The Warburg effect

For an example which involves the interaction of signalling and metabolic pathways I shortly describe the so called Warburg effect. The Warburg effect is named after the biochemist Otto Heinrich Warburg (1883-1970) [en.wikipedia OHWarburg 2014] and describes the fact that cancer cells usually produce much more lactate from pyruvate than normal cells [CellSignaling Warburg 2014]. Pyruvate is the end product of glycolysis as described in Section 1.2 and given pyruvate most cells have different possibilities to process this chemical species further. Usually, given enough oxygen available, this is done via the citric acid cycle (Krebs cycle) in the mitochondria, see Figure 1.10.

In cancer cells however it may happen that pyruvate is converted in unusually high rates to lactate, which is normally only done if the cell lacks oxygen. Ignoring the danger of even further surpassing the acceptable threshold of non self-made, coloured pictures the situation describing the Warburg effect is shown in Figure 1.10. The interactions between signalling and metabolic pathways involved are also indicated.

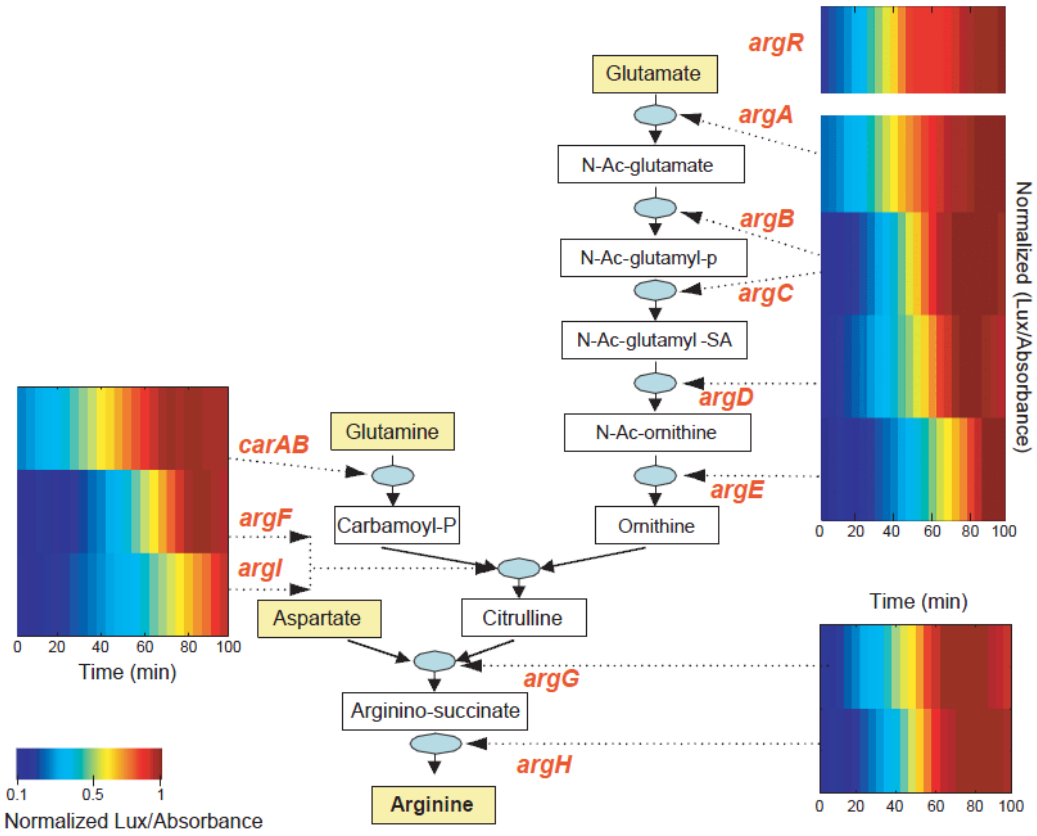


Fig. 1.9. Timed arginine biosynthesis. **From [Zaslaver et al. 2004].** As described in the main text arginine is ultimately produced from its three amino acid precursors. The intermediate species are given by the boxes with the respective names filled in while the enzymes responsible for the respective conversions are represented by the blue ovals between the boxes. The arrows carry the name of the respective enzymes and connect the enzyme boxes to the time-course data of the respective reporter gene constructs (interpretable via the "Normalized Lux/Absorbance" scale on the bottom left). Red values indicate high concentrations of the respective enzyme while blue ones represent low concentrations.

Integration of models for different network types

For conceptual clarification regarding the interactions between the different network types in living organisms I reproduce Fig.1 from [Goncalves et al. 2013], here it is Figure 1.11. As explained in the figure description and as exemplified in the examples above, there are various types of possible interactions.

Some modelling frameworks for biochemical networks are to some degree universally applicable, for example the detailed mechanistic approaches based on a chemical reaction network (see Subsection 1.5.1) like ODE-based or stochastic chemical kinetics models as reviewed in Chapter 2.

In order to model biochemical networks with these approaches "all one has to do" is to set up a detailed model of all the chemical reactions taking place in the network. With this approach it is (theoretically) irrelevant whether one examines signalling, gene regulatory or metabolic networks or a combination thereof.

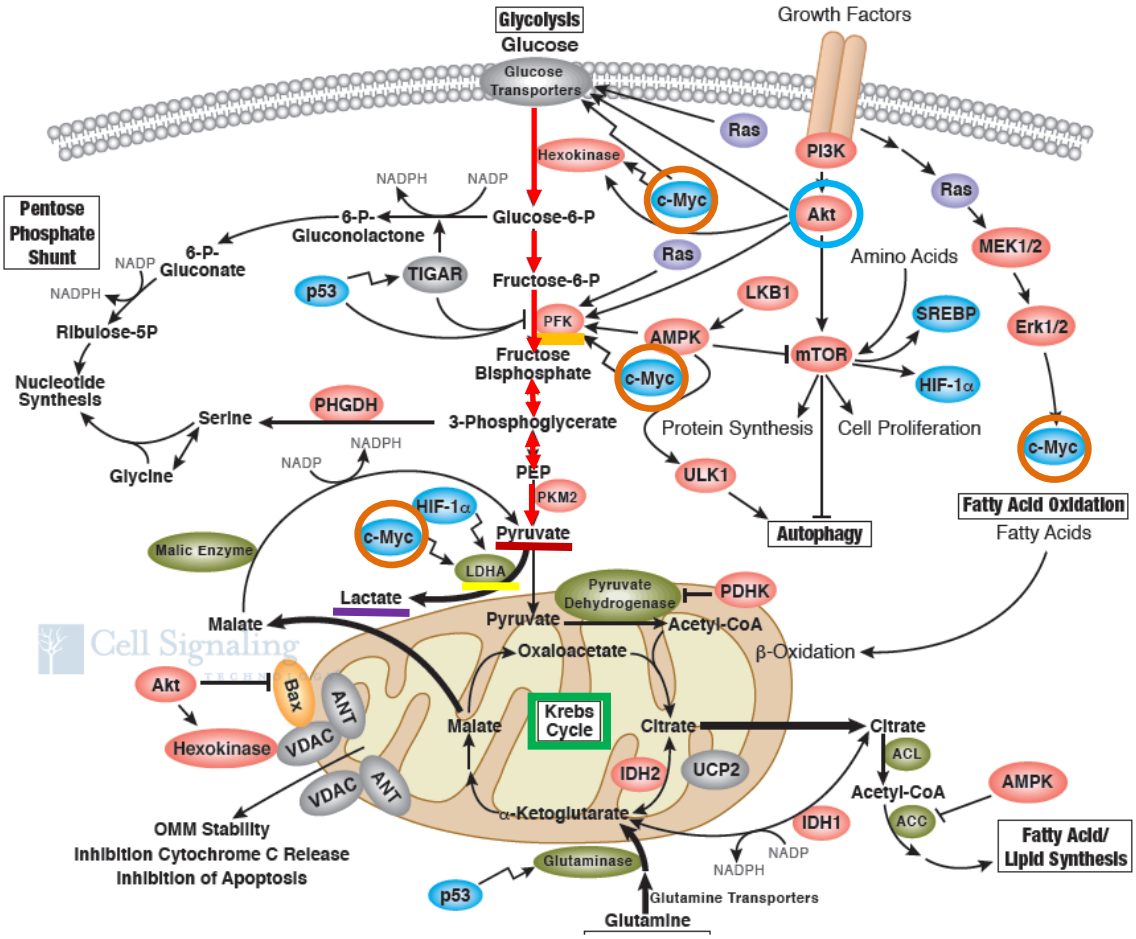


Fig. 1.10. Warburg effect. **From [CellSignaling Warburg 2014].** Ignoring again most aspects of the figure, the main issues relating to the indicated Warburg effect are colored (by myself). First, the reactions of glycolysis are colored red and the end product pyruvate is underlined in dark red. As already explained in the main text pyruvate can be further processed by the Krebs cycle (marked green) or alternatively transformed to lactate (underlined in purple) through the action of an enzyme called LDHA (lactate dehydrogenase A) which is underlined in yellow.

Concerning the interplay of the signaling and metabolic pathways you can see that through the action of so called growth factor receptor signaling pathways, indicated on the upper right of the figure, several signaling molecules are activated, for example Akt (circled light blue) and c-Myc (circled brown). These in turn induce effects on some enzymes involved in glycolysis (like PFK, phosphofructokinase, underlined orange) or the LDHA enzyme responsible for lactate production from pyruvate. In cancer cells the signaling pathways are often impaired due to the cancer-causing gene mutations and hence the described metabolic Warburg effect might become explainable. This is an active area of research and already very difficult for biochemists and hence I regard it as appropriate not to say anything more about these issues. But what remains, is another very interesting example for interacting network types, in this case signaling and metabolic pathways.

But mechanistic models become infeasible when the number of modelled species and interactions becomes larger and hence this approach has its technological limitations. To simplify matters when modelling large networks one often uses more coarse-grained qualitative models like Boolean models (see Chapter 3).

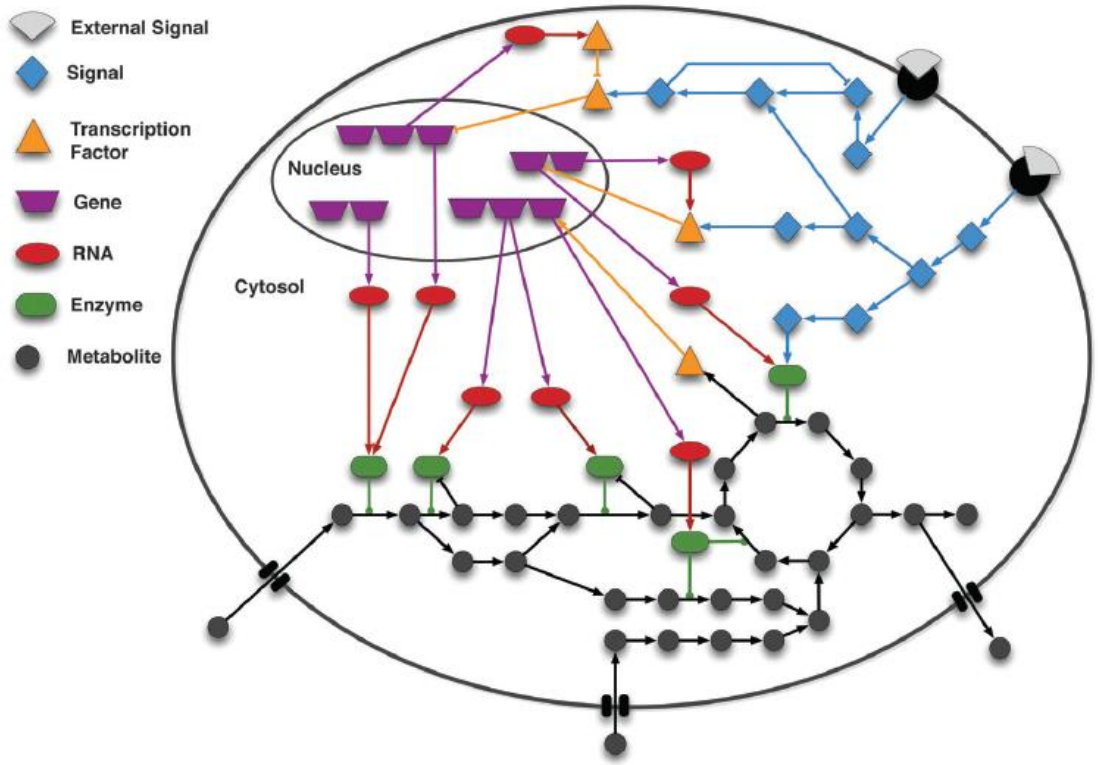


Fig. 1.11. Abstract visualisation between different types of biochemical networks. From [Goncalves et al. 2013]. As defined by the legend different networks and their interactions are shown. For example the merging of two red RNA ovals at one enzyme could mean that the enzyme is actually a dimer which is defined by the subunits defined by the two mRNAs. TFs are influenced by signalling molecules and metabolites or enzymes are of course also affected by signalling pathways as in the Warburg effect example. Conclusion: life is complicated!

Boolean models involve state variables which only can take values in the set $\{0,1\}$ indicating qualitative system properties like "gene transcribed", "no signal" or "no phosphorylation" etc. (see Chapter 3 for details). While this approach is applicable mainly to gene regulatory ("gene transcribed/not transcribed" or "TF present/not present") or signalling pathways ("signal vs. no signal" implemented for example via "methylation/no methylation") it is not well-suited to metabolic pathways since the decisive properties are not a matter of information as in signal or gene regulatory pathways but more related to mass flow as opposed to information flow [Goncalves et al 2013]. This is one reason for the various model types especially crafted for metabolic networks indicated in Section 1.2. I just make these remarks because I want to briefly foreshadow some of the issues which can arise when one tries to combine models of different networks types, for example a Boolean gene regulatory model with a classical metabolic model which also deals with mass flows. For a review on issues relating to the integration of models for different network types one could consult [Goncalves et al. 2013].

Modularity

A related concept concerns what usually is called *modularity* [Szallasi et al. 2006], [Klipp et al. 2009: Section 8.3]. Modularity is the idea that the web of biochemical networks which constitutes a living organism can be subdivided into functional modules where every module

has its characteristic function mainly implemented in the wiring of itself and to some extent independent of the "surrounding" networks. This implies that modules are more or less standalone networks which are capable of specific functionalities. The interaction of different modules then brings about the overall behaviour of larger systems. As an example, one can think of glycolysis. It has the function to produce pyruvate from glucose whereas pyruvate then has diverse routes of further utilization (think of to the Warburg effect above) which constitute other metabolic modules like the Krebs cycle or lactate production. Although the idea is very conceivable at first it is not so straightforward in general and it is an ongoing debate on how to define modularity properly in different contexts or even if modularity in biochemical systems is fact or fiction. For example, two "modules" can show functional modularity with respect to some specific functions but nevertheless together can show higher order functions not producible by each of them alone and so on and so forth [Szallasi et al. 2006], [Klipp et al. 2009: Section 8.3]. For a study concerning modularity in the context of signalling pathways see [Saez-Rodriguez et al. 2004].

1.5 Hypergraphs and systems biology

Systems biology aims at elucidating the highly complex dynamic behaviour which is brought about by the various interactions of a lot of single biomolecular entities [Walhout et al. (eds.) 2013]. That networks of various entities (see the preceding sections) play a crucial role in this endeavour is somewhat obvious. This section first reviews the proper mathematical setting for the conception of a "network" namely graphs or more generally hypergraphs. Subsection 1.5.2 finally ends the chapter by indicating the difference between the so called bottom-up and top-down approaches in systems biology.

1.5.1 Graph-based methods: hypergraphs

Many biological systems can be modelled to a first degree as some sort of graph and many more advanced models still incorporate some graph-based structure as their basis. This subsection gives some basic definitions, which are useful again and again in the sequel, and finally provides two detailed standard examples of graph-based modelling. In Section 3.1 we will see how hypergraphs can be utilized in the context of Boolean models. For reviews on the topic of graph-based models see [Klamt et al. 2009] and [Aittokallio, Schwikowski 2006].

As already mentioned right at the beginning of this chapter, intuitively a graph can be described a set of vertices (nodes) which are "in contact" or "interact" with each other via edges (arcs, connections). The edges can be undirected, meaning that contacts and interactions are symmetrical, or they can be directed, meaning that edges have a "start" vertex and an "end" vertex. Furthermore edges could carry signs or quantitative weights to encode for positive/negative effects or quantitative influences. This basic graph model can be extended to hypergraphs which are defined in the following. I follow [Klamt et al. 2009] which is an

accessible overview on (hyper-) graph-based methods, issues and examples in computational biology) in the definitions and examples.

For technical simplicity (and biological common sense) all occurring sets are assumed to be finite throughout the rest of this subsection.

Definition 1.5.1 (**Undirected hypergraph**) [Klamt et al. 2009]

> An **undirected hypergraph** is a tuple $H=(V,E)$ where V can be any set and E is a set of subsets of V. More formally, V a set, $E\subset\mathcal{P}(V)$.

The elements of V are called vertices or edges while the elements of E are called *hyperedges*. Undirected hypergraphs are also known as *set systems* and if $A\subset B \Rightarrow A=B$ for all $A,B\in E$ (i.e. if no edge contains another edge) the resulting hypergraph is also known as *Sperner hypergraph* or *clutter* [Klamt et al. 2009]. A biological example given in [Klamt et al. 2009] involves *protein-protein interaction networks*. Proteins can interact with each other and form complexes and one protein can participate in several different protein complexes. A natural undirected hypergraph representation of this kind of network is given with the proteins as vertices and the complexes (i.e. the subsets of proteins which form a complex respectively) as edges. See [Klamt et al. 2009] for further explanations (especially about algorithmic issues) concerning undirected hypergraphs. *Classical undirected graphs* (without self-loops) are obtained as hypergraphs with $E\subset\binom{V}{2}$ where $\binom{V}{k}:=\{A\subset V:|A|=k\},\ k=0,1,...,|V|$ and hyperedges are then simply called edges. Undirected graphs with self-loops are obtained with $E\subset\binom{V}{2}\cup\binom{V}{1}$.

Definition 1.5.2 (**Directed hypergraph**) [Klamt et al. 2009]

> A **directed hypergraph** is a tuple $D=(V,A)$ where V can be any set while $A\subset\mathcal{P}(V)\times\mathcal{P}(V)$, i.e. for every $a\in A$ there are $T,H\subset V$ such that $a=(T,H)$.

The elements of V are again called vertices or nodes. The elements of A are called *hyperarcs*. The *classical directed graphs* (with self-loops) can be recovered by setting $A\subset\binom{V}{1}\times\binom{V}{1}$.[3]

[3] Technically one then has to make the following identification: $(\{u\},\{v\})\triangleq(u,v)\ \forall u,v\in V$.

A classical directed graph $D=(V,A)$ is said to be *bipartite* if there exist $G,P\subset V$ such that $G\dot{\cup}P=V$ and $A\subset(G\times P)\cup(P\times G)$. As outlined in Subsection 1.3.4 a *GRN* could be represented by such a (via definition) bipartite graph where G is the set of genes in the network and P are functional (possibly multimeric) proteins. Concerning the edges one has for $g\in G$ and $p\in P$, $(g,p)\in A$ if gene g encodes for (at least) one subunit of protein p and $(p,g)\in A$ if protein p is a TF of g, i.e. edges $e\in G\times P$ correspond to gene expression and edges $r\in P\times G$ correspond to regulatory influences of TFs on genes. For a gene $g\in G$ one could define its *protein range* by $\mathcal{P}_g:=\{p\in P:(g,p)\in A\}$ and for proteins $p\in P$ similarly the *regulatory range* given by $\mathcal{R}_g:=\{g\in G:(p,g)\in A\}$. Non-regulatory proteins (signalling proteins, structural proteins, etc.) can be defined as proteins $p\in P$ with $\mathcal{R}_g=\varnothing$. Heterotypic multimer proteins would be characterized by more than one incoming edge at a given protein vertex and alternative splicing would correspond to several outgoing edges from a given gene vertex.

As for classical graphs one also can define weighted hypergraphs:

Definition 1.5.3 (**Weighted directed hypergraph**) [Klamt et al. 2009]

> A **weighted directed hypergraph** is a tuple $D=(V,A)$ where V is any set and A is the set of all tuples $a=(T,c_T,H,c_H)$ with $T,H\subset V$, $c_T:T\to\mathbb{N}$ and $c_H:H\to\mathbb{N}$.

Alternatively to defining the *weight functions* $c_T:T\to\mathbb{N}$ and $c_H:H\to\mathbb{N}$ to have the respective *tail vertices* T and *head vertices* H as domains one could also take the whole vertex as respective domains and set $c_T:V\to\mathbb{N}_0$ and $c_H:V\to\mathbb{N}_0$ with $c_{T|V\setminus T}\equiv 0$ and $c_{H|V\setminus H}\equiv 0$. In addition one could of course consider more general ranges for c_T and c_H but for most biological applications the integers seem to be sufficient. There are some (very formal) variations on the definition, for example, in Section 3.1 the logical interaction hypergraph is introduced where edges are of the form $a=(T,s,\{i\})$ for some $T\subset V$, $i\in V$ and a weight function $s:T\to\{-1,+1\}$, i.e. only the tail vertices are weighted.

Standard directed weighted graphs can be recovered via $|T|=|H|=1$ and the definition of a weight function that then applies to the edge itself. (One certainly could define the weight function formally in terms of c_T and c_H (in various ways) but this seems a little bit too formal and in our context is considered unnecessary.)

To exemplify the above concepts a detailed description of chemical reaction systems in terms of weighted hypergraphs as outlined also by [Klamt et al. 2009] is given.

Weighted hypergraph representation of chemical reaction networks

A *chemical reaction network* (or *system*) consists of (chemical) species $\mathcal{S}=\{\mathcal{S}_1,...,\mathcal{S}_n\}$ and reactions $\mathcal{R}=\{\mathcal{R}_1,...,\mathcal{R}_m\}$, where $n\in\mathbb{N}$ is the number of species and $m\in\mathbb{N}$ the number of reactions in the system. A reaction $\mathcal{R}_j$ is usually written in the form $\sum_{i=1}^{n} s_{ij}^r \mathcal{S}_i \to \sum_{i=1}^{n} s_{ij}^e \mathcal{S}_i$ where $s_{ij}^r \in \mathbb{N}$ represents the number of molecules of species i involved on the reactant side and $s_{ij}^e \in \mathbb{N}$ the number of molecules of species i involved on the educt side of the reaction, i.e. in order that reaction j can take place one needs s_{ij}^r molecules of species i and as consequence of reaction j taking place s_{ij}^e molecules of species i are produced. For example, consider a chemical reaction system with four species $\mathcal{S}_i$, $i=1,...,4$ and in which the reaction $\mathcal{R}_j$: $\mathcal{S}_1+2\mathcal{S}_2\to\mathcal{S}_2+\mathcal{S}_3$ takes place. Here, we have $s_{1j}^r=1$, $s_{1j}^e=0$, $s_{2j}^r=2$, $s_{2j}^e=1$, $s_{3j}^r=0$, $s_{3j}^e=1$ and $s_{4j}^r=s_{4j}^e=0$.

This kind of system can conveniently be represented formally as a weighted directed hypergraph as follows. The hypergraph is given by the tuple $(\mathcal{S},\mathcal{R})$ with the species $\mathcal{S}$ as vertices and the reactions $\mathcal{R}$ as edges. The edges, i.e. the reactions, are further specified by the tuples $\mathcal{R}_j=\left(\mathcal{S}_{(r)}^j, s_r^j, \mathcal{S}_{(e)}^j, s_e^j\right)$ with reactants $\mathcal{S}_{(r)}^j\subset\mathcal{S}$, educts $\mathcal{S}_{(e)}^j\subset\mathcal{S}$ and weight functions $s_r^j:\mathcal{S}\to\mathbb{N}_0$, $s_e^j:\mathcal{S}\to\mathbb{N}_0$ such that $s_r^j(\mathcal{S}_i)=0$ ($s_e^j(\mathcal{S}_i)=0$) exactly if $\mathcal{S}_i\notin\mathcal{S}_{(r)}^j$ ($\mathcal{S}_i\notin\mathcal{S}_{(e)}^j$). More precisely the reactants are defined as $\mathcal{S}_{(r)}^j=\left\{\mathcal{S}_{r_1(j)},...,\mathcal{S}_{r_{\alpha(j)}(j)}\right\}\subset\mathcal{S}$ where $\mathcal{S}_{r_k(j)}\in\mathcal{S}$ is the k-th reactant of reaction j with $k=1,..,\alpha(j)$ and $\alpha(j)=0,1,...,n$ being the number of reactants. Notice that the case $\alpha(j)=0$ and $\mathcal{S}_{(r)}^j=\varnothing$ is possible. This would correspond for example to some constant production of some housekeeping protein. Analogously $\beta(j)=0,1,...,n$ denotes the number of educts of reaction j and $\mathcal{S}_{(e)}^j=\varnothing$ could correspond to some export processes (with respect to the model boundaries) of the species out of the cell nucleus for example (if the model is restricted to processes in the nucleus).

The classical *stochiometric matrix* (see for example [Kremling 2012: Chapter 13]) $S=(s_{ij})_{\substack{i=1,...,n\\ j=1,...,m}}$ is encoded in the hypergraph framework through $s_{ij}=s_{(e)}^j(\mathcal{S}_i)-s_{(r)}^j(\mathcal{S}_i)\in\mathbb{Z}$ while one can define the so called *directed stochiometric matrix* $D=(d_{ij})_{\substack{i=1,...,n\\ j=1,...,m}}$ via $d_{ij}:=s_{(r)}^j(\mathcal{S}_i)\in\mathbb{N}_0$. The stochiometric matrices are useful for the compact definition of the mass action ODE system corresponding to a given reaction network, see Subsection 2.1.1.

The dependency matrix: An example of graph-based model analysis

Once a graph model is given one can subject it to various analytic techniques. Just to give one example I describe the concept of the so called *dependency matrix* introduced by [Klamt et al. 2007]. In intuitive terms the dependency matrix captures the essential influence species can have on each other. These are defined as activation, inhibition, total activation, total inhibition, ambivalent influence and non-influence and are precisely defined in terms of the underlying graph structure as follows.

Let us assume a (standard) weighted directed graph $G=(V,E,s)$ with vertex set $V\neq\emptyset$, edges $E\subset V\times V$ and weight function $s:E\to\{-1,+1\}$ is given. The weights of the edges are thought to represent the nature of the influence which the tail has on the head vertex and the graph G can also be conceived as a classical signed graph. Here, $s(u,v)=+1$ ($s(u,v)=-1$) means that vertex u has a positive (negative) influence on v. This can mean that v is "switched on" ("off") or simply that more of species u directly (!) leads to more (less) of species v. It may not always be possible to unambiguously associate a sign to a given edge because the influences may also depend on the context (i.e. the states of other vertices) as it is sometimes the case in Boolean models, see Chapter 3. Here, I will suppose that every edge can be uniquely signed independent of the state of the system which corresponds to the assumption that all edges respectively represent the same quality of influence regardless of the states of all other vertices.

Now, one can define a *directed path* of length $k\leq|V|$ as a tuple $(v_1,...,v_k)\in V^k$ of distinct vertices $v_i\in V,\ i=1,...,k$ such that $(v_i,v_{i+1})\in E$ for every $i=1,...,k-1$. v_1 is called the tail vertex and v_k is called the head vertex of the directed path. If in addition, $(v_k,v_1)\in E$ the tuple $(v_1,...,v_k)\in V^k$ is called a *directed cycle* (*loop*) of length k.

Given the above weighted (signed) graph structure and the definition of paths one can define activating and inhibiting paths. An *activating path* between to two vertices $v_1,v_k\in V$ is a directed path $p=(v_1,...,v_k)\in V^k$ such that $s(p):=\prod_{i=1}^{k-1}s(v_i,v_{i+1})=+1$ while an *inhibiting path* is given by a directed path $p=(v_1,...,v_k)\in V^k$ such that $s(p)=-1$. Further for any two nodes $u,v\in V$ we define $\mathcal{I}(u,v):=\{(u,...,v)\in\{u\}\times V^k\times\{v\}:k\in\mathbb{N}_0,(u,...,v)\text{ directed path}\}$ as the set of all directed paths (where $\{u\}\times V^0\times\{v\}:=\{(u,v)\}$) and partition it into the activating paths $\mathcal{I}^+(u,v):=\{p\in\mathcal{I}(u,v):s(p)=+1\}$ and the inhibiting paths $\mathcal{I}^-(u,v)$ defined by $\mathcal{I}^-(u,v):=\{p\in\mathcal{I}(u,v):s(p)=-1\}$, i.e. we have $\mathcal{I}(u,v)=\mathcal{I}^+(u,v)\dot\cup\mathcal{I}^-(u,v)$.

Based on the preceding definition we can already define four of the six above mentioned global influence signatures for given nodes $u,v\in V$:

$$\text{u } \textit{activates} \text{ v} \quad :\Leftrightarrow \quad \mathcal{I}(u,v) = \mathcal{I}^{+}(u,v)$$
$$\text{u } \textit{inhibits} \text{ v} \quad :\Leftrightarrow \quad \mathcal{I}(u,v) = \mathcal{I}^{-}(u,v)$$
$$\text{u is } \textit{ambivalent} \text{ for v} \quad :\Leftrightarrow \quad \mathcal{I}^{-}(u,v) \neq \varnothing \ \wedge \ \mathcal{I}^{+}(u,v) \neq \varnothing$$
$$\text{u is } \textit{without influence} \text{ on v} \quad :\Leftrightarrow \quad \mathcal{I}(u,v) = \varnothing$$

While these definitions are straightforward there are subtleties introduced by the possible existence of negative feedback loops. We already defined the notion of a directed loop above and a *negative feedback loop* is defined to be a directed loop $\ell = (v_1, ..., v_k) \in V^k$ such that $s(\ell) := s(v_k, v_1) \prod_{i=1}^{k-1} s(v_i, v_{i+1}) = -1$. If it happens that for a given directed path $p \in \mathcal{I}(u,v)$ there exist nodes $w_1, w_2 \in p$ (where $\in$ is abused in the canonical way[4]) such that there is a negative feedback loop ℓ such that $w_1, w_2 \in \ell$ we say that p is negatively looped. For $u, v \in V$ we set $\mathcal{L}^{+}(u,v) := \{p \in \mathcal{I}^{+}(u,v) : p \text{ is negatively looped}\}$ and analogously $\mathcal{L}^{-}(u,v) := \{p \in \mathcal{I}^{-}(u,v) : p \text{ is negatively looped}\}$. With these notions defined we can finally define the remaining two global influence signatures for any $u, v \in V$:

$$\text{u } \textit{totally activates} \text{ v} \quad :\Leftrightarrow \quad \mathcal{L}^{+}(u,v) = \varnothing$$
$$\text{u } \textit{totally inhibits} \text{ v} \quad :\Leftrightarrow \quad \mathcal{L}^{-}(u,v) = \varnothing \ .$$

The *dependency matrix* for the graph G is then defined as $D = (d_{uv}) \in \mathcal{C}^{|V| \times |V|}$ where $\mathcal{C} = \{c_1, ..., c_6\}$ is any set of six distinct symbols that respectively code for one of the global influence signatures: u *activates* v $\Leftrightarrow \mathcal{I}(u,v) = \mathcal{I}^{+}(u,v) :\Leftrightarrow d_{uv} = c_1$, etc. For an example of a dependency matrix, see figure 3.1 on p. 96.

1.5.2 Top-down "vs." Bottom-up modelling

Bottom-up modeling refers to the modeling approach which addresses specific small- to medium scale biochemical systems and models these by incorporating biochemical knowledge like reaction mechanisms or at least the qualitative nature of (known) interactions based on data usually generated by small-scale often model-specific experiments [Kell, Knowles 2006]. Bottom-up models happen to be the focus of this thesis and will be examined in some facets in the following chapters. Typical bottom-up approaches comprise ODE-models (Section 2.1) or models based on stochastic chemical kinetics (Section 2.2). But

[4] Let $A \neq \varnothing$ be any set and $k \in \mathbb{N}$. Then the mapping $\text{set} : A^k \to \mathcal{P}(A)$ is defined by $\text{set}(a) := \{a_1, ..., a_k\} \subset A$ for $a = (a_1, ..., a_k) \in A^k$. Notation ($\alpha \in A$, $a \in A^k$): $\alpha \in a \ :\Leftrightarrow \ \alpha \in \text{set}(a)$.

Boolean models (Chapter 3), although usually dealing with larger systems, can also be seen as bottom-up modeling tools (depending on the usage).

In contrast *top-down models* are usually designed for (characteristically large-scale) systems where most mechanism of interactions of the system-constituting parts are not well-known and naturally have a statistical character. Top-down models address the question of inference on partially unknown systems based on (usually high-throughput) datasets like massive DNA microarray data. So in effect, top-down approaches are statistical models. One of the most used model types in statistical modeling in general and in systems biology are so called *Bayesian networks* [De Jong 2002], [Friedman et al. 2000]. One keyword in this area is *reverse engineering* which describes the area of study which tries to infer the overall topological and interactive structure of biochemical systems based on experimental data with the ultimate aim of automatization [Markowetz, Spang 2007], [Huang et al. 2009]. Reverse engineering is model-based and (dynamic) Bayesian networks (DyBNs) is only one example for a statistical model which was applied in order to infer the structure of regulatory networks [Friedman, Koller 2003]. Based on the models used, the inferred models (if it happens to be possible to infer something) of course somehow may provide different "kinds" of structural information which has to be interpreted from model to model. There is also a relatively big literature on reverse engineering Boolean network models (see Chapter 3). The first and most famous algorithm attacking this task is the so called REVEAL algorithm [Liang et al. 1998]. Further approaches include for example (!!!) Bernoulli mixture models [Saeed et al. 2012] or delay inference with MCMC methods [Dümcke et al. 2014].

2 Modelling approaches for biochemical systems

This chapter deals with approaches for the dynamic modelling of biochemical networks. Section 2.1 introduces ODE-based models (ODE: ordinary differential equations), Section 2.2 deals with classical stochastic models and Section 2.3 indicates some contexts that might not be covered by the approaches described in the preceding sections.

2.1 Reaction rate equations: ODE models

General introductions for classical ODE models in the context of biochemical systems can be found [Murray 2004], [Edelstein-Keshet 2005], [Klipp et al 2009: Chapter 2] or [Conrad, Tyson 2006].

In order to apply ODE-based models to a chemical of reaction system, it is assumed that the system is *spatially homogeneous* and *well-stirred*, i.e. everything is mixed up such that at every point in the "*reaction space"* (i.e. the physical volume where the reactions take place, for example a test tube) every reaction is exactly as probable as in every other point of the reaction space. This implies that the concentration levels of all species are constant throughout the reaction space. In addition the system is non-stochastic in the sense that there are so many molecules of each species that the probabilities for the reactions are constant over time and space, i.e. the system is deterministic and can be modelled as a system of ordinary differential equations [Klipp et al. 2009: Section 2.1], [Conrad, Tyson 2006].

An n-dimensional system of ODEs describing a biochemical system with n species can be written down in its basic form as follows:

$$\begin{aligned} \dot{x}(t) &= f\left(x(t)\right) \quad \forall t \in \mathbb{R}_{\geq 0} \\ x(t_0) &= x_0 \end{aligned} \tag{2.1}$$

Here, $x(t) = \left(x_1(t), ..., x_n(t)\right)^T \in \mathbb{R}^n$ is the continuous time-dependent vector of concentration variables, i.e. $x_i(t)$ is the concentration of species $i = 1, ..., n$ at time $t \in \mathbb{R}_{\geq 0}$. $f : \mathbb{R}^n_{\geq} \to \mathbb{R}^n_{\geq}$ describes the dependence of the rate of change of the concentrations (given through the derivative $\dot{x}$ of x) as a function of the concentrations. $x(t_0) = x_0 \in \mathbb{R}^n_{\geq 0}$ with $t_0 \in \mathbb{R}_{\geq 0}$ defines the initial value of the system. (2.1) constitutes a classical initial value problem for ordinary differential equations (with time-independent right-hand side) [Walter 2000].

Most of the time, the right-hand sides involved in biochemical modelling fulfil certain smoothness conditions such that the theorem of Picard-Lindelöf can be applied guaranteeing the existence of a unique solution to (2.1) [Walter 2000: II § 6]. However, see Section 4.2 on piecewise linear differential equations.

As introduced in Subsection 1.5.1 a (bio-) chemical reaction system $(\mathcal{S},\mathcal{R})$ is a hypergraph with species $\mathcal{S}=\{\mathcal{S}_1,...,\mathcal{S}_n\}$ and reactions $\mathcal{R}=\{\mathcal{R}_1,...,\mathcal{R}_m\}$, where $n\in\mathbb{N}$ is the number of species and $m\in\mathbb{N}$ the number of reactions in the system. As also detailed in 1.5.1 the reactions can formally be represented as the edges of a chemical reaction system hypergraph with the species as vertices. Here we only work with the resulting stochiometric matrix $S=(s_{ij})_{\substack{i=1,...,n\\ j=1,...,m}}\in\mathbb{Z}^{n\times m}$ and the directed stochiometric matrix $d=(d_{ij})_{\substack{i=1,...,n\\ j=1,...,m}}\in\mathbb{N}_0^{n\times m}$.

The classical *mass action ODE system* [Klipp et al. 2009: Subsection 2.2.1], [Keener, Sneyd : Chapter 1], associated to such a reaction network is defined by $f(x(t)):=S\upsilon(x(t))$ where S is the stochiometric matrix and $\upsilon=(\upsilon_1,...,\upsilon_m):\mathbb{R}^n_{\geq 0}\to\mathbb{R}^m_{\geq 0}$ is given by $\upsilon_j(x):=k_j\prod_{i=1}^{n}x_i^{d_{ij}}$ where $x=(x_1,...,x_m)\in\mathbb{R}^n_{\geq 0}$, $k_j\in\mathbb{R}_{\geq 0}$ and directed stochiometric matrix D. $\upsilon_j(x)$ is called the *reaction rate* of reaction j given concentrations $x\in\mathbb{R}^n_{\geq 0}$.

Mass action kinetics, i.e. mass action ODE systems, are based on the assumption that the occurrence of a reaction j and hence its rate υ_j is directly proportional to the product of the involved reactants. The proportionality constants $k_j\in\mathbb{R}_{\geq 0}$ are called *kinetic constants* (*parameters*).

As always, there are situations where the assumption of mass action kinetics may not be valid. It turned out that some metabolic systems (which otherwise fulfill all the assumptions concerning the appropriateness of an ODE-based approach) can be better modelled by so called *S-systems* [Klipp et al. 2009: Subsection 2.1.5].

On the other hand, the detailed modelling of all reactions by mass action kinetics can be unnecessary for some reason. For example, the rate of product formation of enzyme-catalyzed reactions may be better modelled via some specific enzyme kinetic law, like *Michaelis-Menten kinetics* for example [Klipp et al. 2009: Subsection 2.1.3], [Cornish-Bowden 2012: Chapter 2]. There, the corresponding differential equation for the product species p could (assuming that it is only produced by an enzyme-catalyzed reaction following Michaelis-Menten kinetics and degraded with rate $\gamma\in\mathbb{R}_{\geq 0}$) be written as $\dot{p}=\frac{v_{max}es}{K+s}-\gamma p$ where $v_{max}\in\mathbb{R}_{\geq 0}$ is the maximal conversion rate, $K\in\mathbb{R}_{\geq 0}$ is the so called Michaelis(-Menten) constant and s and e denote the concentrations of substrate and enzyme respectively. The decisive point here is, that the Michaelis-Menten law can be derived from a detailed mass action law involving an enzyme-substrate intermediary complex ES (a chemical species) and in principle it is possible to model larger systems by explicitly incorporating these kinds of intermediates involved in the enzymatic reactions [Klipp et al. 2009: Section 2.1]. But since it is possible to deduce the ES-independent Michaelis-Menten kinetic given the so called quasi-steady-state assumption (or corresponding to a slightly alternative approach, the quasi-equilibrium assumption) (see [Murray 2004: Chapter 6], [Edelstein-Keshet 2005: Chapter 7]

or [Klipp et al. 2009: Subsection 2.1.3]) one can (given that these assumptions about the enzymatic reaction are fulfilled) substitute the Michaelis-Menten rate in place for the full mass action laws and thus kick out the intermediate ES species from the model.

In a similar fashion one often encounters rate laws which are governed by so called *Hill functions* which are given by $h^+(x_i,\theta_i,m) := \frac{x_i^m}{x_i^m+\theta_i^m}$ and $h^-(x_j,\theta_j,m) := 1-h^+(x_j,\theta_j,m)$ [De Jong 2002], [Alon 2007: Chapter 2 and Appendix A]. Here, $m,\theta_i \in \mathbb{R}_{\geq 0}$ are parameters and x_i is the concentration of some species i. Hill functions are sigmoid functions with inflection point θ_i such that $h^+(0,\theta_i,m)=0$ and $h^+(x_i,\theta_i,m)\to 1$ as $x_i\to\infty$ ($h^-(0,\theta_i,m)=0$ and $h^-(x_i,\theta_i,m)\to 0$ as $x_i\to\infty$) while for large enough m one has $h^+(x_i-\varepsilon,\theta_i,m)\approx 0$ and $h^+(x_i+\varepsilon,\theta_i,m)\approx 1$ ($h^-(x_i-\varepsilon,\theta_i,m)\approx 1$ and $h^-(x_i+\varepsilon,\theta_i,m)\approx 0$ for small $\varepsilon>0$. Formally, $h^+(x_i-\varepsilon,\theta_i,m)\to 0$ and $h^+(x_i+\varepsilon,\theta_i,m)\to 1$ as $m\to\infty$. So, Hill functions are (for large enough m) step-like functions and can therefore describe switch-like changes of influence of a given species in dependence of its concentration. The location of that switch is defined by the *Hill threshold parameter* θ_i and the strength of the switch is defined by the *Hill coefficient* m. Physically Hill functions often arise through a phenomenon called *cooperativity* and *multimerization* already described in Subsection 1.3.5 in the context of gene regulation functions which indicates that Hill functions play a natural role in the modeling of some processes of gene regulation. For illustration assume that some gene is controlled by a TF in such a way that mRNA is transcribed at rate (approx.) $a\in\mathbb{R}_{\geq 0}$ whenever the TF is bound to some operator region in the form of a tetramer and is more or less untranscribed when no TFs are bound to the operator. Then the resulting mRNA dynamics can be modeled via

$$\frac{d}{dt}[\mathrm{mRNA}] = \frac{a[\mathrm{TF}]^4}{\theta+[\mathrm{TF}]^4} - \gamma[\mathrm{mRNA}]$$

where $[\mathcal{S}_i]$ denotes the concentration of species $\mathcal{S}_i$, $\theta>0$ is some multimerization-specific threshold and $\gamma>0$ the rate of mRNA degradation.

Once a reaction network is transformed into an appropriate system of ODEs one can try to solve this system. This is usually only possible by *numerical integration methods* [Deuflhard, Bornemann 2008], [Quateroni et al. 2002]. Notice however that in ODE systems derived from biochemical reaction networks one often has to deal with so called stiff systems [Petzold, Gillespie 2006], [Deuflhard, Bornemann 2008: Chapter 6]. Complementary to numerically solving the ODE system, one can also try to apply the various *qualitative methods of dynamical systems theory* like stability and bifurcation theory to gain insights about the system [Guckenheimer, Holmes 2002], [Murray 2004], [Edelstein-Keshet 2005].

A further challenge is the parameterization of ODE models, i.e. the *inference of suitable kinetic parameters from experimental data* [Ashyraliyev et al. 2009], [Jaqaman, Danuser 2006]. A parameterization clearly is a prerequisite for solving the system numerically. An important question relating to the estimation of the kinetic parameters (and for parameter estimation in general) is for example (!) the issue of *non-identifiability of parameters*. Usually, one differentiates between *a priori* (or *structural*) *non-identifiability* and *a posteriori* (or *practical*) *non-identifiability* [Raue et al. 2009]. Structural non-identifiability describes a situation where a parameter (or a set of parameters) cannot be estimated properly because the overall structure of the system is such that particular parameters are in principle not estimable whatever the data may be. A typical example would be two parameters which only appear via their respective product in the system equations. On the other hand practical non-identifiability describes a situation where parameters are not properly estimable because of the concrete data set used to conduct the estimation. [Raue et al. 2014] compare different methods for identifiability analysis.

2.2 Stochastic models for biochemical networks

In this section I will give a condensed review on the stochastic modelling of biochemical networks. The need for stochastic models is shortly motivated in Subsection 2.2.1. The remaining sections show, in a semi-rigorous way, how to proceed from the chemical master equation (CME) (Subsection 2.2.2) and the equivalent exact simulation schemes like the Gillespie algorithm (Subsection 2.2.3) consecutively to the approximate simulation schemes called τ-leaping methods (Subsection 2.2.4), from there to Langevin-leaping and the chemical Langevin equation (CLE) and finally to the deterministic RRE setting (Subsection 2.2.5). The given presentation of the indicated model cascade is mainly inspired by [Petzold, Gillespie 2006] and supplemented with some elements from [Ullah, Wolkenhauer 2012]. The section finally ends with Subsection 2.2.6 which tries to give a very basic overview on parameter estimation for stochastic models in systems biology.

There are mainly three reasons for the existence of this section. First, it is motivated by the general theme of this thesis, i.e. the relation between different modelling approaches, but in contrast to the title of the thesis “From discrete Boolean networks to stochastic continuous models” in this section the appropriate heading would rather be “From discrete-state stochastic models via continuous-state stochastic models to continuous-state deterministic models” (or vice versa) even if this heading does not capture every aspect of the indicated model hierarchy as will be seen in the sequel. Second, in Section 5.2 I shortly look at so called PLSDEs (piecewise linear stochastic differential equations) which are straightforward extensions obtained by introducing Langevin-type stochasticity to the differential evolution of the well established piecewise linear differential equation (PLDEs) models. PLDEs are reviewed in Section 3.6. Finally, stochastic models increasingly become inevitably more and more used and valued in systems biology [Wilkinson 2009] and the study of stochasticity in biochemical systems is an extremely interesting field of modern biology (Subsection 2.2.1). In

summary, a semi-short description of the most basic modelling tools involved in this area of study is included here.

2.2.1 Stochasticity in biomolecular systems

Stochasticity in biomolecular systems is a huge and highly active field of study. For review, see for example [Raj, van Oudengaarden 2008]. Typical studies address for example the distinction between *intrinsic and extrinsic noise* [Elowitz et al. 2002][5] or how noise spreads in larger networks [Pedraza, van Oudengaarden 2005]. Further issues concern what is called stochastic fate decision. For example, the decision of sporulation in Bacillus subtilis is known to be governed by noise [Maamar et al. 2007]. Also, the fate decision of λ-phages (described in Subsection 1.3.5) was conjectured to be influenced by noisy processes [Arkin et al. 1998], [Singh, Weinberger 2009]. But there are also studies indicating that noise is maybe not everything. For example, [St. Pierre, Endy 2008] shows that phage fate might be significantly influenced by the size of the infected E.coli hosts. [Zang et al. 2010] provide evidence that the number of phages invading a given host is a decisive determinant of fate decision. The principal question is indicated in figure 2.1 (from [St. Pierre, Endy 2008]). There are many more interesting issues and already the literature merely about stochasticity and λ-phages is extremely large. In the following subsections, the main classical modeling tool for stochastic biochemical systems are outlined while in Chapter 5 we come back to another kind of stochastic model class in the context of stochastic hybrid systems.

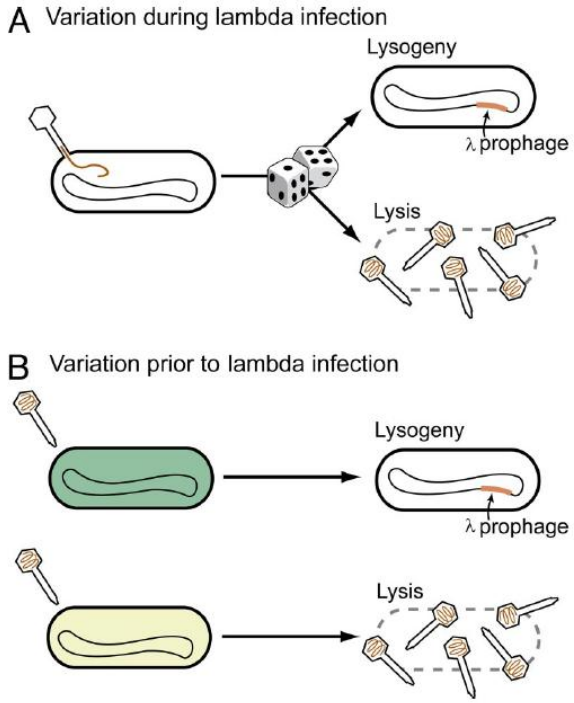


Fig. 2.1 (from [St.Pierre, Endy 2008]): Stochastic or deterministic cell fate decisions?

The figure shows the basic scientific dispute in the field of bioregulatory fate decision. In the λ example, the question is whether the decision to pursue lysis or lysogeny (see Subsection 1.3.5) is governed by chance (A) or by so far unknown deterministic variation (B) in terms of the host cells or the number of infecting phages, etc.

There is evidence for both (see the main text) and hence it seems likely that the truth is some in between.

[5] Intrinsic noise can be seen as noise which stems from molecular noise which is not due to some other kind of coincidence like the number of RNA polymerases in a given cell or so. The latter would be referred to as extrinsic noise since it influences different cells differently while intrinsic noise would (in theory) affect every cell the same way.

2.2.2 Stochastic chemical kinetics: the chemical master equation (CME)

In this subsection the chemical master equation (CME) is reviewed. The next subsection deals with the exact Gillespie-type simulation approaches which are used to sample trajectories from the stochastic dynamics described by the CME. Naturally the CME and Gillespie-type simulation are equivalent in a sense made precise in the following subsection and rely on the same fundamental modelling assumptions. In addition to the assumptions of spatial homogeneity and a well-stirred system (which they share with RREs) they are most fundamentally based on the assumption that the dynamics follows a time-continuous homogeneous Markov jump process [Ullah, Wolkenhauer 2011: Section 5.1] what can also be derived by physical theory as outlined by [Petzold, Gillespie 2006] and conducted in detail by [Gillespie 1992]. Here, these assumptions are taken as given. In the stochastic framework one no longer works with continuous concentrations as in ODE models but with the actual copy number of each species and reactions will change the discrete states (representing the copy numbers of each species) according to the stochiometric matrix of the reaction system under study.

More precisely, if one has a reaction system $(\mathcal{S},\mathcal{R})$ with species $\mathcal{S}=\{\mathcal{S}_1,...,\mathcal{S}_n\}$ and reactions $\mathcal{R}=\{\mathcal{R}_1,...,\mathcal{R}_m\}$, where $n\in\mathbb{N}$ is the number of species and $m\in\mathbb{N}$ the number of reactions in the system, the state of the system at time $t\in[0,\infty)$ is described by a discrete variable $X(t):=(X_1(t),...,X_n(t))^T\in\mathbb{N}_0^n$ where $X_i(t)\in\mathbb{N}_0$ represents the number of molecules of species i present in the system at time t. When $S=(s_{ij})_{\substack{i=1,...,n\\ j=1,...,m}}\in\mathbb{Z}^{n\times m}$ is the stochiometric matrix of the reaction system, the occurrence of the reaction $\mathcal{R}_j$ will shift the state from $X\in\mathbb{N}_0^n$ to $X+S_{*j}\in\mathbb{N}_0^n$ where $S_{*j}:=(S_{1j},...,S_{nj})^T\in\mathbb{N}_0^n$ is the j-th column of S.

Often it is assumed that all reactions are at most bimolecular, i.e. only involve at most two reaction partners. This assumption is referred to as the assumption of elemental *reactions*. This is often justified since real chemical reactions usually proceed by successive reactions which are at most bimolecular although the overall reaction may formally involve more than two reaction partners.

The decisive quantities in regard to the stochastic evolution of the system are the so called *propensities*. Let $\mathfrak{S}\subset\mathbb{N}_0^n$ denote the set of all possible states of the system. Based on theoretical physical considerations one can derive [Gillespie 1992] that for every reaction $\mathcal{R}_j$ there is a propensity function $a_j:\mathfrak{S}\to[0,\infty)$ such that $a_j(x)dt$ is the probability that given the system is in state $X(t)=x$ at time t, the reaction $\mathcal{R}_j$ will occur in the time interval $[t,t+dt)$ where dt is an "infinitesimal amount of time". Mathematically this is of course not a very strict statement but for the purposes of this thesis it is defined to be sufficient. Given a

reaction $\mathcal{R}_j\colon \mathcal{S}_i \to \mathcal{S}_k$ the propensity can be derived to be $a_j(x) = c_j x_i$ for some constant $c_j > 0$. For bimolecular reactions $\mathcal{R}_j\colon \mathcal{S}_i + \mathcal{S}_k \to \mathcal{S}_\ell$ with $i \neq k$ one has $a_j(x) = c_j x_i x_k$ for some constant $c_j > 0$ and for reactions $\mathcal{R}_j\colon \mathcal{S}_i + \mathcal{S}_i \to \mathcal{S}_k$ one gets $a_j(x) = c_j \frac{1}{2} x_i (x_i - 1)$ for some constant $c_j > 0$ [Petzold, Gillespie 2006]. The constants $c_j > 0$ are again called kinetic constants and can be precisely related by physical theory to the deterministic kinetic constants [Petzold, Gillespie 2006], [Ullah, Wolkenhauer 2011: Subsection 5.2.5].

Given a reaction system and the respective propensity functions one can derive the classical chemical master equation (CME). The CME specifies the time evolution of the probabilities $P(x,t|x_0,t_0) := \mathbb{P}(X(t) = x | X(t_0) = x_0)$ where $\mathbb{P}$ is the law of the Markov process governing the dynamics of the system and $x_0 \in \mathbb{N}_0^n$ is the specified initial value of the system at some time $t_0 \leq t$.

A semi-rigorous derivation of the CME based on [Petzold, Gillespie 2006] is now given. First, given a time infinitesimal dt, subjecting $P(x,t+dt|x_0,t_0)$ to the law of total probability gives the following result:

$$\begin{aligned} P(x,t+dt|x_0,t_0) &= \mathbb{P}(X(t) = x | X(t_0) = x_0)\, \mathbb{P}(\text{no reaction in } [t,t+dt) | X(t) = x) \\ &\quad + \sum_{j=1}^{m} \mathbb{P}(X(t) = x - S_{*j} | X(t_0) = x_0)\, \mathbb{P}(\mathcal{R}_j \text{ occurs in } [t,t+dt) | X(t) = x). \\ &= P(x,t|x_0,t_0)\left(1 - \sum_{j=1}^{m} a_j(x)\,dt\right) + \sum_{j=1}^{m} P(x - S_{*j}, t | x_0, t_0)\, a_j(x - S_{*j})\,dt \end{aligned}$$

Under the assumption that dt is "sufficiently infinitesimal" it is subsumed that no more than at most one reaction can take place in the interval $[t,t+dt)$ which justifies the above equations. In terms of the underlying Markov process this equation corresponds to the so called *Chapman-Kolmogorov equation* (just as well as the "final" CME).

By rearranging it follows

$$P(x,t+dt|x_0,t_0) = P(x,t|x_0,t_0) + \sum_{j=1}^{m} \left\{ P(x - S_{*j}, t | x_0, t_0)\, a_j(x - S_{*j}) - P(x,t|x_0,t_0)\, a_j(x) \right\} dt$$

and this finally corresponds to the following differential equation, the CME:

$$\frac{dP(x,t|x_0,t_0)}{dt} = \sum_{j=1}^{m} \left\{ P(x - S_{*j}, t | x_0, t_0)\, a_j(x - S_{*j}) - P(x,t|x_0,t_0)\, a_j(x) \right\} \quad \text{(CME)}$$

By interpreting $x \in \mathfrak{S}$ as an index the above equation system (*the* CME) describes the time evolution in terms of $|\mathfrak{S}|$ coupled ordinary differential equations.

Biochemical systems which evolve according to a Markov jump process as indicated above are said to be governed by *stochastic chemical kinetics* (SCK).

As can be expected, the CME is usually not analytically solvable. Only in the most trivial examples is an analytic treatment possible, for example in the case of a system with two reactions $\mathcal{S}_1 \to \mathcal{S}_2$ and $\mathcal{S}_2 \to \mathcal{S}_1$ under the constraint that $X_1(t) + X_2(t) = N \in \mathbb{N}$ for all $t \geq 0$. In that case it turns out that $P(x, t | x_0, t_0)$ is given by a binomial distribution [Ullah, Wolkenhauer 2011: Subsection 5.5.1]. In principle the CME could be solved by standard numerical methods for ODE systems but this turns out to be difficult because the number of equations $|\mathfrak{S}|$ is usually extremely large. As indicated in [Ullah, Wolkenhauer 2011: Section 5.1], specialized numerical approaches for solving the CME are for example (!) the *finite state projection algorithm* [Munsky, Kammash 2006], the so called *sliding windows method* [Wolf et al. 2010] or the method of conditional moments [Hasenauer et al. 2014]. Further methods try to apply *spectral methods* [Mugler et al. 2009], [Engblom 2009] or utilize *stochastic hybrid systems* [Henzinger et al. 2010].

A philosophically different but in essence equivalent approach is to sample realizations of the process itself. This is referred to as stochastic simulation and is shortly reviewed in the next subsection.

2.2.3 Exact stochastic simulation: the Gillespie algorithm

Exact stochastic simulation of stochastic chemical kinetics refers to the approach in which samples from the stochastic dynamics described by the CME are drawn. The basic algorithm which simulates the Markov jump process of stochastic chemical kinetics is *the Gillespie algorithm* (also called *stochastic simulation algorithm*, *SSA*) introduced by [Gillespie 1977]. A high-level description of the Gillespie algorithm can be given as follows. Given that the system is in a particular state the corresponding propensities can be evaluated and based on these propensities one samples the time and index of the next reaction, i.e. which reaction occurs next and when? The exposition in this subsection is again based on [Petzold, Gillespie 2006].

In order to derive the Gillespie algorithm one defines

$$p(\tau, j | x, t)\, d\tau := \mathbb{P}\left(\text{no reaction in } [t, t+\tau) \text{ and } \mathcal{R}_j \text{ occurs in } [t+\tau, t+\tau+d\tau) \,\middle|\, X(t) = x\right)$$

where $d\tau$ is again an infinitesimal time span such that only one reaction can occur in the interval $[t+\tau, t+\tau+d\tau)$. So, $p(\tau, j | x, t)$ is the probability that given that the system is in state x at time t the next reaction will be $\mathcal{R}_j$ and will happen at time $t+\tau$.

With $P_0(\tau|x,t) := \mathbb{P}(\text{no reaction in } [t,t+\tau) | X(t)=x)$ one has (with the definition of conditional expectation and by ignoring the infinitesimal) the following two identities:

$$p(\tau, j|x,t)d\tau = P_0(\tau|x,t)a_j(x)d\tau \qquad (2.2.1)$$

$$P_0(\tau+d\tau|x,t) = P_0(\tau|x,t)\left(1-\sum_{j=1}^{m} a_j(x)d\tau\right). \qquad (2.2.2)$$

The first one states that the probability that the next reaction occurs at $t+\tau$ and will be reaction $\mathcal{R}_j$ given that $X(t)=x$ can be obtained as the probability of no reaction occurring in $[t,t+\tau)$ given that $X(t)=x$ times the probability that reaction $\mathcal{R}_j$ (and only reaction $\mathcal{R}_j$!) occurs given $X(t+\tau)=x$. The second identity states that the probability of no reaction occurring in $[t,t+\tau+d\tau)$ given $X(t)=x$ can be written as the probability that no reaction occurs in $[t,t+\tau)$ given $X(t)=x$ times the probability of no reaction occurring in $[t+\tau,t+\tau+d\tau)$ given $X(t+\tau)=x$. Equation (2.2.2) corresponds to the following differential equation: $\frac{dP_0(\tau|x,t)}{d\tau} = \sum_{j=1}^{m} a_j(x)$.

This elementary equation has the solution $P_0(\tau|x,t) = \exp\left(-\tau\cdot\sum_{j=1}^{m} a_j(x)\right)$ (the initial value given by one since almost surely no reaction will occur in zero time). Substituting this expression into equation (2.2.1) (and "dividing" by $d\tau$) finally gives the following result:

$$p(\tau, j|x,t) = a_j(x)\exp\left(-\tau\cdot\sum_{j=1}^{m} a_j(x)\right) = \frac{a_j(x)}{a_0(x)}\cdot a_0(x)\exp(a_0(x)\tau) \qquad \text{(Gillespie)}$$

In the above equation it is defined $a_0(x) := \sum_{j=1}^{m} a_j(x)$. The identity (Gillespie) is the basis of the Gillespie algorithm. It contains the fact that the time until the next reaction and the type of the next reaction are independent. Furthermore $a_0(x)\exp(a_0(x)\tau)$ can be identified as the density of an exponential distribution with parameter $a_0(x)$ and hence the time until the next reaction is given by a random variable $(T|X(t)=x) \sim \text{Exp}(a_0(x))$ while the type of the reaction is determined by a categorical distribution over the reactions with weight $\frac{a_j(x)}{a_0(x)}$ for reaction $\mathcal{R}_j$. These considerations then lead to the following theoretical basic form of the Gillespie algorithm:

ALGORITHM 2.1 **[Gillespie algorithm]**

Input: I1. $(\mathcal{S},\mathcal{R})$ with stochiometric matrix S # reaction system

I2. $a_j : \mathfrak{S} \to [0,\infty)$ # propensity functions

I3. $x_0 \in \mathfrak{S}$, $t_0 \in [0,\infty)$ # initial state and start time

I4. $t_{max} \in [t_0,\infty)$ # maximal simulation time

Output: O1. $X : [t_0, t_{max}] \to \mathbb{N}_0^n$ # sample of the stochastic kinetics defined by $(\mathcal{S},\mathcal{R})$

---

(1) $X(t_0) \leftarrow x_0$, $t \leftarrow t_0$, $x \leftarrow x_0$ # initialization

(2) while ($t < t_{max}$)

(2.1) compute $a_j(x)$, $j = 1,...,m$ # update propensities

(2.2) $a_0(x) \leftarrow \sum_{j=1}^{m} a_j(x)$

(2.3) sample $\tau \sim \mathrm{Exp}(a_0(x))$ # sample time to next reaction

(2.4) sample $k \sim \mathrm{Cat}\left(\frac{a_j(x)}{a_0(x)}\right)$ # sample the type of the next reaction

(2.5) if $t + \tau \le t_{max}$

(2.5.1) $X(t') \leftarrow x$ for $t' \in (t, t+\tau)$

(2.5.2) $X(t+\tau) \leftarrow x + S_{*k}$, $x \leftarrow x + S_{*k}$ # update of state

(2.5.3) $t \leftarrow t + \tau$ # update of time

(2.6) else

(2.6.1) $X(t') \leftarrow x$ for $t' \in (t, t_{max}]$

For a quick summary on how to simulate random variables in general and exponential and categorical variables in particular it is referred to Appendix A.2.

The Gillespie algorithm, i.e. stochastic simulation, is equivalent to the CME in the sense that it generates i.i.d. samples from the process described by the CME. In principle one could solve the CME by infinitely often simulating with the Gillespie algorithm for an infinite amount of time respectively.

For larger biochemical systems exact stochastic simulation is not feasible anymore. Although there exist adapted exact simulation schemes like the *next reaction method* [Gibson, Bruck 2000] which are faster for some systems, it becomes inevitably inefficient to exactly simulate

stochastic chemical kinetics if the systems are more complex. In such a situation one has to resort to approximate simulation schemes like the one discussed in the next subsection.

2.2.4 Approximate stochastic simulation: τ-leaping

The approximate stochastic simulation algorithm called *τ-leaping* was introduced by [Gillespie 2001]. Instead of simulating every single reaction event it simulates a whole bunch of reactions which happen in time intervals of fixed length $\tau > 0$ respectively and simulates the state of the system accordingly at multiples of τ. τ is called *leap size* and is a hyperparameter of the algorithm which has to satisfy a so called *leap condition*. The leap condition describes the assumption that the propensities do not change significantly if the system is evolved according to the leap size.

More precisely the τ-leaping algorithm is based on the fact that the reactions have exponentially distributed waiting times with parameter $a_j(x)$ for their respective next occurrences given. This can be seen as follows. Define

$$p_j(\tau|x,t)d\tau := \mathbb{P}\left(\mathcal{R}_j \text{ does not occur in } [t,t+\tau) \text{ but } \mathcal{R}_j \text{ occurs in } [t+\tau, t+\tau+d\tau) \middle| X(t)=x\right).$$

With $P_0^j(\tau|x,t) := \mathbb{P}\left(\mathcal{R}_j \text{ does not occur in } [t,t+\tau) \middle| X(t)=x\right)$ one has

$$p_j(\tau|x,t)d\tau = P_0^j(\tau|x,t)a_j(x)d\tau \qquad (2.2.3)$$

$$P_o^j(\tau+d\tau|x,t) = P_0^j(\tau|x,t)\left(1-a_j(x)d\tau\right) \qquad (2.2.4)$$

by the definition of conditional probability. Deriving the canonical ODE from (2.2.4) leads to $P_0^j(\tau|x,t) = \exp\left(-a_j(x)\tau\right)$ and the substitution of this formula into 2.2.3 finally gives

$$p_j(\tau|x,t) = a_j(x)\exp\left(-a_j(x)\tau\right).$$

Since $p_j(\tau|x,t)$ is the density of the random variable describing the time to the next occurrence of reaction $\mathcal{R}_j$ one can conclude that this time is exponentially distributed with parameter $a_j(x)$.

Based on that, for τ-leaping to work it is further assumed that for each reaction $\mathcal{R}_j$ the number of reactions in a suitable small interval of length τ given $X(t)=x$ follows a Poisson distribution with parameter $a_j(x)\tau$ and that these numbers are independent. This can be satisfied if the leap condition is satisfied: Given $X(t)=x$ the propensities do not change "significantly" during a τ-leaping step of the following form:

$$X(t+\tau) = x + \sum_{j=1}^{m} Y_j S_{*j} \quad \text{with } Y_j \sim \text{Poi}\left(a_j(x)\tau\right) \qquad (\tau\text{-leaping})$$

The above formula is the basis of τ-leaping. For this algorithm to work it is necessary to be able to choose a leap size which satisfies the leaping condition, see for example [Gillespie, Petzold 2003]. Given a suitable leap size and $X(t) = x$ the algorithm simulates the number of reactions $\mathcal{R}_j$ during the interval $[t, t+\tau)$ as $Y_j \sim \text{Poi}\left(a_j(x)\tau\right)$ and the system of the state at time $t+\tau$ is then updated by adding the respective numbers of columns of the stochiometric matrix to the old state x.

τ-leaping is an approximation algorithm and hence does not simulate stochastic chemical kinetics exactly. However, if $\tau \approx 1/a_0(x)$ the next update will be as good as exact. If the leaping condition can only be satisfied such that this relationship holds it is advisable to use the Gillespie algorithm instead [Petzold, Gillespie 2006]. Another issue with τ-leaping is that one has to take care that the state of the system does not become negative.

So far the states describing the system were discrete and given by the respective molecule numbers. One further step can be to again postulate continuous concentration levels instead but to keep the stochasticity. This is shown in the next subsection.

2.2.5 Diffusion approximation: the chemical Langevin equation (CLE)

If in addition to the leap condition one has $a_j(x)\tau \gg 1$ for all reactions $\mathcal{R}_j$ one can utilize the fact that if $\lambda \to \infty$ one has $\frac{\lambda^k}{k!} e^{-\lambda} \approx \frac{1}{\sqrt{2\pi\lambda}} e^{-\frac{(k-\lambda)^2}{2\lambda}}$ and one therefore can draw the "number" of occurrences of reaction $\mathcal{R}_j$ as a normal random variable with mean and variance $a_j(x)\tau$. Note that the leaping condition and $a_j(x)\tau \gg 1$ constitute a „small but large" condition. Assuming both are satisfied one can then introduce the so called Langevin leaping formula for the state of the system at time $t+\tau$ given $X(t) = x$:

$$X(t+\tau) = x + \sum_{j=1}^{m} S_{*j} a_j(x)\tau + \sum_{j=1}^{m} S_{*j} \sqrt{a_j(x)\tau}\, Z_j \quad \text{with } Z_j \sim_{\text{iid}} \mathcal{N}(0,1).$$

This corresponds to $X(t+\tau) = x + \sum_{j=1}^{m} S_{*j} \widehat{Z}_j$ with $\widehat{Z}_j \sim_{\text{independent}} \mathcal{N}\left(a_j(x), a_j(x)\right)$.

The Langevin leaping formula can be interpreted as a discretization of the following stochastic differential equation:

$$dX(t) = \sum_{j=1}^{m} S_{*j} a_j(x)\,dt + \sum_{j=1}^{m} S_{*j} \sqrt{a_j(x)\tau}\; dB_j(t) \qquad \text{(CLE I)}.$$

This equation is known as the chemical Langevin equation (CLE) [Gillespie 2000]. Here, $B(t) = (B_1(t), ..., B_m(t))^T$ is m-dimensional standard Brownian motion (see [Oksendal 2010: Section 2.2]). The state of the system over time is now described as an n-dimensional continuous stochastic process given by the solution $X(t) = (X_1(t), ..., X_n(t))^T$ of (CLE I). The given derivation of the CLE is of course somehow rough since at first τ has to be small but not too small and then it finally goes to zero.

By setting $a(x) := (a_1(x), ..., a_m(x))^T \in \mathbb{R}^m$, $A(x) := Sa(x)$ and $D(x) := S \cdot \mathrm{diag}(a(x))^{1/2}$ one can also write the Langevin equation as $dX(t) = A(X(t))dt + D(X(t))dB(t)$ [Ullah, Wolkenhauer 2011: Subsection 5.4.2]. As also shown in [Ullah, Wolkenhauer 2011: Subsection 5.4.2] one can reformulate as $dX(t) = A(X(t))dt + [\Delta(X(t))]^{1/2} d\bar{B}(t)$ with $\Delta(x) := D(x)D(x)^T$ and n-dimensional standard Brownian motion $\bar{B}(t)$.

The CLE is often termed a *diffusion approximation* of stochastic chemical kinetics (i.e. the respective Markov jump process). There are other such approximations, like for example the Fokker-Planck equation (which is actually equivalent to the Langevin equation) [Fuchs 2013: Chapter 4].

From the CLE one can finally obtain the deterministic RRE systems by a transition termed *thermodynamical limit*, see [Petzold, Gillespie 2006].

2.2.6 Parameter estimation for stochastic biochemical models

As in ODE modeling, SCK models ask for appropriate parameterizations. One of the main differences to the ODE case is that in stochastic models the uncertainty not only comes from measurement errors but also naturally from the model itself. [Wilkinson 2009] is a very good overview on stochastic modeling in general and parameter estimation for stochastic models in particular. Roughly, one can divide the approaches according to whether they try to estimate parameters for discrete stochastic models (i.e. pure stochastic chemical kinetics, i.e. the Markov jump process) [Boys et al. 2008] or do so for the respective diffusion approximations [Golightly, Wilkinson 2005], [Fuchs 2013]. The latter is considered easier but it has to be pointed out that it is still difficult. For further details, see [Wilkinson 2009] and the references therein.

2.3 Spatial inhomogeneity: PDEs and the RD-CME

By dropping the assumption of spatial homogeneity, one is urged to make the transition from ODE models to models involving partial differential equations (PDEs) and take into account the spatial dependencies intrinsic to the processes in question [De Jong 2002], [Kruse, Elf; 2010], [Murray 2004: Chapters 11, 12, 13; 2008], [Edelstein-Keshet 2005: Chapters 9, 10, 11]. This becomes important for example if some reactions only take place at particular sites (some specific cellular organelles for example) in the cell or if one wants to model the transport of molecules (through space) explicitly and so on and so forth.

One particular type of PDE model is *reaction-diffusion models*. In its simplest form (involving only two chemical species) a reaction-diffusion equation (for two chemical species) can be written in the following basic form:

$$\partial_t x_1 = D_1 \Delta_r x_1 + f_1(x_1, x_2)$$
$$\partial_t x_2 = D_2 \Delta_r x_2 + f_2(x_1, x_2).$$

Here $x_1 = x_1(r,t)$, $x_2 = x_2(r,t) \geq 0$ are the concentrations of the involved species which are now space-dependent as indicated through $r \in \mathbb{R}^d$, $d \in \mathbb{N}$. The above equations are a description of the space-time-evolution in terms of *diffusion terms* $D_i \Delta_r x_i$ and *reaction terms* $f_i(x_1, x_2)$ describing the interaction between the species. $D_i > 0$ is the *diffusion constant* of species i.

All issues concerning the analysis of ODE models apply with even more peculiarity in the case of PDE models: dependence on numerical methods, qualitative techniques and the difficulty in parameterizing the system based on experimental data.

By combining spatial inhomogeneity with stochastic chemical kinetics one arrives at the so called *reaction diffusion chemical master equation* (RD-CME), see [Kruse, Elf 2006].

3 From discrete Boolean networks to stochastic continuous models for biochemical networks

This chapter is dedicated to the topic of discrete Boolean models and the attempts to reintroduce continuity in time or space and the possibilities to incorporate various forms of stochasticity in these models. There are many good review articles on Boolean models for biochemical networks: [Bornholdt 2008], [Albert, Wang 2009], [Morris et al. 2010], [Glass, Siegelmann 2010], [Wang et al. 2012], [Albert et al. 2013], [Chaouiya, Remy 2013]. In Section 3.1 there will be some general definitions and considerations concerning the motivation, justification and methodology of Boolean models for biochemical systems. In particular, the issues of *model representation*, *updating schemes* and *attractors* are considered. In Section 3.2 I shortly mention the classical *ensemble approach* of Stuart Kauffman [Kauffman 1969] by defining the classical notion of *random Boolean networks* (RBNs). Extensions are outlined and some studies addressing the question of so called *biologically meaningful update rules* are mentioned. In Section 3.3, *probabilistic Boolean networks* (PBN) are briefly defined and their applications indicated [Shmulevich et al. 2002a, 2002b]. In Section 3.4 several further approaches to introduce stochasticity into Boolean models are discussed. Also, first examples for approaches which incorporate continuous time into an otherwise Boolean framework are given. In Section 3.5 the basic notions concerning the time-continuous *generalized kinetic logic* (GKL) networks (Thomas formalism) [Thomas, D'Ari 1990] are defined. The dynamics of GKL networks are given by species specific time delays which can be either deterministic or stochastic [Thomas 1979]. The stochastic version of GKL is also the topic of Section 4.1. Section 3.6 deals with *piecewise linear differential equation* (PLDE) models [Glass, Kauffman 1973] which are continuous in time and space but which are based on an underlying Boolean dynamics. The connections between GKL or other kinds of discrete models and PLDE models are an active area of research [Jamshidi et al. 2012], [Farcot 2006]. The general rationale for setting up correspondences between discrete and continuous models is of course the hope of simplifying the continuous models via their discrete counterparts or vice versa enriching the dynamics of the discrete models via their continuous counterparts in some meaningful or at least in some abstract sense useful way. Later, in Section 4.2 I will shortly look at a simple example of a stochastic version of PLDE models and outline the topic of how to relate this kind of model to GKL models with stochastic time delays. Section 3.7 shortly introduces *Petri net models* of biochemical networks. In Section 3.8 three further approaches which re-inject continuous states into Boolean models are outlined: *fuzzy logic*, *standardized qualitative dynamical systems* (SQUAD) and *multivariate polynomial interpolation* (Odefy). In Section 3.9 finally some *specific examples of Boolean modeling* are given as illustration. The spectrum of biological systems modeled with Boolean frameworks is extremely large and for many systems there are several Boolean models. See for example the review articles cited above and especially the reviews [Fauré, Thieffry 2009] and [Mbodj et al. 2013] on the various Boolean models for the cell cycle (of various organisms) and fruit fly signaling pathways respectively. Thus, Section 3.9 concentrates on models for *apoptosis* (*programmed cell death*) and exemplifies some

methodologies involved in Boolean modeling. This chapter is a coarse-grained overview. Omissions are by coincidence and do not constitute any kind of judgment.

3.1 Boolean models: representation, update regimes, attractors

Boolean or logical models were introduced to biology through the work of [Kauffman 1969] and are 'simple' models of biochemical systems where the state of every biochemical species (variables, nodes) in the system only has a finite set of possible states. In the classical setup these states are 0, corresponding to "gene off", "protein not expressed", etc. and 1, corresponding to "gene on", "protein expressed", etc.. Given this kind of simplification, one specifies the interactions of the species through so called *logic functions* (transfer-, update-, transition-, Boolean functions), for example as AND-gates defined via the property that some target gene is on if and only if its, say, two regulators are also on. Having this kind of discretization of the states of the system ("on", "off") and a network of logical interactions between the involved variables one has to decide on the so called *update scheme* (update regime, -schedule). In the classical setup, time is discretized and all variables are updated according to their logical interactions simultaneously: synchronous updating. Further approaches update the variables one at a time according to some deterministic or stochastic scheme. Of course the dynamics of the network crucially depends on the update scheme. A further characteristic property studied in relation to Boolean networks is their so called *attractor structure*. An attractor is a set of states which describes a possible long term behavior of the system, for example a point in the state space which is a fixed point of the update function and hence once in that state the system will remain there forever. Further canonical examples of attractors are periodic limit cycles, i.e. sequences of points in the state space that repeat over and over again once the system evolves to one of its defining states.

The justification of Boolean modeling stems from the observation that many regulatory systems seem to be based on switch-like interactions. One can think for example of Hill-type kinetics with large Hill exponents, i.e. high degrees of cooperativity (Subsection 2.1.1), or the actually measured gene regulation functions, for example the one shown in figure 1.6 in Chapter 1 from [Setty et al. 2003], which also often show a step-like character. In summary, if the underlying dynamics is defined by 'sufficiently' step-like interactions one can hope to capture the essential features of such a system in terms of Boolean models. There are studies which address these questions, see for example [Wittmann et al. 2010] or [Macía et al. 2009].

An approximately general definition of a Boolean network and the formal starting point for this section can be framed as follows.

Definition 3.1.1 (**Boolean Network**)

A *Boolean network* is a tuple (V, f) with *species* (*nodes, vertices, variables)* V given by a non-empty finite set, w.l.o.g. $V = [n]$, and a *logic function* $f : \{0,1\}^n \to \{0,1\}^n$ where $n \in \mathbb{N}$ is the number of species in the system.

For a state $x \in \{0,1\}^n$, the state $y := f(x) \in \{0,1\}^n$ is called the *successor* (state) of x. There are several variations concerning definition 3.1.1. One can study systems with more than just two possible qualitative states and accordingly define the logic function to be $f : \prod_{i=1}^{n} \{0,1,...,k_i\} \to \prod_{i=1}^{n} \{0,1,...,k_i\}$ where $k_i \in \mathbb{N}$ is the number of qualitative states for variable $i \in [n]$. The classical case corresponds to $k_i = 1$ for all i. Furthermore, although logically not necessary, the definition of a Boolean model is often based on an underlying interaction graph $G = (V, E)$ with the species $V = [n]$ and 'regulatory' interactions encoded in the directed edges E. The logic function is then specified component-wise by $f_i : \prod_{j \in pa(i)} \{0,1\} \to \{0,1\}$ for every $i \in [n]$ with $pa(i) := \{j \in V : (j,i) \in E\}$, i.e. the interaction graph already encodes which variables are non-irrelevant for the logic function f_i. One further possibility is given by the so called *logical interaction hypergraph* [Klamt et al. 2007] (see also Subsection 1.5.1) which can be seen as the other extreme to definition 3.1.1 in the sense that it already encodes the entire logic function in term of a graph structure. Formally, every Boolean function $f_i : \prod_{j \in pa(i)} \{0,1\} \to \{0,1\}$ can be written in *sum-product form* (*disjunctive normal form*, [Aigner 2006: Section 11.2], [Wittmann et al. 2009a]) as follows:

$$f_i(x_{i1},...,x_{ik_i}) = \vee_{\xi_1,...,\xi_{k_i} \in \{0,1\}^{k_i} \mid f_i(\xi_1,...,\xi_{k_i})=1} \left[\left(\wedge_{j:\xi_j=1} x_{ij} \right) \wedge \left(\wedge_{j:\xi_j=0} \neg x_{ij} \right) \right].$$

Here, $k_i := |pa(i)|$ and $x_{i1},...,x_{ik_i} \in \{0,1\}$ denote the states of the parents of i. Based on this representation the logical interaction hypergraph is then defined as a weighted directed hypergraph $H = (V, A)$ (see Subsection 1.5.1) with $V = [n]$ being the set of species and $a \in A$ if and only if $a = (pa(i), s, \{i\})$[6] for $i \in V$ and $s : pa(i) \to \{-1, +1\}$ such that there is $\xi = (\xi_1,...,\xi_{k_i}) \in \{0,1\}^{k_i}$ with $f_i(\xi) = 1$ and $s(j) = +1$ if and only if $\xi_j = 1$. Intuitively the logical interaction hypergraph thus contains an edge for every summand in the logical sum of products form such that the tails are the respective parents, the head is the regulated species and the weights on the parents are such that they represent the on-off relations of the parents'

[6] $pa(i)$ is the parent set associated to the 'normal' interaction graph; one could define the logical interaction graph also without an underlying 'normal' interaction graph just "in terms" of the Boolean functions but the idea remains the same.

states that give rise to $f_i\left(x_{i1},...,x_{ik_i}\right)=1$. Note that for every product in the sum-product representation there is one hyperedge from the respective parents to the respective head species.

Once the interaction structure and the Boolean logic functions are fixed, the next decision concerns the choice of a so called update schedule (-regime, -scheme). Given a Boolean network (V,f), I follow [Aracena et al. 2009] and define a *deterministic update schedule* to be a function $s:V=\{1,...,n\}\to\{1,...,m\}$ with $1\le m\le n$. The intuitive interpretation of the update schedule s is that species i is updated before species j if and only if $s(i)<s(j)$. Formally, the *instantaneous dynamical behavior* of the Boolean network (V,f) with deterministic update schedule s can then be framed as follows. Given a starting point $x^0\in\{0,1\}^n$ the trajectory of x^0 is given by the sequence $\left(x^t\right)_t\in\left(\{0,1\}^n\right)^{\mathbb{N}_0}$ of global 0-1-states such that for every $i=1,...,n$ one has $x_i^{t+1}=f_i\left(x_1^{\ell_1^i(t)},...,x_n^{\ell_n^i(t)}\right)$ with $\ell_j^i(t)=t$ if $s(i)\le s(j)$ and $\ell_j^i(t)=t+1$ if $s(i)>s(j)$ for any $j=1,...,n$.

The classical *synchronous update schedule* [Kauffman 1969] is given by $s\equiv 1$ with $m=1$. Other approaches include *block-sequential updating* where $s\equiv\pi$ and $m=n$ for a permutation $\pi\in S_n$[7] of the species or block sequential updating given by a partition $V=S_1\dot\cup\cdots\dot\cup S_m$ of the species such that $s(i)=\sum_{\alpha=1}^m\alpha\mathbb{I}(i\in S_\alpha)$ [Aracena et al. 2009]. The last two update regimes concern the internal structure of the update within one time step when used to define the *instantaneous dynamical behavior* as given above ('instantaneous' in the sense that everything happens during a single time step). A different situation is given by the asynchronous update schedules [Harvey, Bossomaier 1997] which involve the update of only one species during every time step. This can happen according to a prespecified update order such that a particular species is updated every third, say, time step while another one is updated only at every fifth step and so on. The above block-sequential update regime can also easily be interpreted in a globally asynchronous fashion by updating species in S_α at every α-th time step. [Gershenson 2002] describes the following update schedule. For each vertex $i\in V$ we have update parameters $Q_i,P_i\in\mathbb{N},\ P_i>Q_i$ and in time step $t\in\mathbb{N}$ vertex i will be updated if and only if $t=Q_i\left(\mathrm{mod}\,P_i\right)$. If more than one vertex has to be updated at a given time step one can either update them all synchronously (semi-asynchronous update) or do so in an arbitrary (i.e. random) order. In general, randomness can be variously incorporated for the introduced update schedules. For example, in the sequential update schedule $s\equiv\pi$ a permutation could be chosen according to a density over the symmetric group while in the block-sequential case a partition could be chosen randomly according to some suitable probability law. The so called *random order asynchronous update* involves choosing a vertex uniformly at every step [Harvey, Bossomeier 1997]. In Sections 3.5 and 4.1 I deal with so

[7] S_n, the *symmetric group* of order n, see for example [Karpfinger, Meyberg 2010: Chapter 9]

called generalized kinetic logic (GKL) models which are logical models following update rules defined by deterministic or stochastic time delays associated to the up- or down-regulation of specific genes [Thomas, D'Ari 1990]. Asynchronous updates where the asynchronicity stems from more general stochastic processes have also been considered [Deng et al. 2007].

Mathematically speaking there seems to be an almost infinite multitude of possible update regimes and the question what kind of changes in dynamic behavior can arise from changes to the update regime is certainly crucial as well as difficult in full generality. See for example [Aracena et al. 2009]. In terms of biology however, the update schedules can be thought to represent the characteristic timescales on which the respective processes typically act. If, for example, a species is updated at every second step while another is updated only every fourth step, this can be interpreted as the latter species having a typical time scale which is twice that of the first species. Therefore, random order asynchronous update schedules are biologically implausible since they would basically imply the completely random mixing of relevant timescales during the dynamical evolution of a biological system. They can however be useful for exploring the set of possible behaviours of a given Boolean network, see for example [Álvarez-Buylla et al. 2008]. In contrast, the classical synchronous update regime seems also not generally appropriate since it leaves out the possibility of differing timescales [Harvey, Bossomeier 1997]. [Chaves et al. 2008] compare different update schedules in a Boolean model of Drosophila segment polarity gene regulation. Parts of the main part of this thesis (especially Section 4.1) will be concerned with appropriate choice of stochastic asynchronicity in the framework of GKL models.

Another approach relating to update schedules and timescales is the one taken for example in [Saez-Rodriguez et al. 2007] or [Schlatter et al. 2009]. There, several processes which are modeled with a Boolean model respectively are known to act on different time scales, i.e. regulatory influence A is active right from the start while regulatory influence B becomes active only after some time delay. This knowledge was incorporated a priori into the study of the respective Boolean models such that the model is successively 'updated' according to which regulatory influences are known to be active at a particular time step and hence one has in effect different Boolean models at different time steps which can then be analyzed in succession. I refer to this approach in following as the *timescale approach*.

One last very important aspect of Boolean modeling has to be covered in this section, namely the notion of *attractors*. In terms of gene regulatory networks, attractors are thought to represent different functional or developmental states of cells. In the case of a Boolean model of the cell cycle[8] for example, different functional states (i.e. proliferation state or resting phase, see [Munk et al. 2008: Chapter 12]) are represented by specific global 0-1, i.e. Boolean states of the involved species. These functional states are generically transient and reversible. Developmental states on the other hand relate to the decision of cells which developmental pathway to pursue and this decision is often irreversible. One such decision is the fate decision of λ-phages described in Subsection 1.3.6. In this case the decision is reversible, but in case of blood cell formation (hematopoeisis) for example, a classical paradigm system for

[8] see [Davidich, Bornholdt 2008a] for a specific example and [Fauré, Thieffry 2009] for a review.

stem cell differentiation, the development into different specialized cell types of the blood system is largely irreversible [Miranda-Saavedra, Göttgens 2008]. Again, the different cell types or fates are described by characteristic Boolean states of the involved genes and species.

The simplest kind of attractor is a *fixed point* (or *steady state*) given by a state $\bar{x} \in \{0,1\}^n$ such that $f(\bar{x}) = \bar{x}$, i.e. regardless on which deterministic or stochastic update regime is used (!) the system will always stay in that state once it is reached (as long as there are no noisy effects which may switch node states randomly, see Section 3.4). For deterministic update regimes *periodic limit cycles* of length $k \in \mathbb{N}$ are a further class of attractors which are defined as a repeating sequence of states $(x_\nu)_\nu \in \left(\{0,1\}^n\right)^k$ with k minimally chosen such that if one starts in state x_ν for one $\nu = 1,...,k$ after k update steps (conducted under a certain deterministic (!) update regime) the system is again in that same state x_ν. The *basin of attraction* of a given attractor is defined as the set of states which lead to this attractor. The size of the basin of attraction of a biologically meaningful attractor in a given model (i.e. an attractor representing a functional or developmental state) can be used to evaluate the appropriateness of the model in the sense that the system should have "a tendency" to reach that attractor) [Davidich, Bornholdt 2008a].

In the case of probabilistic update schedules the definition of attractors is not so straightforward anymore. While fixed points can be defined in the same way, the definition of limit cycles becomes difficult due to the fact that from a given state $x \in \{0,1\}^n$ there may now be several probabilistic possibilities for the successor state according to the stochastic update involved and hence the 'limit cycles' may 'branch out'. [Harvey, Bossomeier 1997] defined so called *loose attractors* which basically correspond to strongly connected components of the underlying state transition graph (see Section 3.5 for the definition of the state transition graph and of 'loose attractors' in the context of GKL networks). More on attractors under random asynchronous update regimes can be found in [Saadatpour et al. 2010].

There is certainly more that could be said about the general framework of Boolean models but I leave it at that.

3.2 Random Boolean networks (RBNs): The ensemble approach

As already shortly mentioned in the preceding section, the historical starting point for Boolean modeling of biochemical networks was the work of S. Kauffman in [Kauffman 1969]. In these days the data on bioregulatory systems necessary to set up even coarse-grained models like Boolean networks was seldomly available and so the original aim of Kauffman's paper was not so much to setup models for specific biochemical systems but to introduce a method to examine general design principles for the entirety of such networks.

With this aim in mind he defined random Boolean networks (RBNs). Formally, RBNs are random variables such that their range comprises Boolean networks with specified properties, see for example [Wittmann 2010: Chapter 3] or [Gershenson 2004]. For example, in Kauffman's original publication he considered networks where every vertex has $K \in \mathbb{N}$ uniformly chosen parents and the logic function of every vertex is also chosen uniformly over all possible Boolean functions. Then he examined the statistical properties of these randomly obtained networks, for example their average attractor number or their average attractor length. These classical RBNs showed 'ordered' behavior for $K = 1$ (perturbations do not spread "very much") and chaotic behavior for $K > 2$ (perturbations spread "widely"). Networks with $K = 2$ he identified to be located "on the edge of chaos" and claimed that a plausible property of real-world biochemical networks could be that they are located on this edge of chaos, because in that case, on the one side perturbations should not spread too heavily (which would result in the non-functionality of the network) and on the other side the network also would not be too insensitive to perturbations what would be advantageous in terms of the potential of the system to evolve new beneficial traits according to natural selection.

There have been many developments of the RBN approach since 1969, for example [Wittmann et al. 2010] considered multi-valued Kauffman networks. [Mesot, Teuscher 2003] considered networks with asynchronous updating. The described approach is also often called the *ensemble approach* and is described in detail in [Kauffman 2000] or [Aldana et al. 2003].

3.2.1 Biologically meaningful update rules

It was noticed that not all logic functions are equally plausible for biological systems and hence it would be interesting to study ensembles of RBNs which only have certain so called *biologically meaningful update rules*. For example it was recognized that so called *canalizing functions* are often part of biochemical networks. A canalizing function is a logic function such that there is one input variable which determines the output completely when it is in one state and only if it is in the other state do the other variables become relevant [Harris 2002], [Kauffman et al. 2003], [Kauffman et al. 2004]. Other works dealing with biologically meaningful update rules are [Raeymaekers 2002], [Nikolajewa 2006] or [Wittmann et al. 2010].

A further special case of Boolean logic functions are the so called *threshold functions* defined by

$$f_i(x) = 1 \quad \Leftrightarrow \quad \sum_{j \in pa(i)} \kappa_j x_j > \theta_i$$

where $x \in \{0,1\}^n$, constants (weights) $\kappa_j \in \mathbb{R}$ and $\theta_i \in \mathbb{R}$ being the so called threshold constant. Boolean networks where the logic functions are all threshold functions of the above form are called *threshold Boolean networks* (TBNs) and were studied for example in [Rohlf, Bornholdt 2002] or [Szejka et al. 2008].

3.3 Probabilistic Boolean Networks (PBNs)

One further approach introduced by [Shmulevich et al. 2002a], [Shmulevich et al. 2002b] are probabilistic Boolean networks (PBNs). PBNs introduce stochasticity into the Boolean framework at the level of an existing Boolean network, in contrast to the ensemble approach where the stochasticity comes into play at a higher level. In the basic PBN setup, the connectivities of the network are assumed to be known and stochasticity enters in terms of uncertainty concerning the update functions:

[Shmulevich, Dougherty 2010] define for a network with N species the so called *gene activity profiles* $s \in Q := \{0,1,...,d-1\}^N = \prod_{i=1}^{N} D$ as elements of a multi-valued state space. Further, instead of just one logic function $f : Q \to Q$, a PBN can have several update 'contexts' $f_\beta : Q \to Q$, $\beta = 1,...,K \in \mathbb{N}$, where $f_\beta^{(i)} : Q \to D = \{0,1,...,d-1\}$ is the update function of species i if context β (out of K) is 'chosen'. The contexts represent uncertainty about the logic functions. The probabilistic dynamics is now defined as follows. For a given categorical random variable $C \sim \mathrm{Cat}(c)$, $c = (c_1,...,c_K)$ (corresponding to the "law" of the uncertainty), the time evolution $(s_n) \in Q^{\mathbb{N}}$ is given by a realization of the stochastic process (S_n) with $\forall j = 1,...,K,\ n \in \mathbb{N}: \mathbb{P}(S_n = f_j(s') | S_{n-1} = s') = c_j$, $s' \in S$. So, in every update step one of the K contexts is randomly chosen according to a probability vector c and the update then proceeds according to the chosen logic function (context). The described process (S_n) constitutes a Markov chain [Norris 1998] (see also Chapter 4) with transition probabilities

$$p(s',s) := \mathbb{P}(S_{n+1} = s | S_n = s') = \sum_{j=1}^{K} \delta_{f_j(s')}(s) c_j .$$

The above described PBNs are also termed instantaneously random PBNs because at every update step a new function (which may be the old one) is chosen. An extension are the so called *context-sensitive PBNs* which, as the name suggests, involve the (maybe more persistent) choice of one of the contexts $f_\beta : Q \to Q$. A context, as indicated above, is nothing else than a particular choice of $\beta = 1,...,K \in \mathbb{N}$. For instantaneously random PBNs this choice is made at every update step. Context-sensitive PBNs now involve a probability $q \in [0,1]$ concerning the decision whether to choose a new context or not, i.e. a new context (which may also be the old one) is chosen when a Bernoulli random variable realizes to one while the old context is used to update the system if the Bernoulli variable realizes to zero. The resulting dynamics is again a time-homogeneous, time-discrete Markov chain (Z_n) but in contrast to above, the state space is now a combination of the context and the 'state' of the network, i.e. $Z_n = (\kappa_n, S_n) \in \{1,...,K\} \times S$. Concerning the transition probabilities one obtains:

$$
\begin{aligned}
p\big((\kappa,s),(\kappa',s')\big) &:= P\Big(Z_{n+1}=(\kappa,s)\Big|Z_n=(\kappa',s')\Big)\\
&= P\Big(Z_{n+1}=(\kappa,s)\Big|Z_n=(\kappa',s'),\kappa_{n+1}=\kappa'\Big)P\big(\kappa_{n+1}=\kappa'\big)\\
&\quad + P\Big(Z_{n+1}=(\kappa,s)\Big|Z_n=(\kappa',s'),\kappa_{n+1}\neq\kappa'\Big)P\big(\kappa_{n+1}\neq\kappa'\big)\\
&= \delta_{f_{\kappa'}(s')}(s)\big(1-q+qc_{\kappa'}\big)+\sum_{\kappa\neq\kappa'}\delta_{f_\kappa(s')}(s)\,qc_\kappa
\end{aligned}
$$

$$
\Rightarrow\quad p\big((\kappa,s),(\kappa',s')\big)=\delta_{f_{\kappa'}(s')}(s)\big(1-q\big)+q\cdot\sum_{\kappa=1}^{K}\delta_{f_\kappa(s')}(s)\,c_\kappa
$$

There are more refined versions of PBNs, for example one can take into account perturbations of the states s_n or one could choose contexts in such a way that the probability of choosing a new one depends on the present one. For further details, see [Shmulevich, Dougherty 2010].

The PBN literature is mainly concerned with inference of the model structure based on experimental data, see [Shmulevich, Dougherty 2010: Chapter 3] and the references therein. Also, control and intervention theory for PBNs, see [Shmulevich, Dougherty 2010: Chapters 4 and 5] and the references therein, is a highly active field. Asynchronous PBNs also have been proposed, [Shmulevich, Dougherty 2010: Chapter 6] and the references therein.

For a connection between PBNs and Bayesian networks, see [Lähdesmähki et al. 2006].

3.4 Stochasticity and continuous time

This section is dedicated to several approaches which also incorporate stochasticity into a Boolean framework. In addition, approaches which deal with time-continuous Boolean models are also included since sometimes these features overlap. Note however that an often used class of time-continuous and (potentially) stochastic models, namely GKL (generalized kinetic logic) networks are introduced in a separate section (Section 3.5, i.e. the next section) and their exposition is a little bit more detailed since they form an object of further study in this thesis in Chapter 4.

[Murrugarra et al. 2012] remain in a time-discrete Boolean setting and propose to introduce noise into Boolean networks as follows. Given a species $i\in[n]$, there is only one transition function. But at every time step the actually occurring transition probabilistically depends on whether the state of species i will increase or decrease. Formally, for every species i there is a transition function $f_i:\{0,1\}^n\to\{0,1\}$ and two associated probabilities $p_i^{\uparrow},p_i^{\downarrow}\in[0,1]$ such that the dynamics of the system can be described as follows. Given a state $x^t\in\{0,1\}^n$ at time $t\in\mathbb{N}_0$ the next state of species i will be equal to $x_i^{t+1}=f_i\left(x^t\right)$ with probabilities

$$\begin{cases} p_i^{\uparrow} & \text{if} \quad x_i^t < f_i\left(x^t\right) \\ p_i^{\downarrow} & \text{if} \quad x_i^t > f_i\left(x^t\right) \\ 1 & \text{if} \quad x_i^t = f_i\left(x^t\right) \end{cases}$$

and it will be equal to its present state, i.e. $x_i^{t+1} = x_i^t$, with probabilities

$$\begin{cases} 1-p_i^{\uparrow} & \text{if} \quad x_i^t < f_i\left(x^t\right) \\ 1-p_i^{\downarrow} & \text{if} \quad x_i^t > f_i\left(x^t\right) \\ 1 & \text{if} \quad x_i^t = f_i\left(x^t\right) \end{cases} .$$

This means that the state of a species can switch according to its logic function with a certain probability which depends on the direction of the switch and that species do not switch at all with one minus that respective probability. The probabilities $p_i^{\uparrow}, p_i^{\downarrow} \in [0,1]$ can be interpreted as representing the timescales of the respective processes involved in increasing or decreasing the state of a species. The higher the probability the faster the respective processes. Further, species are assumed to make their update decisions in independently and therefore global state transition probabilities can be written as the product of all the respective probabilities for the single species.

[Garg et al. 2009] examined the concepts of '*noise in nodes'* and '*noise in functions*' which, roughly speaking, relate to the random switching of states of nodes (interpretable for example as measurement or data discretization errors, termed *perturbations* in the PBN framework) and the PBN-type stochasticity applying to the uncertainty in the update functions respectively. As pointed out by [Garg et al. 2009] the 'noise in nodes' concept is often used throughout the literature, for example in [Álvarez-Buylla et al. 2008] or [Davidich, Bornholdt 2008a]. The basic idea of 'noise in nodes' can be described as follows. Given a state $x^t \in \{0,1\}^n$ 'some' noisy processes lead to a noisy state $\tilde{x}^t \in \{0,1\}^n$ and the next state is then obtained via $x^{t+1} = f\left(\tilde{x}^t\right)$. Alternatively, 'noise in nodes' can be applied at the updates themselves according to the following logic. Given a state $x^t \in \{0,1\}^n$ and its formal successor $\tilde{x}^{t+1} = f\left(x^t\right)$, some noisy processes lead to a next state $x^{t+1} \in \{0,1\}^n$ such that some species' states are flipped with respect to the formal successor according to some node-dependent or –independent state flipping probability. 'Noise in functions' is accurately exemplified by the PBNs from Section 3.3. [Garg et al. 2009] tested the *robustness* of an existing Boolean model with respect to both types of noise (for the exact choice and implementation of the noise types, see [Garg et al. 2009]. Robustness in this case relates to the (probabilistic) persistence of the deterministic attractors of the original model under the influence of noise, i.e. how often do trajectories lead to the known deterministic attractors of the original model. The conclusion was, that 'noise in nodes' mostly destroys the (biologically relevant) long-term behavior while 'noise in functions' conserves the attractor structure in the

sense that attractors which had the biggest basins of attractions in the deterministic Boolean model also had the highest probability to be reached by the noisy trajectories.

Robustness is a very general concept and roughly means the 'invariance' of certain properties of a system with respect to changes of other properties of the system or the environment of the system and is also extensively studied in the context of biology [Kitano 2004, 2007], [Stelling et al. 2004], [Barkai, Shilo 2007], [Wagner 2005]. Of course 'robustness' is so general a concept that it plays a role in almost every branch of science and technology. For a further study examining 'robustness' in the context of Boolean network models it is referred to [Willadsen, Wiles 2007].

[Teraguchi et al. 2011] implicitly introduced exponentially distributed time delays into asynchronous Boolean models like the GKL framework described in the next section. [Stoll et al. 2012] proposed a time-continuous stochastic version of asynchronous Boolean modeling which is based on the same principles as the model of [Teraguchi et al. 2011] but in addition [Stoll et al. 2012] also consider state-dependent exponentially distributed time delays and provide a software tool called *MaBoSS* (*Markov Boolean stochastic simulator*) which implements the Gillespie algorithm for the resulting time-continuous Markov process. Formally, every species has associated up- and down-regulation rates $R_i^{up}(x)$ and $R_i^{down}(x)$ which are dependent on the present state $x = x(t) \in \{0,1\}^n$ of the system at time $t \in [0,\infty)$. These rates are thought to be the rates of the continuous-time Markov jump process [Norris 1998: Chapters 2 and 3] as well as [Stoll et al. 2012: Supplement]. For example such a rate can be of the following form: $R_i^{up}(x) = a_1 x_j + a_2(\neg x_j \wedge x_k)$ for some species $k, j \neq i$ and some constants $a_1, a_2 > 0$, i.e. the rate of node i is equal to a_1 if species j is on and it is equal to a_2 if species j is off but species k is on. Given that the system is in state $x = x(t) \in \{0,1\}^n$ at time t, the simulation procedure now simulates the time of the next switching event and the species which will bring about the asynchronous switch according to the rates $R_i^{up}(x)$ and $R_i^{down}(x)$ just as in the Gillespie algorithm described in Subsection 2.2.2 proceeds with the time to the next reaction and the type of the next reaction. This works since the Gillespie algorithm is more generally applicable than in the restricted setting of biochemical reaction networks and can actually be used to simulate any time-continuous Markov jump process (with only finitely many transitions being possible in every state) based on the knowledge of the transition rates. As mentioned the models of [Teraguchi et al. 2011] and [Stoll et al. 2012] can be interpreted as the assumption of exponentially distributed waiting times in a GKL model. However, as outlined in Chapter 5, the assumption of this kind of distribution applied to the heterogeneous processes which are involved in switching on and off Boolean variables might be an oversimplification.

More general time continuous Boolean models have also been proposed, for example by [Öktem et al. 2002] and [Öktem et al. 2003] who deal with a general kind of Boolean delay systems which were mathematically studied by [Dee, Ghil 1984] and [Dee, Mullhaupt 1985] and which are of the general form $x_i(t) = f_i(x_1(t-t_{i1}),...,x_n(t-t_{in}))$ for species i and

delays $t_{ij} > 0$ which describe the delay with which a state of species j gets effected through the Boolean logic function of species i.

Finally, it is noted that [Ivanov, Dougherty 2006] proposed a way to relate time- and state-continuous models based on stochastic differential equations (SDEs) to time-continuous and state-discrete Markov chains.

3.5 Generalized kinetic logic (Thomas formalism)

In this section the Generalized Kinetic Logic (GKL) approach developed by R. Thomas and coworkers [Thomas 1973], [Thomas, D'Ari 1990], [Thomas 2013] is presented.

The most important idea involved in GKL is that of switch-like influences characterized by thresholds. As already mentioned before, it is often the case that the interactions in biochemical networks are governed by steep sigmoid regulation functions, called Hill-type functions (see Subsections 1.3, 2.1 and 3.1). As we will see in Section 3.6 on piecewise linear differential equations (PLDEs) one possible simplifying departure from the general ODE formalism with general Hill-type regulation functions is that the regulation functions are assumed to be "infinitely non-linear", i.e. real discontinuous switch functions. However, the state of a PLDE system is still described by a continuous state vector. In contrast, in the GKL framework the states are now discretized. But the idea of switch-like interactions is, of course, naturally conserved. In contrast to classical Boolean models the GKL involves multi-valued discrete levels for the biochemical species. Concerning the dynamics, GKL is characterized by so called logical parameters (originating from a formal connection to PLDEs [Snoussi 1989], [Thomas, D'Ari 1990]) or more generally by transition functions and an asynchronous updating scheme which is defined via species-specific time-delays.

Since the dynamics of GKL networks is ultimately determined by so called time-delays (introduced below) the values and relations of these time-delays are crucially important. In Chapter 4 I introduce several approaches which aim at meaningful choices of probability distributions for these time-delays. The idea of introducing probability distributions for the time-delays is almost as old as the field of GKL itself [Thomas (ed.) 1979], [Thomas 2013] and also has been conducted in some instances (with uniform distributions over (connected) intervals): [Thomas (ed.) 1979], [Abou-Jaoudé et al. 2009]. As mentioned in the preceding section, [Teraguchi et al. 2011] and [Stoll et al. 2012] implicitly introduced exponentially distributed time delays to GKL networks.

Since notations and some definitions are not unified throughout the literature, the ones adopted in the following are a (to some extent arbitrary) mixture from [Snoussi 1989], [Thomas, D'Ari 1990], [Thomas 1991], [Snoussi, Thomas 1993], [Jamshidi et al. 2013], [Siebert, Bockmayr 2009] and some of my own notational preferences.

Before proceeding to the formal definitions in Subsection 3.5.2 I shall first give a semi-formal motivating description of the details concerning the GKL framework.

3.5.1 Semi-informal description and motivation of GKL

Given one particular species in a biochemical network under study, say species i, it is assumed that i carries out its various influences on some other species in the network only if the concentration level of i has surpassed some characteristic threshold for the respective influences of i with respect to the threshold specific other species. For example, let i influence another species j. Then, there is postulated to exist a specific threshold $\theta_{ij} > 0$ (located on the continuous scale of concentration level for species i) such that i has an influence on j if and only if the concentration level of i is above or below θ_{ij}. So far this is nothing more than the idea of the step-like interaction of species already encountered in Chapters 1 and 2. If we now assume that i influences $k_i \in \mathbb{N}$ other species, we have $p_i \leq k_i$, $p_i \in \mathbb{N}$ thresholds $0 < \theta_i^{(1)}, ..., \theta_i^{(p_i)} < \infty$ (some influences may have the same threshold) which can be w.l.o.g. ordered such that $\theta_i^{(1)} < \cdots < \theta_i^{(p_i)}$. Now it is obvious that we have, depending on the concentration level of species i, $p_i + 1$ qualitatively different situations defined by the influences which are exerted due to the fact that i is above or below certain thresholds. This provides the motivation to define the state space for a regulatory system with N species as $\mathcal{D} := \prod_{i=1}^{N} D_i$ with $D_i := \{0,1,...,p_i\}$.

The interpretation of the introduction of multi-valued logical states as a correspondence to continuous thresholds describing fundamentally different influence regimes for a species is originally due to [Van Ham 1979].

For convenience we introduce the following notation: for $k, m \in \mathbb{Z}$, $k < m$ we define $[k:m] := [k,m] \cap \mathbb{Z}$. For example, we have $D_i = [0:p_i]$.

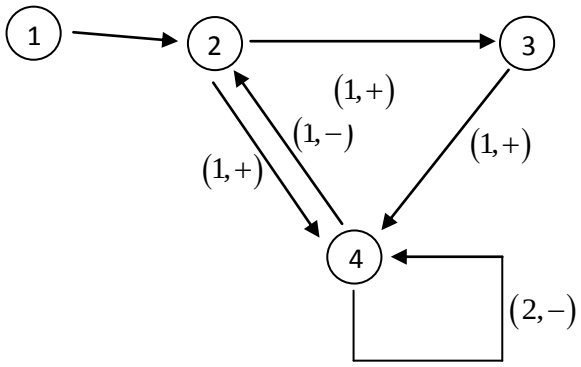
There is an easy and very useful way to transform the model described so far into a purely Boolean framework, see for example [Snoussi, Thomas 1993]. This is usually done by associating to every state variable $q_i \in D_i$, $i \in [1:N]$ a whole set of variables $\{q_i^{(\alpha)} : \alpha \in [1:p_i]\}$ such that $q_i = k$, $k = [1:p_i]$ if and only if $q_i^{(\alpha)} = 1$ for every $\alpha \in [1:k]$ and $q_i^{(\alpha)} = 0$ for every $\alpha \in [k+1:p_i]$ (of course, $\alpha \in [p_i+1:p_i]$ is defined to mean $\alpha \in \varnothing$). In terms of the thresholds $0 < \theta_i^{(1)} < \cdots < \theta_i^{(p_i)} < \infty$ associated to i we have $\theta_i^{(k)} < x_i < \theta_i^{(k+1)}$ (with $k \in [0:p_i]$, $\theta_i^{(0)} := 0$, $\theta_i^{(p_i+1)} := \infty$ and $x_i \geq 0$ being the (for the GKL approach irrelevant but physically real) concentration level of species i) if and only if $q_i = k$ if and only if $q_i^{(\alpha)} = 1$ for every $\alpha \in [1:k]$ and $q_i^{(\alpha)} = 0$ for every $\alpha = [k+1:p_i]$.

In summary, up to now we (implicitly) assumed some interaction graph and introduced the concept of nodes influencing other nodes only in relation to a certain threshold specifically associated to the edge corresponding to the specific influence. To model the relation of nodes to their various thresholds the variables describing the state of the nodes are assumed to be discrete multi-valued and are thought to represent which thresholds are surpassed by the physical but (with respect to the GKL) imaginary continuous concentration variable associated to the node in question. Further we introduced a simple transformation approach which allows one to transform the multi-valued states to a set of purely Boolean variables and vice versa.

We can incorporate the information concerning which of the thresholds is decisive in the interaction of two nodes into the interaction graph by simply weighing the corresponding edge such that the weight indicates the threshold which is important for the 'activity' of the edge in question. What 'activity' can mean, will become clearer in the sequel. In addition, one can sign the weights according to the qualitative impact of the regulating function of the edges (influences) which they weigh, i.e. activating or inhibiting. Note, however, that a priori some edges may have different impacts under different contexts, i.e. inhibiting in one situation and activating in another. We explore that issue below.

Example 3.5

We now introduce a simple example consisting of four species to exemplify the notions introduced so far. First we present the example by the following interaction graph:



What this interaction graph represents, is a situation with three biochemical species interact according to influences represented with the directed edges. The weight (w_e, sgn_e) shown over some edge e means that the appropriate threshold for the edge to be 'active' is the w_e-th threshold of the regulating node. (The regulating variable of a directed edge (i, j) is the node i; j is called the regulated node.) The sign weight sgn_e on the other hand indicates that the regulating node of the edge is an activator $(sgn_e = +)$ or an inhibitor $(sgn_e = -)$ of the regulated node.

So, in terms of the formalism the logic behind the example above is the following. Nodes 2 and 3 positively regulate node 4 which has two thresholds associated while nodes 2 and 3

possess only one threshold. Nodes 2 and 3 activate node 4 whenever they are above their respective thresholds and in addition node 3 is also activated by node 2. Node 4 on the other hand has two thresholds. Above the first it negatively regulates node 2 and above threshold two it additionally inhibits itself (negative auto-regulation). Last but not least node 2 receives an input signal from node 1.

The overall 'system-wide' logic of the network could be that node 4 has to be up-regulated periodically from time to time (to induce some specific reaction to some input situation (osmotic stress, toxicity, etc.) sensed by node 2 and modeled by node 1) which is done via the influences of nodes 2 and 3. But since node 4 has to be down-regulated again (because too long intervals of high level of species 4 might again be toxic to the cell, for example) there are two feedback loops which are meant to achieve the down-regulation of species 4. The first one acts via the whole circuit by inhibiting species 2 as soon as species 4 is above its first threshold and in effect the activation of 4 by 2 and 3 should be switched off after some time-delay. A second mechanism acts directly on species 4 via a negative auto-regulatory loop which is activated if species 4 has still accumulated further such that it exceeds also its second threshold.

To examine the example further and to finally give meaning to expressions like "j negatively regulates i" or "i activates j" we have to specify the precise logic of the interactions and ultimately the dynamics of the system. ▲

So, what still needs to be specified is the dynamics of the GKL. In the most general setting a first step towards a dynamics of the formalism introduced so far is the introduction of a transition function $f:\mathcal{D}\to\mathcal{D}$ which assigns to every state vector a successor state and describes the overall regulatory logic of the system. Historically it is common to call $Q(q_i):=f_i(q)$ the image of variable i given state $q\in\mathcal{D}$. We denote by Q_i the image of variable I given an abstract state of the system in the sense that we can make assertions such as '… the images of Q_i and Q_j differ by three and therefore…' indicating that the assertion made is invariant with respect to the exact states which led to the respective images. $f(q)$ is simply called the image of state $q\in\mathcal{D}$.

Example 3.5 (continued)

Given the interaction graph of our example from above, we can now specify a transition function. Note however that since the variable 'input' is assumed to be constant or in some sense 'external' or under 'control' we do not incorporate it into the state space and hence arrive at the following transition table(s) (i.e. transition function) split up according to the input node (i.e. transition function):

q	f(q)
0000	0000
0100	0011
0010	0001
0001	0000
0002	0000
0011	0001
0101	0011
0110	0012
0012	0000
0102	0010
0111	0012
0112	0010

q	f(q)
1000	1100
1100	1111
1010	1101
1001	1000
1002	1000
1011	1001
1101	1011
1110	1112
1012	1000
1102	1010
1111	1012
1112	1010

(For aesthetic simplicity we write the vectors $q=(q_1,q_2,q_3,q_4)^T\in\mathbb{N}^4$ in sequence form, i.e. $q=q_1q_2q_3q_4$; the same rationale is applied to the image vectors $f(q)$.)

We have already seen how one can transform the multi-valued discrete states to purely Boolean states and vice versa and naturally it is also possible to transform the respective transition functions. One can easily check that the transition table above is equivalent with the following (intuitively convincing) component-wise definitions of the images via the purely Boolean variables associated to the multi-valued states as introduced above:

$$Q(q_1):=q_1^{(1)},\ Q(q_2):=q_1^{(1)}\wedge\overline{q_4^{(1)}},\ Q(q_3):=q_2^{(1)},\ Q(q_4):=\left(q_2^{(1)}+q_3^{(1)}\right)\overline{q_4^{(2)}}.$$

Of course, it is much more natural to specify the component-wise images like that first and only then deduce the transition table (which was actually also done here). Nevertheless the formally decisive entity is the transition function. ▲

The transition function associates to every state exactly one image. The image of a variable is thought to indicate the tendency of the respective variable to change its state, i.e. if the image is larger than the value of the respective state variable, the variable will have the tendency to rise its level and vice versa if the image is lower, the state variable has the tendency to lower its state value. It will become clearer below what 'tendency' to change state actually means. Finally, if image and state coincide the variable won't change its state since the regulatory logic (i.e. the image) indicates that its state is currently consistent with its state value.

One further assumption of GKL is that given a state and its image only one variable can change its state. Once this one variable switched its state, the system is in a new state and therefore also the image has to be changed again which may then lead to again other variables with the tendency to change and so on so forth. The exact mechanism of the switching (i.e. which of the variables switches given that several of them have the tendency to switch and the points in time when such switches take place) is later implemented via time-delays associated to every variable. For now it is sufficient to note that given state and image the next state can

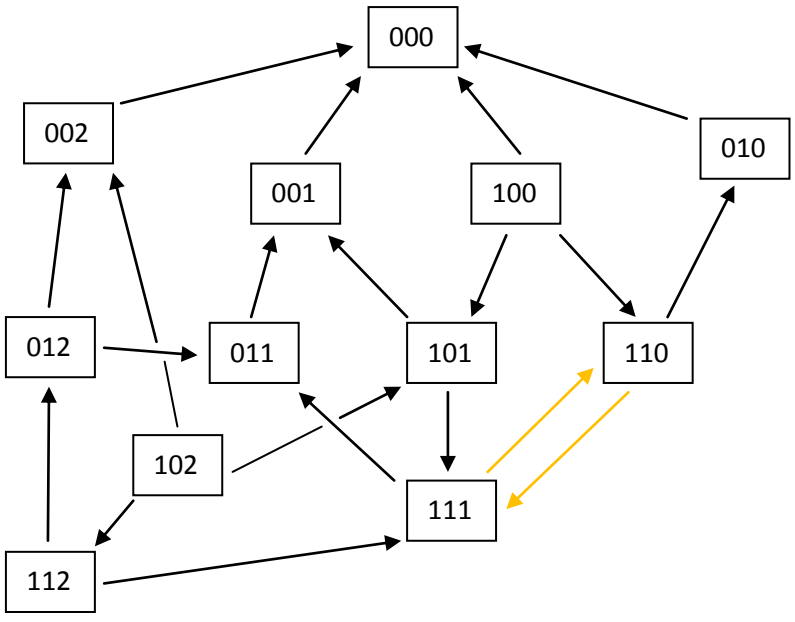
in principle be every state such that this new state differs in exactly one of the state variables which had the tendency to change their state given the old state. Furthermore the switches only lead to changes of one, i.e. if a variable is in state $k \in \mathbb{N}$ then its new state (given that it does not coincide with its image) can only be $k-1$ or $k+1$ (irrespectively of how different the state may have been from its image.) In the light of the interpretation of state variables indicating the location of a continuous concentration level with respect to successive thresholds this seems very natural since thresholds have to be passed one after the other and the concentration levels cannot just jump over an interval between two thresholds what would then correspond to a state variable changing by more than one at one single transition.

One can represent the kind of structure described in the preceding text section via an asynchronous transition graph. The nodes of the graph represent all the possible states of the system and two of them are connected by a directed edge if and only if the root vertex represents a state such that its image allows the transition to the other state according to the logic described above.

Example 3.5 (continued)

For our example we can see that the state space is partitioned into two sets of states such that there cannot be any transitions between them. Because the input node, node 1, always has its very value as its image it will never change its state and hence there are no edges in the state transition graph leading from a state q with $q_1 = 0$ to a state q' with $q_1' = 1$ or vice versa. So, for illustration purposes we can split the transition graph into two subgraphs each one representing either the situation with or without input.

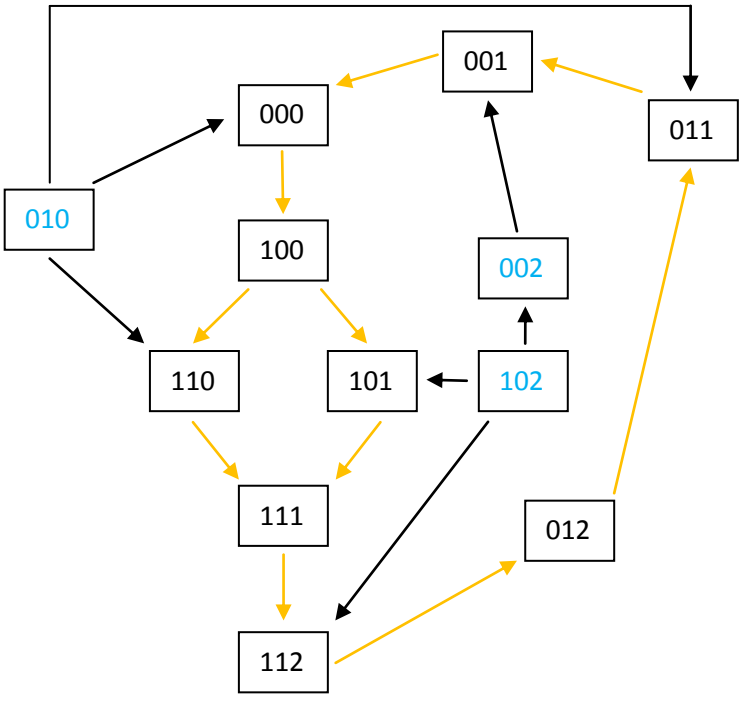
The part of the transition graph which represents the no-input regime (and where the first variable is left out since it is always equal to zero, i.e. the states shown are of the form $q_2q_3q_4$) is as follows:



This graph looks already very complicated. A closer look reveals that the only directed cycle is the 111-110-cycle (the corresponding edges are colored in orange) and that from every state we can go in at most three transitions to 000 which is a steady state, i.e. a state which is equal to its image.

Since the network is thought to react to some input and to be non-functional otherwise the 000 state seems to be most probable for species 2,3 and 4 if no input is present and since 000 is a steady state the system will show no reaction if there is no input (given that the system is already completely relaxed, i.e. in state 000). The other vertices in the transition graph represent states such that the system is not entirely relaxed, maybe because some input preceded the now occurring no-input phase and the transition graph then reveals the various ways in which the system can relax itself to the final relaxation state 000. The 111-110-cycle could have the meaning of some temporary memory such that it may be easier (i.e. faster) to switch to activity (i.e. high values of species 4) if input occurs to be present again. We explore this possibility later on.

On the other hand, the part of the transition graph which represents the regime with present input (again represented by the sequences $q_2q_3q_4$ knowing that $q_1 = 1$) looks as follows:



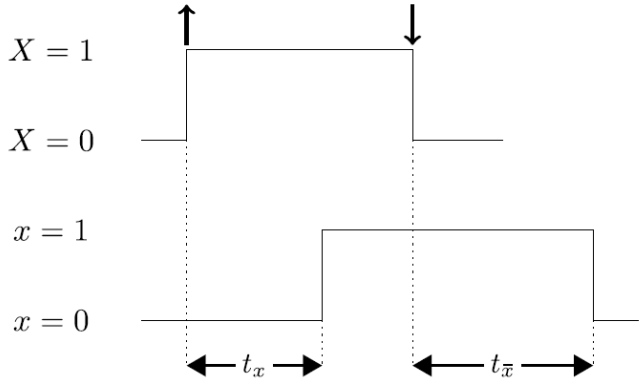
What can be seen regarding this part of the transition graph is the presence of a directed cycle (colored in orange) from state 000 (again 'state' now ambiguously refers to $q_2q_3q_4$, i.e. coordinates 2,3 and 4 of the actual state) to state 112 and 012, i.e. to states where species 4 exceeds even its second threshold and hence can influence some processes (which are not part of the model) designed to be activated exactly when this situation occurs in response to some initial input $q_1 = 1$. What also can be seen is the fact that all states which are not themselves

part of the directed cycle (marked blue) ultimately all lead to some state which is again part of the cycle. Therefore the cycle represents a stable limit cycle and in the presence of input the system periodically drives the fourth species above its second threshold. ▲

In a last step one still has to specify the exact update schedule (see Section 3.1) according to which the transition function f is bound to be applied. In the GKL framework this is usually done via the introduction of time-delays associated to the variables. Generally, every species can switch on or off and to every such process a threshold-characteristic time delay is introduced such that these time delays describe the duration of the respective processes [Thomas, D'Ari 1990], [Thomas 2013]. In summary, if a species happens to have $m \in \mathbb{N}$ thresholds then there are two times m time delays associated to that variable corresponding to the timescales for the respective threshold-dependent processes.

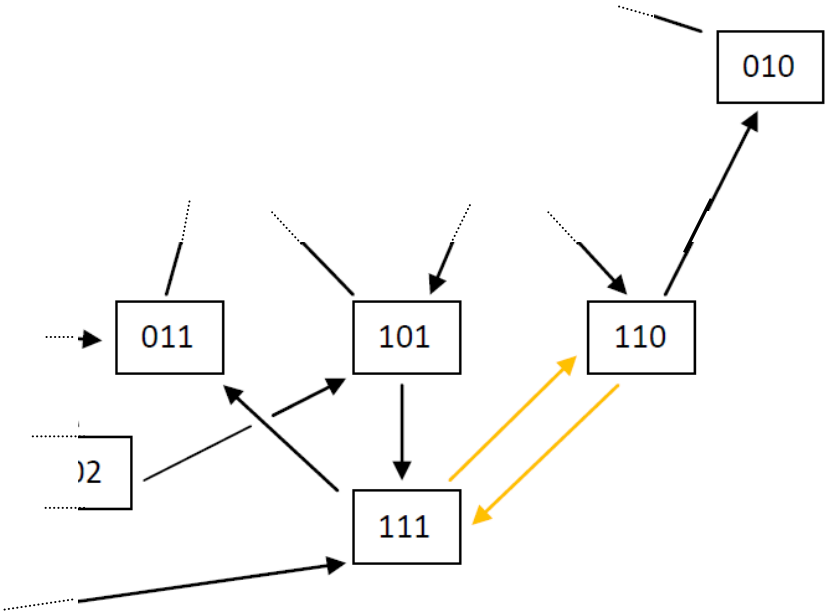
With the introduction of time delays, time is effectively modeled by a continuous variable. As described above the image of a state describes the tendency of the state to change its value and this now happens according to some species- and threshold-specific time-delay. Let us denote the state of species i by x for simplicity, i.e. $x_i := x \in \mathcal{Q}_i$. The time-delay of species i, say the one for the up-regulation corresponding to the first threshold, i.e. the switch from 0 to 1 is denoted by $t_x = t_x^{(1)}$ (on-switch time-delay for first threshold) while the time-delay in case of the switch from 1 to 0 is denoted by $t_{\bar{x}} = t_{\bar{x}}^{(1)}$. Assuming that the discrepancy between state and image, and therefore the first order for the variable to switch, appeared at time $t > 0$, the state will switch to the corresponding new state after time $t + t_x$ or $t + t_{\bar{x}}$ (depending on the character of the proposed switch) (or stay the same if state and image agrees again before time $t + t_x$ or $t + t_{\bar{x}}$ because some other variable switched accordingly).

The following figure (from [Thomas 2013]) depicts the logic of the model for the state x and its image X. The picture shows the up-switching of x with delay t_x relative to the up-switching of its image and the following down-switching with delay $t_{\bar{x}}$ relative to the down-switching of its image. Note that although only the state and the image of X is shown the other species inevitably are also playing a role since the first up-switching of the image as well as the down-switching of the image which follows ultimately have to be brought about by some of the other variables changing its state such that the image X of x changes again:



from [Thomas 2013]

Given that every variable has time delays associated the system now evolves according to inequalities involving the sums of these delays. For example, look at the following snapshot from the above state transition graph for the GKL network of example 3.5 with input zero:



Assume that the system is in state 110 and that the system just initialized, i.e. we can ignore all the history the system might possess. There are now two possibilities for the system to evolve: either species 2 (corresponding to the first number in the state…) switches down first in which case the next state of the system will be 010 or species 4 (corresponding to the third number in the state) switches up first in which case the the next state of the system will be 111. This decision is brought about according to which of the time delays, the one associated to the up-switching of 4, say t_4, or the one associated to the down-switching of 2, say $\overline{t}_2$, is smaller. So, in a deterministic setup the smaller time delay decides the dynamics. Let us assume that in this example the delay for the up-regulation of 4 is smaller, i.e. $t_4 < \overline{t}_2$, and the next state of the system is therefore 111. Now there are again two possibilities, either species 4 switches down again and the next state is again 110 or species 2 switches down such that the next state would be 011. In principal, the decision now again depends on the comparison of the responsible time delays $\overline{t}_4$ and $\overline{t}_2$. But now we have to take into account that the image of species 2 differs from its state not just since the last switch of the system but actually since the step before. So, in order that species, say, 4 switches down, the relation which has to be fulfilled is now $t_4 + \overline{t}_4 < \overline{t}_2$. Then, again in state 110 the system will switch back to 111 since we assumed $t_4 < \overline{t}_2$. Notice that species 2 remains 'activated' all the time in the sense that it disagrees with its image without switching. In order for the system to again switch back to 110 the following inequality has to be satisfied: $2t_4 + 2\overline{t}_4 < \overline{t}_2$. Generally, one can see that if the system starts (without history) in state 110 with $t_4 < \overline{t}_2$ there will be $\eta \in \mathbb{N}$ switches from 110 to 111 (and $\eta - 1$ back) until the system finally reaches state 011 where $\eta \in \mathbb{N}$ is the smallest integer with $\eta(t_4 + \overline{t}_4) > \overline{t}_2$. In this sense deterministic time delays determine the

dynamical behavior of a GKL network. In terms of continuous time, if the system starts at 110 at time $t = 0$ (without history), it finally reaches state 011 at time $t = \eta t_4 + (\eta - 1)\bar{t}_4 + \bar{t}_2$.

Applications of GKL networks to biological systems

GKL networks have been applied to model biological systems (what is only mildly surprising taking into consideration the fact that they were invented to do so). In [Thomas (ed.) 1979] GKL is applied to various example networks. [Thomas 1979] and [Thieffry, Thomas 1995] apply GKL to model the gene regulatory network of λ-phage (see Subsection 1.3.7).

[Sánchez, Thieffry 2001] and [Thieffry, Sánchez 2002] address the modeling of the so called gap gene regulatory system in Drosophila melanogaster embryos. [Sánchez, Thieffry 2003] then studies the next regulatory step in the Drosophila embryo, the pair-rule gene regulatory network, while [Sánchez et al. 2008] finally model the so called segment polarity gene network which is still a step further in the overall fruit fly development than the pair-rule genes.[9]

[Mendoza et al. 1999] model flower morphogenesis in Arabidopsis thaliana with a GKL model.

[Abou-Jaoudé et al. 2009] apply GKL modeling to the so called p53-Mdm2 system. One interesting aspect of this study with respect the GKL framework is the introduction of deterministically time-dependent time delays.

The next subsection will provide the formal definitions concerning GKL networks.

3.5.2 Formal definitions for GKL networks

In this subsection the intuitive notions of the preceding subsection are made precise and having the informal explorations from the latter in mind, the definitions should be readily interpretable and in order to keep the exposition within certain bounds, minimal further explanatory remarks are made in this subsection. First one can define the general form of a GKL network as follows.

Definition 3.5.1 (GKL network) (adapted from [Jamshidi et al. 2013])

A ***GKL network*** is a tuple (V, η, f) with a finite set $V \neq \varnothing$, a *level map* $\eta : V \to \mathbb{N}$ and an *image function* $f : \prod_{i \in V} \{0,1,...,\eta(i)\} \to \prod_{i \in V} \{0,1,...,\eta(i)\}$.

[9] For a general overview on the mathematical modeling of Drosophila embryogenesis it is referred to [Jaeger 2009].

V is also called the *species set* and in the following I assume w.l.o.g. $V = [n] = \{1,...,n\}$ for some $n \in \mathbb{N}$. Any $i \in V$ is called a *species* (of the GKL network) and $p_i := \eta(i)$ is called the *level* of species i. For $i \in V$ the set $\mathcal{Q}_i := \{0,1,...,p_i\}$ is called the (state) *range* of species i. The set $\mathcal{Q} := \prod_{i=1}^{n} \mathcal{Q}_i$ is called the *state space* of the GKL network and the image function can thus be conceived as a mapping from the state space to itself, i.e. $f : \mathcal{Q} \to \mathcal{Q}$.

GKL networks with $p_i = 1$ for all $i \in V$ correspond to classical Boolean networks as defined in Section 3.1.

Based on a given GKL network its state transition graph is defined such that transitions are only possible between states which differ in exactly one state such that this respective state was ordered to switch according to the image function.

Definition 3.5.2 (State transition graph; STG) [Jamshidi et al. 2013]

> Given a GKL network $N = (V, \eta, f)$ the (***state***) ***transition graph*** $STG(N)$ of N is defined as $STG(N) := (\mathcal{Q}, E)$ with state space $\mathcal{Q}$ and the *transitions* $E \subset \mathcal{Q} \times \mathcal{Q}$ defined such that $(q, \tilde{q}) \in E$ if and only if
>
> $$\exists j \in [n] : \tilde{q}_j = q_j + \operatorname{sgn}(f_j(q) - q_j) \neq q_j \ \wedge \ \forall i \in [n] \setminus \{j\} : \tilde{q}_i = q_i .$$

Logical steady states (fixed points) are defined in straight analogy to the Boolean case as states $q \in \mathcal{Q}$ for which $f(q) = q$ holds. More generally, attractors of a GKL network are defined as for asynchronous Boolean networks.

Definition 3.5.3 (Attractors of GKL networks)

> Let $N = (V, \eta, f)$ be a GKL network with transition graph $STG(N) := (\mathcal{Q}, E)$.
> $\mathcal{A} \subset \mathcal{Q}$ is called ***attractor*** of N if $\forall (q, \tilde{q}) \in E : q \in \mathcal{A} \Rightarrow \tilde{q} \notin \mathcal{Q} \setminus \mathcal{A}$ and $\forall q, \tilde{q} \in \mathcal{A}$ $\exists$ directed path with tail vertex q and head vertex $\tilde{q}$.

So far, I defined GKL networks without dynamics. Dynamical behavior in GKL networks is defined through the introduction of time delays. The following definitions are especially designed in order to meet the needs of the thesis.

Definition 3.5.4 (Timed GKL network)

A ***timed GKL network*** is a tuple $\Sigma = (N, \mathcal{T})$ where $N = (V, \eta, f)$ is a GKL network and $\mathfrak{T} = \{\tau_i : [p_i] \times \{-1, +1\} \to [0, \infty) : i \in V\}$.

For every $i \in V$, the map $\tau_i : [p_i] \times \{-1, +1\} \to [0, \infty)$ associates level-dependent time-delays to the respective species which also depend on whether the variable is about to increase or decrease (see below). I define further $\tau_i^{(\sigma)} := \tau_i(\sigma, +1)$ and $\bar{\tau}_i^{(\sigma)} := \tau_i(\sigma, -1)$ for all $i \in V$ and $\sigma \in [p_i]$. $\tau_i^{(\sigma)} \in [0, \infty)$ describes the time needed to switch the state of species i from $\sigma - 1$ to σ while $\bar{\tau}_i^{(\sigma)} \in [0, \infty)$ describes the time needed to switch the state of species i from σ to $\sigma - 1$. The transition graph for timed GKL networks is just defined to be the transition graph of the respective GKL network N.

A different approach to formally incorporate time delays was for example taken by [Siebert, Bockmayr 2009]. There the Thomas formalism was formulated and extended with a view towards the application of the established theory of so called *timed automata* studied in theoretical computer science. The definition above is manufactured in order to fit the needs of this thesis.

Now, one can define the dynamics for a timed GKL network in terms of its time delays. In order to be able to make the following definition however it is inevitable to introduce two more formal notions. Let a (timed) GKL network as in Definition 3.5.4 be given. First, for $q \in \mathcal{Q}$ define the *children* (or *successor*) *states* of the state q with respect to the given transition graph as $\mathcal{C}_{STG}(q) := \{\tilde{q} \in \mathcal{Q} : (q, \tilde{q}) \in E\}$. Second, I define the so called *switching map* $\phi^{(q)} = (\phi_1^{(q)}, \phi_2^{(q)}, \phi_3^{(q)}) : \mathcal{C}_{STG}(q) \to V \times \bigcup_{j \in V} [p_j] \times \{-1, +1\}$ for every state $q \in \mathcal{Q}$ by the following component-wise definitions:

$$\phi_1^{(q)}(\tilde{q}) := \sum_{j=1}^{n} j \mathbb{I}(q_j \neq \tilde{q}_j),$$

$$\phi_2^{(q)}(\tilde{q}) := \sum_{j=1}^{n} \mathbb{I}(q_j \neq \tilde{q}_j)\left[q_j + \mathbb{I}(\operatorname{sgn}(\tilde{q}_j - q_j) = +1)\right],$$

$$\phi_3^{(q)}(\tilde{q}) := \sum_{j=1}^{n} \operatorname{sgn}(\tilde{q}_j - q_j).$$

Although this definition of the switching map looks rather complicated, it just describes for a given state q and one of its successors $\tilde{q} \in \mathcal{C}_{STG}(q)$ which species has to change its state ($=\phi_1^{(q)}(\tilde{q})$), which corresponding threshold has to be crossed ($=\phi_2^{(q)}(\tilde{q})$), i.e. which time-delay plays a role and finally $\phi_3^{(q)}(\tilde{q})$ encodes whether the species specified by $\phi_1^{(q)}(\tilde{q})$ has to decrease or increase its state by one in order to bring about the transition from q to $\tilde{q}$.

For state $q \in \mathcal{Q}$ the set $\mathcal{D}(q) := \left\{ \tau_{\phi_1^{(q)}(\tilde{q})}\left(\phi_2^{(q)}(\tilde{q}), \phi_3^{(q)}(\tilde{q})\right) : \tilde{q} \in \mathcal{C}_{STG}(q) \right\}$ is called *active delay set* of the state and its elements are said *to be active* for state q. A second useful notion associated to a state $q \in \mathcal{Q}$ is the *set of active species* $\mathcal{A}(q) = \left\{ \phi_1^{(q)}(\tilde{q}) \in V : \tilde{q} \in \mathcal{C}_{STG}(q) \right\}$. The elements of the active species set are called *active species* with respect to state $q \in \mathcal{Q}$. Note the formal difference between active delays and species respectively. However, there is a bijective mapping between these two such that every active species gets mapped to its corresponding active delay and they, active delay and active species, essentially describe the same situation of switching potential (associated to a certain state $q \in \mathcal{Q}$).

Next, a *switching delay* $\alpha : V \times \mathbb{N}_0 \to \mathbb{N}_0$ will be used to describe the dynamics: $\alpha(i,\omega) = \kappa$ represents the fact that after the ω-th overall switching event, species i is still active with delay $\tau_i\left(q_i^{(k-\kappa)} + 1, +1\right)$ or $\tau_i\left(q_i^{(k-\kappa)}, -1\right)$ without having switched or a qualitative change in the respective image having taken place. Formally, we have for a state $q \in \mathcal{Q}$ the *active delay association map* for $\omega \in \mathbb{N}_0$ as

$$\tau^{(\omega)} : \mathcal{A}(\omega) \to \mathcal{D}(\omega),\ \phi_1^{(\omega)}(\tilde{q}) \to \tau_{\phi_1^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right)$$

where $\mathcal{A}(\omega) := \mathcal{A}\left(q^{(\omega)}\right) = \left\{ \phi_1^{(\omega)}(\tilde{q}) \in V : \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right) \right\}$ is the set of active species in state $q^{(\omega)} \in \mathcal{Q}$ and where $\mathcal{D}(\omega) := \mathcal{D}\left(q^{(\omega)}\right) := \left\{ \tau_{\phi_i^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right) : \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right) \right\}$ is the set of active delays in $q^{(\omega)} \in \mathcal{Q}$.

Now, we are prepared to algorithmically define the dynamics of a timed GKL network.

Algorithm 3.5 [**Dynamics of timed GKL networks**]

Input: I.1 $\Sigma = (N, \mathcal{T})$ # a timed GKL network

with, $N = (V, \eta, f)$ being a GKL network and $\tau_i^{(\sigma_i)}$ and $\bar{\tau}_i^{(\sigma_i)}$ being

the time delays associated to threshold $\sigma_i \in [p_i]$ of species $i \in V$ (see above)

I.2 $q^{(0)} \in \mathcal{Q}$ # initial state

I.3 $t_{max} \in [0, \infty)$ # maximal time

Output: O1. $Q_{q^{(0)}} : [0, t_{max} + \varepsilon] \to \mathcal{Q}$ # continuous time evolution for some $\varepsilon > 0$

O2. $\left(q^{(\omega)}\right)_{\omega} \in \mathcal{Q}^{\Omega+1}$ # associated jump dynamics, $\Omega \in \mathbb{N}_0$ is the number of switches which have taken place

---

(1) $\forall i \in V: \alpha(i,0) = 0$ # initialization of switching delays

(2) $\xi_{q^{(0)}}(0) \leftarrow q^{(0)}$ # initialization of jump process

(3) $t \leftarrow 0$ # initialization of continuous time

(4) $\omega \leftarrow 0$ # discrete jump time

(5) $\mathcal{A}(-1) \leftarrow \varnothing$ # "active states before start" (technically necessary)

(6) while $t \le t_{max}$:

(5.1) $\mathcal{C}_{STG}\left(q^{(\omega)}\right) := \left\{\tilde{q} \in \mathcal{Q} : \left(q^{(\omega)}, \tilde{q}\right) \in E\right\}$ # successor states, this can be done by computing the image, i.e. the STG has not to be computed a priori!

(5.2) $\phi^{(\omega)} = \left(\phi_1^{(\omega)}, \phi_2^{(\omega)}, \phi_3^{(\omega)}\right) : \mathcal{C}_{STG}\left(q^{(\omega)}\right) \to V \times \bigcup_{j \in V} \left[p_j\right] \times \{-1, +1\}$

compute the switching map

(5.3) $\mathcal{A}(\omega) := \left\{\phi_1^{(\omega)}(\tilde{q}) \in V : \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right)\right\}$ # active species

(5.4) $\mathcal{D}(\omega) = \left\{\tau_{\phi_1^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right) : \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right)\right\}$ # active delays

(5.5) $\tau^{(\omega)} : \mathcal{A}(\omega) \to \mathcal{D}(\omega),\ \phi_1^{(\omega)}(\tilde{q}) \to \tau_{\phi_1^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right)$

active delay association map

(5.6) for $\tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right)$ with $\phi_1^{(\omega)}(\tilde{q}) \in \mathcal{A}(\omega) \cap \mathcal{A}(\omega - 1)$:

update switching delays for active delays

(5.6.1) if $\tau^{(\omega)}\left(\phi_1^{(\omega)}(\tilde{q})\right) = \tau^{(\omega-1)}\left(\phi_1^{(\omega)}(\tilde{q})\right)$:

(5.6.1.1) $\alpha\left(\phi_1^{(\omega)}(\tilde{q}), \omega\right) \leftarrow \alpha\left(\phi_1^{(\omega)}(\tilde{q}), \omega - 1\right) + 1$

$\phi_1^{(\omega)}(\tilde{q})$ is active with same delay as before, hence the switching delay for the respective delay has to be updated

(5.6.2) else:

(5.6.2.1) $\alpha\left(\phi_1^{(\omega)}(\tilde{q}), \omega\right) \leftarrow 0$

$\phi_1^{(\omega)}(\tilde{q})$ is active with new delay, only possible if the difference of state and image of the species changed

sign with the switch from $q^{(\omega-1)}$ to $q^{(\omega)}$

(5.7) for $i \in V \setminus \mathcal{A}(\omega)$:

(5.7.1) $\alpha(i,\omega) \leftarrow 0$ # "switching" delays of non-active species

(5.8) for $i \in \mathcal{A}(\omega)$:

(5.8.1) $\kappa \leftarrow \alpha(i,\omega)$ # switching delay

(5.8.2) $\Sigma_i \leftarrow \sum_{\ell=\omega-\kappa+1}^{\omega} t_\ell$ # time since activation of delay

$$\left(\sum_{\ell=a}^{b} t_\ell := 0 \text{ for all } a > b \right)$$

(5.8.3) $\tau_i \leftarrow \tau^{(\omega)}(i) - \Sigma_i$

residual delay with respect to its activation time

(5.9) $s \leftarrow \arg\min_{i \in \mathcal{A}(\omega)} \tau_i$ # switching species

(5.10) $t_\omega \leftarrow \min_{i \in \mathcal{A}(\omega)} \tau_i$ # time to next switch

(5.11) $q^{(\omega+1)} \leftarrow \tilde{q}$ with $s = \phi_1^{(\omega)}(\tilde{q}),\ \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right)$

next state according to switching species

(5.12) $\xi_{q^{(0)}}(\omega+1) \leftarrow q^{(\omega+1)}$ # update of discrete jump output

(5.13) $\forall \tau \in [t, t+t_\omega): Q_{q^{(0)}}(\tau) \leftarrow q^{(\omega)}$ # update of continuous-time output

(5.12) $t \leftarrow t + t_\omega$ # update of continuous time

(5.13) $\omega \leftarrow \omega + 1$ # update of jump time

In Section 4.1 the above algorithm is slightly adapted to allow for the incorporation of probabilistic time delays, see algorithm 4.1.

As mentioned before the preceding definitions are manufactured to meet the needs of this thesis. Other approaches to formally incorporating time delays are [Bernot et al. 2004], [Ahmad et al. 2007] or [Bockmayr, Siebert 2009].

3.6 Piecewise linear differential equations

Piecewise linear differential equations (PLDEs) were introduced by [Glass, Kauffman 1973]. For a review see for example [De Jong et al. 2004a] and also the respective section in [De Jong 2002] for a more condensed explanation of the approach.

PLDEs are a specialization of the general ODE model framework introduced in Section 2.1:

$$\dot{x}_i = g_i(x) - \gamma_i x_i \ , \ i = 1,...,n \, .$$

Here x_i is again the continuous concentration level of the i-th chemical species with n being the number of species involved. $\gamma_i > 0$ is the (linear) degradation rate of species i and degradation is assumed to be linear.

What makes such a system of general ODEs a piecewise linear system of ODEs is the special structure of the right-hand side, more precisely the nature of the functions $g_i : \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$. In PLDEs they are just Boolean-like step functions, i.e. radical versions of the more physically realistic sigmoid (Hill-type) regulation functions introduced in section 2.1:

$$g_i(x) = \sum_{j \in L_i} \kappa_{ij} b_{ij}(x) \, .$$

Here, L_i is some (finite) index set, the $\kappa_{ij} > 0$ are parameters describing the strength of the encoded regulatory interactions and the functions $b_{ij} : \mathbb{R}^n_{\geq 0} \to \{0,1\}$ finally are sums of products of step functions $s^+\left(x_j, \theta_j^{(\ell)}\right) := \begin{cases} 1 & x_j > \theta_j^{(\ell)} \\ 0 & x_j < \theta_j^{(\ell)} \end{cases}$ or $s^-\left(x_j, \theta_j^{(\ell)}\right) := 1 - s^+\left(x_j, \theta_j^{(\ell)}\right)$ where the constants $\theta_j^{(\ell)} > 0$ are thresholds for activation or repression. Every species j can have several thresholds $\theta_j^{(\ell)} > 0$, $\ell = 1,...,p_i \in \mathbb{N}$ depending on the respective thresholds with which it influences other species i via their regulation functions g_i. This means that given thresholds $\theta_i^0 := 0 < \theta_i^1 < \theta_i^2 < \cdots < \theta_i^{p_i} < \infty =: \theta_i^{p_i+1}$ for every $i = 1,...,n$ a system of PLDEs is given by

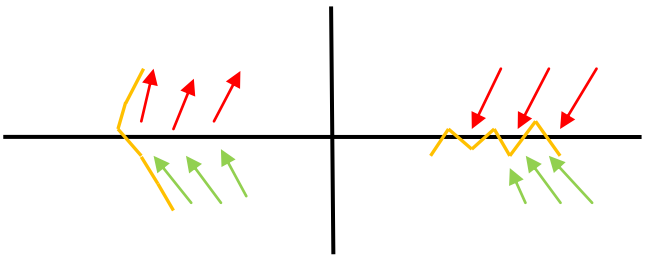
$$\dot{x}_i = \beta_{i,\mathcal{D}} - \gamma_i x_i \ , \ i = 1,...,n$$

in every open hypercube $\mathcal{D}_{j_1 \cdots j_n} = \prod_{i=1}^{n} \left(\theta_i^{j_i}, \theta_i^{j_i+1}\right)$ with $j_i \in \{0,...,p_i\}$ where the hypercube-specific constant $\beta_{i,\mathcal{D}} \geq 0$ is obtained via $\beta_{i,\mathcal{D}} = g_i(x)$ for some (and hence all) $x \in \mathcal{D}_{j_1 \cdots j_n}$. The hypercubes $\mathcal{D}_{j_1 \cdots j_n} = \prod_{i=1}^{n} \left(\theta_i^{j_i}, \theta_i^{j_i+1}\right)$ are also called the *regulatory domains* of a given PLDE system.

To give an example, we model the situation of an AND-logic regulation of some species i via two other species j and k. Imagine that there are thresholds $\theta_j, \theta_k > 0$ such that when both the state variables of j and k are above their respective thresholds the production rate of species i is set to some constant $\beta_i > 0$ while if just one of the two regulators j or k is below its respective threshold, species i is not produced at all. The situation described can be easily captured in the following functional form (with some linear degradation rate $\gamma_i > 0$):

$$\dot{x}_i = \beta_i s^+\left(x_j, \theta_j\right) s^+\left(x_k, \theta_k\right) - \gamma_i x_i \,.$$

Given this kind of simplified ODE formalism one can try the same analyses as with general ODE systems: examine the steady states, solve numerically, etc. But one has to be careful. While the system is very well behaved in the regulatory domains (and even analytically solvable in every such domain) the system as a whole can show subtle behavior depending on the behavior of the system at the regulatory domain boundaries. Formally, the system is not defined there but intuitively on can picture a typical situation which can occur at a boundary as follows (the lines represent boundaries, the red and green arrows on either side of the boundaries represent the vector fields in the respective regulatory domains and the orange zig-zag routes exemplify two typical situations of the (time-discretized) dynamics for a PLDE system at its domain boundaries):



While on the boundary on the left the dynamics just 'jumps over' the discontinuity the dynamics on the right shows a case were the dynamics is somehow trapped onto a boundary of the system. The situation on the right is usually called a sliding mode solution and is a well-known phenomenon in the theory of differential equations with discontinuous right-hand sides and there are ways to formalize the described issues. One is the definition of so called *Fillipov solutions* [Fillipov 1988], [Gouzé, Sari 2002], [Casey et al. 2006] where the vector field of the discontinuous system is extended onto the boundaries of discontinuity by means of convex combinations of the respective neighboring vector fields. See also [Sastry 1999]. The domain boundaries of a PLDE system are also called *switching boundaries* or *switching domains*.

I point out two further things here. For every regulatory domain $\mathcal{D}$, a unique 'steady state' is given by $\bar{x}_{i,\mathcal{D}} = \beta_{i,\mathcal{D}} / \gamma_i$ for $i = 1,...,n$ and if $\bar{x}_{i,\mathcal{D}} \in \mathcal{D}$ for all $i = 1,...,n$ the trajectories of the system will converge to that state once they (somehow) reached the domain. But since the system is more than just one ODE on only one domain the 'steady states' do not always lie in the regulatory domain to which they are associated to. The state $\bar{x}_{\mathcal{D}} = \left(\bar{x}_{1\mathcal{D}},...,\bar{x}_{n\mathcal{D}}\right)^T \in \mathbb{R}^n_{\geq 0}$ is called focal point of the regulatory domain $\mathcal{D}$. See [Casey et al. 2006] or [Edwards 2000].

Lastly, it can be shown that if $\max_{x \geq 0} \frac{g_i(x)}{\gamma_i} < \max_i \in [0, \infty)$ the set $\Omega := \prod_{i=1}^{n} [0, \max_i]$ is invariant [De Jong et al. 2004a].

3.6.1 Relation to logical models and qualitative simulation

Intuitively it should be clear that the PLDE formalism has logical or Boolean flavor since regulation is modeled via the combination of functions only taking values 0 or 1, i.e. logic functions. I exemplified this in the preceding section with the simple model of an AND-gate.

Theorem 1 of [Snoussi 1989] shows that fixed points in the the asynchronous transition graph (see definition 3.5.2) correspond to asymptotically stable steady states of the 'associated' PLDE system and vice versa. It is relatively straightforward to associate a logical model to a PLDE system just by translating the various combinations of step functions into multi-valued logical functions. Snoussi formalized that notion by ascribing logical values to the different regulatory domains where a logical state $g = (g_1,...,g_n) \in \prod_{i=1}^{n} \{0,1,...,p_i\}$ reflects the fact that the continuous system is situated in the domain $\mathcal{D}_{g_1...g_n} = \prod_{i=1}^{n} \left(\theta_i^{j_i}, \theta_i^{j_i+1}\right)$ with $\theta_i^0 := 0,\ \theta_i^{p_i+1} := \infty$. Let us call $g = (g_1,...,g_n)$ the logical index of the regulatory domain $\mathcal{D}_{g_1...g_n}$. Snoussi's theorem then says that every focal point which lies within its respective domain (and hence is an asymptotically stable steady state) is also a logical steady state (fixed point) of the associated logical model in the sense that the logical index of the domain which belongs to the 'within-its-domain' focal point is a fixed point of the logical model.

[Snoussi, Thomas 1993] introduced the concept of *loop characteristic states* which allows to identify PLDE steady states which are not 'within-its-domain' focal points but so called *singular steady states* arising from the behavior of the PLDEs at the switching boundaries. The approach has to make some specializing assumptions concerning the structure of the PLDE model and was further evaluated in [Thomas et al. 1995]. See also [Devloo et al. 2003] and [Plathe et al. 1998].

While the described approaches are mainly thought to be used in order to derive the steady state behavior (or at least some aspects thereof) of a continuous model by means of related logical models in Section 3.8 I will describe methods which try to enrich Boolean models with some kind of continuous dynamics. This ways or the other, the question always arises whether the behavior (i.e. steady states for example) of one model corresponds 'somehow' (i.e. exact, asymptotically or only under some conditions, etc.) to the behavior of the other model. A general result concerning the correspondence between steady states which encompasses results proven in [Snoussi 1989] and [Wittmann 2009a] was proven in [Veliz-Cuba et al. 2012].

I further remark that a qualitative abstraction of PLDE systems termed *qualitative simulation* which takes into account the peculiarities which can arise at the switching boundaries was designed by [De Jong et al. 2004a] and successfully applied for example to model the regulatory network responsible for the sporulation of Bacillus subtilis [De Jong et al. 2004b].

[Chaves et al. 2010] and [Jamshidi et al. 2013] are two further studies addressing the question of formal relationships between Boolean networks, GKL networks and PLDEs. [Chaves et al. 2010] examine possible transformations between Boolean networks, GKL networks and PLDE models while [Jamshidi et al. 2013] show that the dynamical behavior can differ considerably between the GKL and the PLDE framework. For even more details one can consult [Jamshidi 2012].

3.7 Petri nets

Petri nets have also been used to model regulatory biochemical networks, see [Chaouiya 2007] or [Hardy, Robillard 2004] for review and [Matsuno et al. 2000], [Sackmann et al. 2006] or [Steggles et al 2007] for more specific examples. Petri nets are (in their basic form) discrete-time, discrete-state models but of a slightly different flavor than Boolean networks. The inclusion of Petri nets in this chapter is motivated by existing methods to formally relate Boolean networks and GKL networks to Petri nets: [Chaouiya et al. 2004], [Chaouiya et al. 2008]. Furthermore, there are well-established extensions of Petri nets which incorporate stochasticity and/or continuity, see for example [Alla, David 1998], [Marsan et al. 1995], [Haas 2002] or [Bause 2002], which opens up alternative ways to include these features into discrete Boolean approaches.

In this subsection I briefly define Petri nets based on the definition given in [Chaouiya et al. 2008]. For a general treatment of Petri nets it is referred to [Murata 1989].

Definition 3.7.1 (**Petri net**) [Chaouyia et al. 2008]

A Petri net is a 5-tuple $(\mathrm{P}, \mathrm{T}, \mathrm{Pre}, \mathrm{Post}, \mathrm{M}_0)$ such that P and T are finite sets (called *places* and *transitions* respectively) with $\mathrm{P} \cap \mathrm{T} = \varnothing$ and $\mathrm{P} \cup \mathrm{T} \neq \varnothing$, $\mathrm{Pre}: \mathrm{P} \times \mathrm{T} \to \mathbb{N}_0$, $\mathrm{Post}: \mathrm{T} \times \mathrm{P} \to \mathbb{N}_0$ and $\mathrm{M}_0 : \mathrm{P} \to \mathbb{N}_0$.

The above definition can also be interpreted in terms of a weighted directed bipartite graph. The interpretation of the above definition is roughly as follows. The places P represent species and the amount or the activity of the respective species is represented by so called *tokens* where $\mathrm{M}_0 : \mathrm{P} \to \mathbb{N}_0$ defines the initial number of tokens for every place (called an *initial marking*). The places 'participate' in some of the transitions T according to 'participation strengths' (measured in tokens) defined by the weights $\mathrm{Pre}: \mathrm{P} \times \mathrm{T} \to \mathbb{N}_0$. The transitions which take place again have an influence on the activity (amount) of tokens associated to the places according to $\mathrm{Post}: \mathrm{T} \times \mathrm{P} \to \mathbb{N}_0$. Pre and Post can be canonically interpreted as $|\mathrm{P}| \times |\mathrm{T}|$- and $|\mathrm{T}| \times |\mathrm{P}|$-matrices respectively. In order to be able to define the dynamical behavior (given

by the evolution of token associated to the places over (discrete) time, i.e. by markings $M_n : P \to \mathbb{N}_0$ for $n \in \mathbb{N}_0$) the following definition is necessary. A transition $t \in T$ is called *enabled* by a marking $M : P \to \mathbb{N}_0$ if $Pre(p,t) \leq M(p)$ for all $p \in P$, i.e. if the 'participation strength' of any place (with respect to the transition in question) does not exceed the available number of tokens at place p as given by the marking M.

The dynamic evolution of a Petri net can now be defined as follows. At every time step, given a marking $M_n : P \to \mathbb{N}_0$, 'some' enabled transition $t \in T$ will happen. It is said that the transition is *firing*. The exact choice which of the enabled transition fires is up to the specific choice of the modeler and leaves plenty of room for model-dependent solutions, see [Murata 1989]. When a particular transition fires, the $Pre(p,t)$ tokens are removed from every place p while $Post(p,t)$ tokens are added at each place. Formally this can be expressed with the *incidence matrix* $C := Post^T - Pre \in \mathbb{N}_0^{|P|\times|T|}$ as $M_{n+1} := M_n + Ce_i$ where $e_i := (\delta_{ij})_{j=1,\ldots,|T|} \in \{0,1\}^{|T|}$ describes the firing of the i-th transition.

Steady states in the Petri net framework correspond to so called *non-live markings* M which are defined as markings such that not transition $t \in T$ is enabled by M.

Petri nets where originally invented by Carl Adam Petri [Petri 1962, 1963], [Murata 1989] in order to model chemical reactions and first mainly have found biological applications in models of mass flow networks (i.e. chemical reaction networks) such as metabolic networks [Hofestäd, Thelen 1998], [Heiner, Koch 2004], [Zevedei-Oancea, Schuster 2003] which is somehow intuitive since tokens can be straightforwardly interpreted as molecule numbers and pre- and post-weights as stochiometries.

3.8 Fuzzy logic, SQUAD and Odefy

This section summarizes three approaches which were taken to associate some kind of continuous state space to Boolean models. Subsection 3.8.1 deals with *fuzzy logical models* (see references therein), Subsection 3.8.2 with so called *standardized qualitative dynamical systems* [Mendoza, Xenarios 2006] which are implemented in the software *SQUAD* [Di Cara et al. 2007] and finally, Subsection 3.8.3 summarizes the approach taken by [Wittmann et al. 2009a] via *multivariate polynomial interpolation* which is implemented in the software *Odefy* [Krumsiek et al. 2010]. Both of the latter approaches automatically associate an ODE model to an existing Boolean model. Reviews on the overall theme of this section can be found in [Samaga, Klamt 2013] and the respective subsection of [Wittmann et al. 2009a]. Note that the reverse approach also exists, i.e. the construction of a Boolean model based on existing ODE models. [Davidich, Bornholdt 2008b] transformed an ODE model of the yeast cell cycle of [Novak et al. 2001] into a Boolean network model by cleverly 'substituting' the respective normalized rate laws with Boolean logic functions (for Hill-type kinetics for example, this is

straightforward). The derived Boolean model was found to resemble the coarse-grained dynamics of the ODE model.

3.8.1 Fuzzy logical models

Fuzzy logic has recently been applied in order to model biochemical networks: [Zielinski et al. 2008], [Aldridge et al. 2009], [Huang, Hahn 2009]. The description here is based on the summary in [Wittmann et al. 2009a].

The idea is that, based on logical interactions, as also given in the standard definition of Boolean models, the logic is now not only binary or multi-valued but even fuzzified. This means that there is a continuum between true, i.e. 1, and false, i.e. 0. So, the state of a given gene is part of the closed interval $[0,1]$. One interpretation may be that for example a state of 0.5 does not mean something like "the gene is either on nor off but something in between…" but rather something like "the gene is transcribed half maximal" (take transcription, just to be concrete) while a state of 0.95 would mean that the gene is almost transcribed at maximal rate which ultimately would correspond to a state of 1.

Formally, fuzzy logical modeling is done via the introduction of a so called *degree of membership* (*DOM*) *function* for every variable and in addition, a way to fuzzify the involved logic functions [Zadeh 1965]. There are two prominent ways of fuzzification for ordinary logic functions: *min-max logic* and *product-sum logic* [Zadeh 1965], [Wittmann et al. 2009a].

Since every logical function can be expressed in normal form by combining AND, OR and NOT operations via the *conjunctive normal form* (*CNF*, *product-of-sums form*) or the *disjunctive normal form* (*DNF*, *sum-of-product form*), see [Aigner 2006: Subsection 11.2] and Subsection 3.1, one 'only' needs to fuzzify these general representations.

In the following I describe the two mentioned transformation approaches in an intuitive way. There are many more subtleties involved in fuzzy modeling, see for example [Zadeh 1995] or [Sugeno, Yasukawa 1993].

The *min-max logic* is defined via the following transformation:

$$\begin{aligned} x \wedge y &\rightarrow \min\{\bar{x}, \bar{y}\} \\ x \vee y &\rightarrow \max\{\bar{x}, \bar{y}\} \\ \neg x &\rightarrow 1 - \bar{x}\,. \end{aligned}$$

Here, $\bar{x}$ and $\bar{y}$ denote the fuzzyfied 'variables' (actually it is more complex than that but it captures the main point) while x and y denote the original binary Boolean variables. One nice thing about the min-max logic is that the 'outputs' $\bar{x}, \bar{y}$ are automatically contained in the interval $[0,1]$ and identical to the original logic if and only if $\bar{x}, \bar{y} \in \{0,1\}$.

On the other hand, the *product-sum logic* is obtained via the following transformation:

$$\begin{aligned} x \wedge y &\rightarrow \bar{x} \cdot \bar{y} \\ x \vee y &\rightarrow \bar{x} + \bar{y} \\ \neg x &\rightarrow 1 - \bar{x}. \end{aligned}$$

The fuzzification of the OR-gate clearly needs some kind of normalization.

After a fuzzification is chosen, the underlying interaction graph together with the fuzzified logic rules now constitutes a dynamical system which can now be analyzed. For an ensemble-type study of fuzzy logical networks see [Wittmann, Theis 2011].

3.8.2 Standardized qualitative dynamical systems (SQUAD)

[Mendoza, Xenarios; 2006] introduced a method to transform a given Boolean network satisfying certain constraints to a system of ODEs. The constraints appear rather strict since they demand the logic functions to involve a set of activators and inhibitors in such a way that if one of the inhibitors is present, the target is automatically off while if none of the inhibitors is present, only one of the activators has to be present to set the target variable to the on-state. Semi-formally, for a given node i in the network and its parent nodes $j_1, ..., j_k$ the logic function governing the evolution of node i, i.e. its state $x_i \in \{0,1\}$, is given in terms of a relationship of the following form:

$$f_i\left(x_{j_1}, ..., x_{j_k}\right) = \left(\vee_{j_\kappa \in A_i} x_{i_\kappa}\right) \wedge \neg\left(\vee_{j_\kappa \in I_i} x_{i_\kappa}\right).$$

Here, $A_i \dot{\cup} I_i = \{j_1, ..., j_{k_i}\}$ represents a partition of the nodes which influence node i into a set A_i of activating nodes and a set I_i of inhibiting nodes.

For all Boolean networks satisfying the above constraint on their logic functions the standardized qualitative dynamical system associated to that Boolean network is given by a differential equation for the (qualitative and normalized) continuous state $\bar{x}_i \in [0,1]$ for all the respective nodes i. The equations consist of a production term and a degradation term where the degradation term is simply given by a constant times the continuous state. The production term is a complicated (but systematic) combination of the inputs such that the resulting term is sigmoidal and thus may resemble Hill-type kinetic laws. The method of automatically constructing these production terms crucially relies on the above described restrictions concerning the logic function of the underlying Boolean network. See [Mendoza, Xenarios 2006] for details on the exact form and motivation of the respective production terms.

In order to assess whether steady states of the Boolean model correspond somehow to steady states of the automatically derived ODE model [Mendoza, Xenarios 2006] performed (clever) numerical simulations and found that steady states often seem to be preserved. However, this is not generally true. As pointed out by [Wittmann et al. 2009a], examples can be constructed

such that the Boolean steady states do not correspond to continuous steady states of the system derived by the methodology of [Mendoza, Xenarios 2006].

3.8.3 Multivariate Polynomial Interpolation (Odefy)

[Wittmann et al. 2009a] introduced another methodology to automatically transform Boolean network models to systems of ODEs via the use of multivariate polynomial interpolation. In contrast to the standardized qualitative dynamical systems described in the preceding subsection the method is universally applicable and does not rely on assumptions concerning the functional form of the logic functions. Moreover, the preservation of steady states can be formally demonstrated.

Applying multivariate polynomial interpolation [Gasca, Sauer 2000] to transform Boolean logic functions into functions which accept continuous states as their arguments (so called continuous homologues of the Boolean functions) leads to a system of ODEs in a analogous fashion as in [Mendoza, Xenarios 2006] but now the production terms are given by the respective homologues obtained by multivariate polynomial interpolation. [Wittmann et al. 2009] derive three different homologues: BooleCube, HillCube and normalized HillCube homologues. The BooleCube homologues are the basic ones which are obtained by polynomial interpolation, the HillCube homologues are obtained from the BooleCube homologues by means of transforming the continuous variables with Hill functions (see Section 2.1) before subjecting them to the normal BooleCube homologues in order to get the production terms. In contrast to the BooleCube homologues the HillCube homologues are not perfect homologues in the sense that they coincide with the original Boolean functions on the n-dimensional hypercube $\{0,1\}^n$ (n the number of species). By normalizing the Hill functions of the HillCube homologues the normalized HillCube homologues are obtained which are again perfect homologues of the Boolean functions.

[Wittmann et al. 2009a] prove that steady states of the Boolean model are also steady states of the derived homologue ODE system when BooleCube or normalized HillCube homologues are used. For HillCube homologues an asymptotic (with respect to the Hill coefficients) result was proven that guarantees the existence of continuous steady states in certain neighborhoods of the original Boolean model steady states.

The approach was extended to spatial systems by [Wittmann et al. 2009b] and [Hock 2010].

3.9 Boolean models for apoptosis

This section exemplifies the application of Boolean models by means of specific Boolean approaches to *programmed cell death* or *apoptosis*. None of the described models is covered in detail but instead it is tried to give a brief impression on Boolean modelling of real

biochemical systems. Note that there also exist Boolean models for higher order biological systems, see for example [Thakar et al. 2007] where a model for the immune response (in terms of the various immune cells like T- or B-cells etc., see [Purves et al. 2006: Chapter 18]) in reaction to some infection is proposed.

Apoptosis [Kerr et al. 1972], also called programmed cell death, is the process by which individual cells (in multi-cellular organisms) are killed in an ordered and programmed fashion in response to certain stimuli via the activation of corresponding *apoptosis pathways* which involve signal transduction and gene regulation. This is *in contrast to necrosis-type cell death* which is brought about by less intricate ways (in the sense that no elaborate regulatory networks are involved) by mechanical destruction like unbearable tension or pressure, or immediately deathly temperatures. Programmed cell death is important in various biological functions and processes, for example in development. One of the most famous examples are the webs between the digits of human embryos which are removed by apoptosis before birth (in contrast to frogs, for example) [Wolpert et al. 2011: Sections 10.12 and 11.17]. Apoptosis also plays a decisive role to prevent cancer since normally cells with irreversibly damaged DNA receive the internal signal to start their apoptosis program. When this process fails the particular cell and the mutations caused by its damaged DNA might constitute the starting point of cancer. Mutations in genes involved in the regulation of apoptosis ultimately can also be the manifestation of cancer by altering the apoptosis pathway such that normally apoptotic conditions do not induce apoptosis anymore. Several other diseases are associated to the failure of proper apoptosis, for example Parkinson's or Alzheimer's disease [Danial, Korsmeyer 2004]. While an original (and ultimately apoptosis inducing) apoptotic pathway exists, the regulation of apoptosis is highly complex and involves the integration of various cell-internal (DNA damage, for example) and cell-external signals (molecules from neighbouring cells for example) and integrates also the interplay of different signalling and gene regulatory pathways. The NFκB pathway described in Section 1.1 for example has several outputs which operate directly as inputs for the actual apoptosis pathway [Chaves et al. 2009].

For a review on the detailed biological facts I recommend [Danial, Korsmeyer 2004] (in the sense of partial accessibility for non-biologists with non-systematic biological knowledge), for a short overview see [Klipp et al. 2009: Section 3.5].

Besides Boolean and logical models a number of other modelling approaches were of course also used to study apoptosis. For ODE models (see Section 2.1) see for example [Klipp et al. 2009: Section 3.5] and the references therein. For Petri net approaches (see Section 4.2) see [Heiner et al. 2004] and [Li et al. 2007].

[Schlatter et al. 2009] manually developed a Boolean model that integrates several of the key pathways involved in the regulation of apoptosis on the basis of existing literature and analyzed it with CNA (CellNetAnalyzer) [Klamt et al. 2007]. Concerning the updating scheme the *timescale approach* described was used in Section 3.1 which proceeds by defining different timescales which determine the interactions (according to their specific scales) which are active and can therefore influence their respective target nodes. Then, the

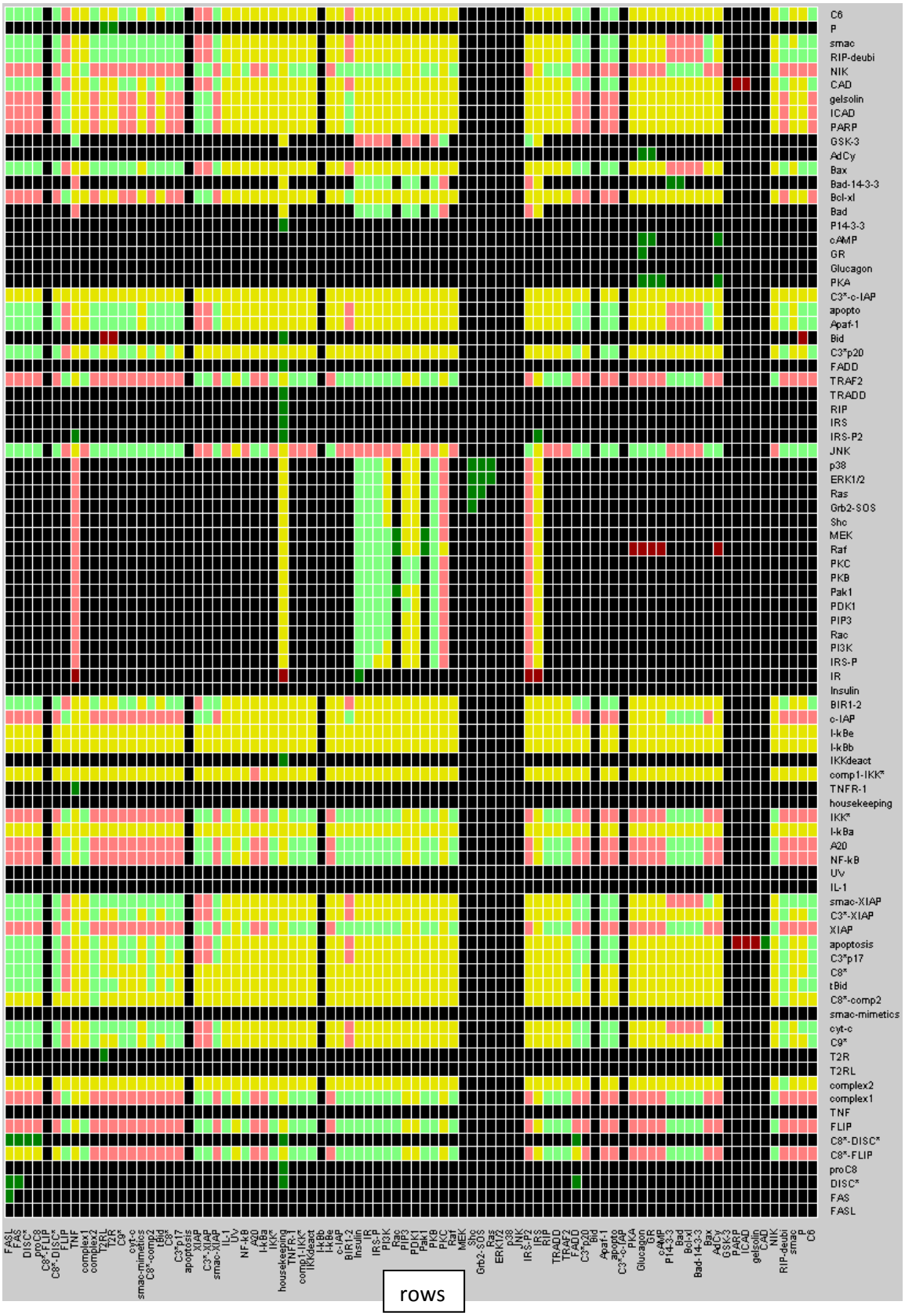
assessment the system-wide influences by the *dependency matrix* approach for every separate timescale ([Klamt et al. 2006] and Subsection 1.5.1) was constructed and the computation of the *logical steady-states* ([Devloo et al. 2003] and Section 3.1) under different network structures and knock-out scenarios was performed. A further feature of the model is the use of *multi-valued logical states* motivated by the existence of different thresholds associated to the effect of some nodes (see Section 3.5 for the explanation for the threshold motivated introduction of multi-valued logical states into Boolean modelling due to [Van Ham 1979]). The dependency matrix for time scale 10 is shown in figure 3.1 (for aesthetical reasons).

[Mai, Liu 2009] manually developed another Boolean network model which also integrates various pathways involved in apoptosis and performed its analysis by simulating the dynamics starting from 10,000 *sampled initial states* (the network has 40 nodes and hence $2^{40} \approx 10^{12}$ initial states are possible, so they covered 0.000001 % of all possible initial states). The network is based on a directed signed interaction graph and synchronous updates with classical weighted threshold-sum logical functions (see Subsection 3.2.1).

[Zhang et al. 2008] study a Boolean model of apoptosis signalling in so called CTL cells (cytotoxic T lymphocytes; a certain cell type of the immune system, see [Purves et al. 2006: Chapter 18] in which the malfunctioning of apoptotic signalling causes the so called T-LGL leukaemia (T cell large granular lymphocyte leukaemia), a type of cancer. By examination of their manually created Boolean model they arrived at suggesting various biological conclusions. For example, they identified two species (IL-15 and PDGF) which when constantly in the on-state (i.e. when their states are kept constantly at state 1 during dynamic examination of the Boolean model) led to a Boolean dynamics that captures all known irregularities associated to T-LGL. One aspect of the model which represents a general strategy in Boolean modelling is the incorporation of so called *abstract nodes* representing complex processes instead of single genes or molecules. In the example here, these were for example "proliferation" or "apoptosis" (as medically decisive output variables) or "cytoskeleton signalling" representing a bunch of processes interacting with other nodes of the network which are not all explicitly modelled but condensed into one abstract variable. Concerning the update scheme the authors adopted asynchronous updates in order to cope with different timescales. They evaluated the appropriate timescale regime via *update sampling*. More precisely the update for node i is defined by $x_i^{t+1} = f_i\left(x_{i_1}^{\tau_{i_1}(t+1)}, ..., x_{i_{k_i}}^{\tau_{i_{k_i}}(t+1)}\right)$ with $\tau_{i_j}(t+1) \in \{t, t+1\}$ for every i_j such that the decision whether $\tau_{i_j}(t+1) = t$ of $\tau_{i_j}(t+1) = t+1$ is determined by a partition of the nodes as described in section 3.1, i.e. given a permutation s of the nodes one has $\tau_{i_j}(t+1) = t+1$ if and only if $s(i) > s(i_j)$ for all i and i_j. In order to sample update schedules, permutations of the nodes are uniformly sampled. Dynamical simulations of the Boolean network according to every of the sampled schedules are then examined in regard to their ability to capture features of the modelled biological system. The authors also mention that the described sampling in conjunction with initial state sampling might also be interpreted as a means of simulating cell-to-cell variability in heterogeneous cell populations. **[Saadadpour et al. 2011]** analyzed the model of [Zhang et al. 2008] further.

Figure 3.1: Dependency matrix of the Boolean network of [Schlatter et al. 2009](for timescale 10)

Legend: dark green: total activation, dark red: total inhibition, yellow: ambiguous effect
black: no influence, light green: activation, light red: inhibition (see Subsection 1.5.1)



[Kazemzadeh et al. 2012] developed a literature- and database-based model for the apoptosis network in the yeast S.cerevisae adopting the timescales approach and first analyzed it with CNA (steady-state, dependency matrix). In addition they then *transformed their model into a continuous ODE model* via two different existing software tools (see Chapter 4). First they used the SQUAD software based on the so called standardized qualitative dynamical systems transformation method [Mendoza, Xenarios 2006] (see Subsection 3.8.2) and second they used Odefy [Krumsiek et al. 2010] which is, as seen in Subsection 3.8.3, based on multivariate polynomial interpolation of the logic functions in conjunction with Hill-type transformations of the resulting continuous state variables [Wittmann et al. 2009].

Table 3 in [Kazemzadeh et al. 2012] gives an instructive impression how much literature-based reconstruction is involved in the process of assembling the logical rules for a Boolean model and there are even more extensive compilations, see for example the supporting material of [Zhang et al. 2008]. In parallel, this underlines the (trivial) fact that in order to be able to build up a Boolean model of any biological process one relies on the laborious work of many experimental researchers. I counted 29 ± 2 distinct experimental sources leading to ~ 85 interactions. Note, that the model possesses in summary even 115 logical rules but some of them describe (controllable or constant) inputs or species/processes constantly present (termed 'housekeeping').The other way round, once a core Boolean network is given one can try out the effects of different modifications of the network like the addition of specific hypothesized interactions and compare the dynamical consequences to further experimental evidence in order to ideally elucidate new interactions in the actual biological network. Ideally of course, researchers try to reverse-engineer Boolean models automatically entirely from scratch based on suitable largely non-interpreted (!) experimental data (see Subsection 1.5.2).

[Chaves et al. 2009]. Based on a manually constructed Boolean network describing the interplay of the NFκB and the apoptosis pathway several model extensions (in this case, the addition of regulatory influences between nodes) where evaluated according to their capability to explain experimental data. The analysis relies on the PLDE formalism described in Section 3.6 which combines an underlying Boolean regulatory logic with continuous variables governed by ordinary differential equations which turn out to be linear (affine) on domains in the state space which are associated to respective logical states defined by thresholds.

4 Parameter estimation for GKL networks with probabilistic time delays

In this section I propose two choices for distributions on the time delays of a GKL network (see Section 3.5 on GKL). The first choice assumes exponentially distributed time delays as introduced by [Teraguchi et al. 2011]. After evaluating the appropriateness of this choice in terms of the biological systems which are thought to be modeled by the formalism I conclude that exponential distributions are not the best choice and hence I propose to use Weibull-distributed delays. The motivation for the postulation of time delays in general and for the particular choices made is outlined in Section 4.1. The next section is then concerned with the task of statistically estimating the parameters of the delay distributions based on mostly hypothetical and simulated but ideally experimental data. Section 4.2 addresses parameter estimation in the light of data which is given by absorption frequencies to the respective logical attractors of the network. Subsection 4.3 finally makes some final remarks concerning the interpretation and intuition of the two models.

4.1 Model philosophy and specification

In Subsection 3.5.2 I reviewed and adapted the formal definition of timed GKL networks. The idea of introducing probability distributions for the time-delays is almost as old as the field of GKL itself: [Thomas (ed.) 1979], [Thomas 2013]. The idea also has been implemented in some instances with uniform distributions over intervals, for example in [Thomas et al. 1979] [Abou-Jaoudé et al. 2009]. In [Abou-Jaoudé et al.; 2009] the authors used uniform distributions for the time-delays to investigate a model for p53 activity where in addition time-delays are considered context-sensitive, i.e. time delays are in addition to being stochastic not constant in the sense that they differ in dependence of the present state of the entire system. Context-sensitive time-delays were also considered by [Siebert, Bockmayr 2009]. Here, time delays are assumed to be context-independent, i.e. they follow distributions but these distributions are always the same irrespectively of the state of the system.

Assume a timed GKL network which is thought to model a pure GRN with only genes and TFs. Then the state of the system would correspond to gene activity and the images indicate the regulatory evolution of the system based on the present actual state. Let for example the state of gene i mismatch its image. Then the active time delay will represent the duration of all the possibly diverse processes which have to happen in order to bring about the desired change of the state of gene i. In the simplest case, imagine i to be regulated by another gene k such that given that gene i is off and gene k is on, gene i will be activated after the characteristic time delay. Then, this time delay represents the duration of the processes of mRNA transcription from gene k, export of that mRNA to the cytoplasm, translation of the mRNA to a TF, the importing of the TF to the nucleus and finally the process of regulation of gene i by the TF itself. In addition the time delay not just describes the duration of these

processes but rather the "accumulation" of these such that the TF finally reaches a critical concentration threshold necessary for the up-regulation of gene i. This is only the simplest case. If the regulatory process involves heterotypic multimers of TFs for example the respective time delay describes the effective duration of several process chains as described together such that it finally represents the time-scale on which these many diverse processes together bring about their regulatory effect. This means that the time delay (stochastic or non-stochastic) describes very heterogeneous processes and it therefore seems questionable to postulate a single family of probability distributions to describe them. Nonetheless, this approach is taken here.

Based on the definitions of Subsection 3.5.2 I first precisely define the introduction of probabilistic time-delays into timed GKL networks. At this stage no restriction on the time delay distributions is imposed. Let $\mathcal{M}_1[0,\infty)$ denote the set of probability measures on $[0,\infty)$.

Definition 4.1.1 (**Probabilistic timed GKL network**)

A *probabilistic timed GKL network* is a tuple $\Gamma = (N, \mathfrak{T})$ such that $N = (V, \eta, f)$ is a GKL network and $\mathfrak{T} := \{\tau_i : [p_i] \times \{-1,+1\} \to \mathcal{T}(\mathcal{M}_1[0,\infty)) : i \in V\}$ is a family of maps with the range $\mathcal{T}(\mathcal{M}_1[0,\infty))$ being a set of random variables (defined on some probability space) which are distributed according to a measure from $\mathcal{M}_1[0,\infty)$ respectively.

For $i = 1,...,n$ and $\sigma_i \in [p_i]$ the random variable $\mathfrak{T}(\sigma_i, s) \in \mathcal{M}_1[0,\infty)$ describes the time delay which is responsible for switching variable i from $\sigma_i - 1$ to σ_i if $s = +1$ and from σ_i to $\sigma_i - 1$ if $s = -1$. In the following, let $\tau_i^{(\sigma_i)} := \mathfrak{T}(\sigma_i, +1)$ and $\bar{\tau}_i^{(\sigma_i)} := \mathfrak{T}(\sigma_i, -1)$. To specify a probabilistic GKL network one has to specify the distributions of the random variables $\tau_i^{(\sigma_i)}$ and $\bar{\tau}_i^{(\sigma_i)}$ for all $i = 1,...,n$ and all $\sigma_i \in [p_i]$. All definitions made in Subsection 3.5.2 which do not concern time delays, like that of attractors or the state transition graph, can be directly transferred to the case of probabilistic timed GKL networks by defining the respective notions in terms of the GKL network N which is part of the tuple constituting a probabilistic timed GKL network. See definitions 3.5.2 and 3.5.3.

Before proceeding to the definition of special model types defined by specific assumptions concerning the distribution of the delays, the general definition of the dynamics of probabilistic timed GKL networks is given. The following algorithm naturally makes use of the formal definitions of Subsection 3.5.2 on timed GKL networks, for example of the

switching map $\phi^{(q)} = \left(\phi_1^{(q)}, \phi_2^{(q)}, \phi_3^{(q)}\right) : \mathcal{C}_{STG}(q) \to V \times \bigcup_{j \in V} \left[p_j\right] \times \{-1, +1\}$ for states $q \in \mathcal{Q}$ and the corresponding *set of active delays* $\mathcal{D}(q) := \left\{\tau_{\phi_1^{(q)}(\tilde{q})}\left(\phi_2^{(q)}(\tilde{q}), \phi_3^{(q)}(\tilde{q})\right) : \tilde{q} \in \mathcal{C}_{STG}(q)\right\}$. Here, as in Subsection 3.5.2, $\mathcal{C}_{STG}(q) := \left\{\tilde{q} \in \mathcal{Q} : (q, \tilde{q}) \in E\right\}$ is the set of potential successors of state q in the state transition graph. Further remember that the definition of the *set of active species* relative to state $q \in \mathcal{Q}$ is given by $\mathcal{A}(q) := \left\{\phi_1^{(q)}(\tilde{q}) \in V : \tilde{q} \in \mathcal{C}_{STG}(q)\right\}$. Since the algorithm again essentially computes a sequence $\left(q^{(\omega)}\right)_\omega \in \mathcal{Q}^{\Omega+1}$ with $\Omega \in \mathbb{N}_0$ being the number of switches that occurred, we can again simplify the notation and set $\phi^{(\omega)} := \phi^{\left(q^{(\omega)}\right)}$ for all $\omega \in \mathbb{N}_0$. The algorithm presented below resembles the one given in Subsection 3.5.2 (algorithm 3.5) for timed GKL networks. But in addition, algorithm 4.1 below also has to deal with variable time delays in the sense that delays are probabilistic according to definition 4.1.1. This will lead to a successive conditioning on events concerning the duration of a given delay until the state of the associated variable finally switches, coincides with its image again or the sign of the state-image difference changes without the variable having switched since its activation. As in algorithm 3.5 a *switching delay* $\alpha : V \times \mathbb{N}_0 \to \mathbb{N}_0$ will be used where $\alpha(i, \omega) = \kappa$ represents the fact that after the ω-th overall switching event, species i is still active with delay $\tau_i\left(q_i^{(k-\kappa)} + 1, +1\right)$ or $\tau_i\left(q_i^{(k-\kappa)}, -1\right)$ without having switched or a qualitative change in the respective image having taken place. Formally, we have for a state $q \in \mathcal{Q}$ the *active delay association map* for $\omega \in \mathbb{N}_0$ as

$$\tau^{(\omega)} : \mathcal{A}(\omega) \to \mathcal{D}(\omega),\ \phi_1^{(\omega)}(\tilde{q}) \to \tau_{\phi_1^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right)$$

where $\mathcal{A}(\omega) := \mathcal{A}\left(q^{(\omega)}\right) = \left\{\phi_1^{(\omega)}(\tilde{q}) \in V : \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right)\right\}$ is the set of active species in state $q^{(\omega)} \in \mathcal{Q}$ and where $\mathcal{D}(\omega) := \mathcal{D}\left(q^{(\omega)}\right) := \left\{\tau_{\phi_i^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right) : \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right)\right\}$ is the set of active delays in $q^{(\omega)} \in \mathcal{Q}$.

Algorithm 4.1 [**Simulation of probabilistic timed GKL networks**]

Input: I.1 $\Gamma = (N, \mathfrak{T})$ # a probabilistic timed GKL network

with, $N = (V, \eta, f)$ being a GKL network and $\tau_i^{(\sigma_i)}$ and $\overline{\tau}_i^{(\sigma_i)}$ being

the time delays associated to threshold $\sigma_i \in \left[p_i\right]$ of species $i \in V$ (see above)

I.2 $q^{(0)} \in \mathcal{Q}$ # initial state

I.3 $t_{max} \in [0, \infty)$ # maximal simulation time

Output: O1. $Q_{q^{(0)}} : [0, t_{max} + \varepsilon] \to \mathcal{Q}$ # continuous time evolution for some $\varepsilon > 0$

O2. $\left(\xi_{\omega}^{\left(q^{(0)}\right)} \right)_{\omega} \in \mathcal{Q}^{\Omega+1}$ # associated jump process, $\Omega \in \mathbb{N}_0$ is the number of switches which have taken place

---

(1) $\forall i \in V: \alpha(i,0) = 0$ # initialization of switching delays

(2) $\xi_{q^{(0)}}(0) \leftarrow q^{(0)}$ # initialization of jump process

(3) $t \leftarrow 0$ # initialization of continuous time

(4) $\omega \leftarrow 0$ # discrete jump time

(5) $\mathcal{A}(-1) \leftarrow \varnothing$ # "active states before start" (technically necessary)

(6) while $t \le t_{max}$:

(5.1) $\mathcal{C}_{STG}\left(q^{(\omega)}\right) := \left\{ \tilde{q} \in \mathcal{Q} : \left(q^{(\omega)}, \tilde{q}\right) \in E \right\}$ # successor states, this can be done by computing the image, i.e. the STG has not to be computed a priori!

(5.2) $\phi^{(\omega)} = \left(\phi_1^{(\omega)}, \phi_2^{(\omega)}, \phi_3^{(\omega)}\right) : \mathcal{C}_{STG}\left(q^{(\omega)}\right) \to V \times \bigcup_{j \in V} \left[p_j\right] \times \{-1, +1\}$

compute the switching map

(5.3) $\mathcal{A}(\omega) := \left\{ \phi_1^{(\omega)}(\tilde{q}) \in V : \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right) \right\}$ # active species

(5.4) $\mathcal{D}(\omega) = \left\{ \tau_{\phi_1^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right) : \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right) \right\}$ # active delays

(5.5) $\tau^{(\omega)} : \mathcal{A}(\omega) \to \mathcal{D}(\omega),\ \phi_1^{(\omega)}(\tilde{q}) \to \tau_{\phi_1^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right)$

active delay association map

(5.6) for $\tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right)$ with $\phi_1^{(\omega)}(\tilde{q}) \in \mathcal{A}(\omega) \cap \mathcal{A}(\omega - 1)$:

update switching delays for active delays

(5.6.1) if $\tau^{(\omega)}\left(\phi_1^{(\omega)}(\tilde{q})\right) = \tau^{(\omega-1)}\left(\phi_1^{(\omega)}(\tilde{q})\right)$:

(5.6.1.1) $\alpha\left(\phi_1^{(\omega)}(\tilde{q}), \omega\right) \leftarrow \alpha\left(\phi_1^{(\omega)}(\tilde{q}), \omega - 1\right) + 1$

$\phi_1^{(\omega)}(\tilde{q})$ is active with same delay as before, hence the switching delay for the respective delay has to be updated

(5.6.2) else:

(5.6.2.1) $\alpha\left(\phi_1^{(\omega)}(\tilde{q}), \omega\right) \leftarrow 0$

$\phi_1^{(\omega)}(\tilde{q})$ is active with new delay, only possible if the difference of state and image of the species changed sign with the switch from $q^{(\omega-1)}$ to $q^{(\omega)}$

(5.7) for $i \in V \setminus \mathcal{A}(\omega)$:

(5.7.1) $\alpha(i,\omega) \leftarrow 0$ # "switching" delays of non-active species

(5.8) for $i \in \mathcal{A}(\omega)$:

(5.8.1) $\kappa \leftarrow \alpha(i,\omega)$ # switching delay

(5.8.2) $\Sigma_i \leftarrow \sum_{\ell=\omega-\kappa+1}^{\omega} t_\ell$ # time since activation of delay

$$\left(\sum_{\ell=a}^{b} t_\ell := 0 \text{ for all } a > b\right)$$

(5.8.3) $\tau_i \leftarrow$ sample from $\tau^{(\omega)}(i) - \Sigma_i \,\big|\, \tau^{(\omega)}(i) > \Sigma_i$

sample from delay conditioned on time passed since activation

(5.9) $s \leftarrow \arg\min_{i \in \mathcal{A}(\omega)} \tau_i$ # switching species

(5.10) $t_\omega \leftarrow \min_{i \in \mathcal{A}(\omega)} \tau_i$ # time to next switch

(5.11) $q^{(\omega+1)} \leftarrow \tilde{q}$ with $s = \phi_1^{(\omega)}(\tilde{q}),\ \tilde{q} \in \mathcal{C}_{STG}\left(q^{(\omega)}\right)$

next state according to switching species

(5.12) $\xi_{q^{(0)}}(\omega+1) \leftarrow q^{(\omega+1)}$ # update of discrete jump output

(5.13) $\forall \tau \in [t, t+t_\omega): Q_{q^{(0)}}(\tau) \leftarrow q^{(\omega)}$ # update of continuous-time output

(5.12) $t \leftarrow t + t_\omega$ # update of continuous time

(5.13) $\omega \leftarrow \omega + 1$ # update of jump time

The output $\left(\xi_\omega^{\left(q^{(0)}\right)}\right)_\omega = (\xi_\omega)_\omega \in \mathcal{Q}^{\Omega+1}$ can be seen as a realization of a time-discrete stochastic process. This process is in the following denoted as $(\Xi_\omega)_{\omega \in \mathbb{N}_0}$.

Now that it is theoretically clear how to simulate probabilistic timed GKL networks one can look at specific types of networks according to different distributions for the delays. In the following I look at two types of probabilistic timed GKL networks. The first one involves only time delays which are distributed according to an exponential distribution and the second extends the first framework such that the delays can be distributed according to a Weibull distribution. See Appendix A.1 for some basic definitions and facts concerning these distributions. In the first case it turns out that the simulation comes down to simulating the jump chain of a continuous-time Markov jump process by just simulating suitable exponential distributions, see below. In the second case (and in the general case) the ability to simulate the corresponding probabilistic timed GKL network crucially depends on the ability to efficiently simulate the delay variables given that the delay is already active for a certain time ((5.8.3) in algorithm 4.1). It is shown in Appendix A.3 that in the case of Weibull distributed time delays

the demanded task is feasible by means of adaptive rejection sampling (ARS). Hence, one can actually simulate both model classes which are introduced in the following.

The first model, model $\mathbb{M}_{\text{Exp}}$ is thus defined to be a probabilistic timed GKL $\Gamma = (N, \mathfrak{T})$ with time delays specified by $\tau_i^{(\sigma_i)} \sim \text{Exp}\left(\lambda_i^{(\sigma_i)}\right)$ and $\overline{\tau}_i^{(\sigma_i)} \sim \text{Exp}\left(\overline{\lambda}_i^{(\sigma_i)}\right)$ for parameters $\lambda_i^{(\sigma_i)}$, $\overline{\lambda}_i^{(\sigma_i)} > 0$ for all $i = 1, ..., n$ and $\sigma_i \in [p_i]$.

One characteristic (actually defining) property (see Appendix A.1) of the exponential distribution is the memoryless property: if $T \sim \text{Exp}(\lambda)$ with $\lambda > 0$ we have for every $t > 0$ that $T - t \,|\, T > t \sim \text{Exp}(\lambda)$. This means that if we know that the event modeled via its exponential waiting time has not occurred until time t the distribution of the waiting time is again the same exponential distribution. This especially means that the mean waiting time is always the same regardless of the time one is already waiting. This might seem biologically implausible since the processes modeled with the introduction of time delays are real physical processes which are in some sense "accumulative", namely transcription, molecular transport, translation, etc. These processes are "accumulative" in the sense that they are not memoryless: the universe will remember if some mRNAs were transcribed (and not degraded afterwards) simply because they are still there. Now, assuming that some threshold has to be passed by, say, the number of mRNAs to make the event happen which is described by the time delay it seems therefore implausible that the delay can be modeled to be memoryless. Nevertheless, we will for mathematical simplicity adopt the assumption of exponential time delays. One advantage of this assumption is that the evolution of the states can now be framed in the formalism of Markov chains [Norris 1998]. Markov chains have already been probed for their capabilities to describe biological regulatory networks [Kim et al. 2002], [Tournier, Chaves 2009].

If one takes a look at step (5.8.3) of algorithm 4.1 (and also recalls the notation from there), one realizes that for exponentially distributed delays $\tau^{(\omega)}(i) \sim \text{Exp}(\lambda)$, sampling from $\tau^{(\omega)}(i) - \Sigma_i \,|\, \tau^{(\omega)}(i) > \Sigma_i$ just comes down to sampling from $\tau^{(\omega)}(i)$ itself because of the memoryless property of the exponential distribution (see Result A.1 in Appendix A.1). This means in particular that the history of the system, encoded in the time Σ_i since activation of the delay, is not relevant for the probabilistic behavior of the system at every jump and hence the resulting jump dynamics can be framed as a time-homogenous (delay laws are always the same), time-discrete Markov chain $(\Xi_\omega)_{\omega \in \mathbb{N}_0}$. More precisely, let $\mathcal{A}(\omega) := \left\{\phi_1^{(\omega)}(\tilde{q}) \in V : \tilde{q} \in \mathcal{C}_{\text{STG}}\left(q^{(\omega)}\right)\right\}$ the set of active species at (jump) time $\omega \in \mathbb{N}_0$, then the probability that a particular active species $\phi_1^{(\omega)}(q')$ switches its state is given by the

$$\frac{\lambda\left(\tau_{\phi_1^{(\omega)}(q')}\left(\phi_2^{(\omega)}(q'), \phi_3^{(\omega)}(q')\right)\right)}{\sum_{\tilde{q} \in \mathcal{C}_{\text{STG}}\left(q^{(\omega)}\right)} \lambda\left(\tau_{\phi_1^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right)\right)}$$

where $\lambda\left(\tau_{\phi_1^{(\omega)}(\tilde{q})}\left(\phi_2^{(\omega)}(\tilde{q}), \phi_3^{(\omega)}(\tilde{q})\right)\right)$ canonically denotes

the parameter of an active delay of an active species $\phi_1^{(\omega)}(\tilde{q})$. This fact is due to Result A.2 in Appendix A.1 which gives the probability for the event that a given exponentially distributed variable is minimal with respect to a family of independent exponentially distributed random variables. The above expression gives the probability that, given the system is in state $q^{(\omega)}$ at jump $\omega \in \mathbb{N}_0$, the next state will be the one which is reached if the active species $\phi_1^{(\omega)}(q')$ switches, i.e. the probability of the transition $q^{(\omega)} \to q'$. Since this probability only depends on $q^{(\omega)}$ (because of the memoryless property) this show that the jump process of a probabilistic timed GKL network with exponentially distributed time delays is indeed a Markov chain with the transition matrix $P = (p_{qq'})_{q,q' \in Q} \in [0,1]^{2^n \times 2^n}$ (see Appendix A.4) given by $p_{qq'} = 0$ if $q' \notin \mathcal{C}_{STG}(q)$ and $p_{qq'} = \dfrac{\lambda\left(\tau_{\phi_1^{(q)}(q')}\left(\phi_2^{(q)}(q'), \phi_3^{(q)}(q')\right)\right)}{\sum_{\tilde{q} \in \mathcal{C}_{STG}(q)} \lambda\left(\tau_{\phi_1^{(q)}(\tilde{q})}\left(\phi_2^{(q)}(\tilde{q}), \phi_3^{(q)}(\tilde{q})\right)\right)}$ if $q' \in \mathcal{C}_{STG}(q)$ (see the next section for concrete examples).

But as indicated before, the assumption of exponentially distributed time delays seems inappropriate due to the *accumulative character* [Ahmad et al. 2007] of the processes which determine the switching events which are mainly given by accumulation until a certain threshold is exceeded. The general idea on how to overcome the described memoryless implausibility is to introduce distributions with increasing "hazard" rates. This simply means that the event described by the time delay becomes more and more probable if more time elapses. This can lead to the choice of Weibull distributions with shape parameters greater than one (see Appendix A.1 for the meaning of the shape parameter and Appendix A.5 for an illustration of the meaning associated to such distributions).

Model $\mathbb{M}_{WB}$ therefore finally is defined to be a probabilistic timed GKL $\Gamma = (N, \mathfrak{T})$ with time delays specified by $\tau_i^{(\sigma_i)} \sim \mathcal{WB}\left(\lambda_i^{(\sigma_i)}, k_i^{(\sigma_i)}\right)$ and $\bar{\tau}_i^{(\sigma_i)} \sim \mathcal{WB}\left(\bar{\lambda}_i^{(\sigma_i)}, \bar{k}_i^{(\sigma_i)}\right)$ for scale parameters $\lambda_i^{(\sigma_i)}$, $\bar{\lambda}_i^{(\sigma_i)} > 0$ and shape parameters $k_i^{(\sigma_i)}$, $\bar{k}_i^{(\sigma_i)} \geq 1$ for all $i = 1,...,n$ and $\sigma_i \in [p_i]$. $\mathbb{M}_{WB}$ actually contains $\mathbb{M}_{Exp}$ since $\mathbb{M}_{WB}$ models where all shape parameters are equal to one correspond to an instance of model $\mathbb{M}_{Exp}$ (see Appendix A.1).

Concerning step (5.8.3) of algorithm 4.1, Appendix A.5 shows how to sample from $\tau - \Sigma \,|\, \tau > \Sigma$ for $\Sigma \geq 0$ when $\tau \sim \mathcal{WB}(\lambda, k)$.

Some canonical differences which can arise between the models $\mathbb{M}_{WB}$ and $\mathbb{M}_{Exp}$ are shortly indicated in Section 4.3. The next Section deals with the estimation of the parameters λ for model $\mathbb{M}_{Exp}$ based on absorption frequencies to steady states.

4.2 Parameter estimation based on absorption frequencies

In this section, a simple procedure to estimate the parameters of an instance of $\mathbb{M}_{Exp}$ is proposed which only relies on basic Markov chain theory [Norris 1998] and which is based on absorption frequencies to steady states.

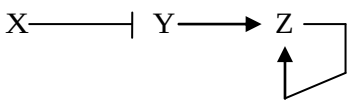
A steady state (or fixed point) is the simplest kind of attractor a logical model can possess (see Section 3.1 and definition 3.5.3). The following considerations are restricted to such steady state attractor but the procedure just as well applies (in theory) to the case of more general attractors as defined in definition 3.5.3. In addition, the described example only deal with GKL networks where every species has only one threshold, i.e. with classical 0-1 networks (or classical asynchronous Boolean networks). But this is also not a principal constraint and the indicated methods (in theory) do not rely in this particular situation.

I start with a simple example dealing with a toy circuit from [Thomas 2013].

4.2.1 A simple example

This first example is from [Thomas 2013]. Everything that goes beyond the mere model formulation up to image functions and the state transition graph are of course already contained in [Thomas 2013]. The postulation of exponential time delays was already considered by [Teraguchi et al. 2011] and [Stoll et al. 2012], but explicitly in the framework of Thomas' GKL networks. The proposed procedure to estimate the parameters is new (although not very fancy and basically just based on very basic Markov chain theory).

Consider the following simple regulatory circuit consisting of three species



i.e. species X inhibits species Y which activates a third species Z which also activates itself.

Formalizing this system in the sense of the GKL approach described above, we assign to each species the state of its gene (denoted by the lower case letters x, y, z) and the image of the corresponding gene state (denoted by the upper case letters X, Y, Z)[10]. The unique image of a given state is supposed to be determined by the following logic functions:

$$X = \text{const.} \in \{0,1\}$$
$$Y = \neg x$$
$$Z = y \vee z$$

[10] X, Y and Z thus denote both, the species and the images of the states of the respective species.

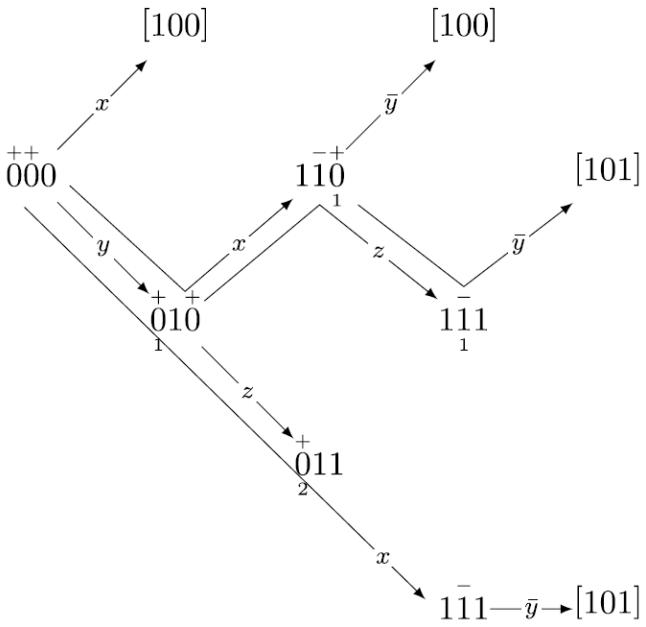
The image X of species associated with x is just arbitrarily set to 0 or 1 what can be interpreted as the (maybe artificial) knock-down or constitutive expression of the gene associated to x. In the following we arbitrarily set $X = 1$.

For every possible state there is a unique image according to the above logic functions:

xyz	XYZ
$\overset{+\,+}{0}00$	110
100	**100**
$\overset{+\quad+}{0}10$	111
$\overset{+\,+}{0}01$	111
$\overset{+}{0}11$	111
101	**101**
$\overset{-\,+}{1}10$	101
$\overset{-}{1}11$	101

The plus or minus signs indicate the active species and whether the corresponding switch of the variable would demand an on-switch or an off-switch. The rows where states and images are equal correspond to steady states.

The state transition graph is given as follows (from [Thomas 2013]):



The small numbers below the states give the number of steps (switching events) the corresponding call for a switch is already active. According to the preceding section they are of course irrelevant when dealing with exponentially distributed time delays. But in Section

4.3 the behavior of the above GKL network is compared for exponential and Weibull delays and then the respective number indicating the history of the system become crucial. The arrows in the above graph are signed according to the corresponding switching event, i.e. if a transition happens via the on-switch of species X the arrow is signed with "x" while if the transition happens via the off-switch of species X the arrow is signed with "$\bar{x}$". So far the example described in [Thomas 2013].

Let us now transform the above GKL network into a probabilistic timed GKL network by assuming it to be part of the model class $\mathbb{M}_{\text{Exp}}$, i.e. we assume (independent) exponentially distributed time delays. For simplicity, denote the parameters for the respective distributions as $\lambda_x, \lambda_{\bar{x}}, ..., \lambda_{\bar{z}} > 0$ where λ_x and $\lambda_{\bar{x}}$ for example are understood to denote the parameters of the up-switching delay distribution and the down-switching distribution for species x respectively. Then, after indexing the states as follows

q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8
000	100	010	001	011	101	110	111

we can set up the transition matrix $P = \left(p_{q_i q_j}\right)_{1\le i,j\le 8} = \left(p_{ij}\right)_{1\le i,j\le 8} \in [0,1]^{8\times 8}$ of the corresponding Markov chain as described in the preceding section.

For example, if the system is in state $q_1 = 000$, the active species are species 1 and 2 with corresponding successor states $q_2 = 100$ and $q_3 = 010$ and hence one obtains $p_{12} = \frac{\lambda_x}{\lambda_x + \lambda_y}$, $p_{13} = \frac{\lambda_y}{\lambda_x + \lambda_y}$ and $p_{1k} = 0$ for all $k \in \{1,...,8\} \setminus \{2,3\}$. Proceeding analogously with the other rows of the transition matrix one finally gets

$$P = \begin{pmatrix} 0 & \frac{\lambda_x}{\lambda_x+\lambda_y} & \frac{\lambda_y}{\lambda_x+\lambda_y} & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\lambda_z}{\lambda_x+\lambda_z} & 0 & \frac{\lambda_x}{\lambda_x+\lambda_z} & 0 \\ 0 & 0 & 0 & 0 & \frac{\lambda_y}{\lambda_x+\lambda_y} & \frac{\lambda_x}{\lambda_x+\lambda_y} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{\lambda_{\bar{y}}}{\lambda_{\bar{y}}+\lambda_z} & 0 & 0 & 0 & 0 & 0 & \frac{\lambda_z}{\lambda_{\bar{y}}+\lambda_z} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

For $A \subset \{q_1,...,q_8\}$ define $h_i^A := P\left(\bigcup_{\omega \in \mathbb{N}_0} \{\Xi_\omega \in A\} \middle| \Xi_0 = s_i \right)$. h_i^A is the probability that the Markov chain reaches one of the states contained in the subset of states A at some point in time if the chain is started in state q_i and is also called hitting probability. For $A = \{q_k\}$ for $k = 1,...,8$ we write $h_i^k := h_i^A$. Since $q_2 = 100$ and $q_6 = 101$ are absorbing states of the Markov chain the probabilities h_i^2 and h_i^6 represent the respective absorption probabilities.

Biologically the absorption probabilities could for example correspond to the frequencies of the respective steady state behaviors within a heterogeneous cell population and therefore it seems somehow reasonable to assume that h_i^2 and h_i^6 can be measured at least in principle.

There is a straightforward way to calculate the so called hitting probabilities h_i^A for given A and transition matrix $P = (p_{ij})_{ij}$ [Norris 1998]. The vector of hitting probabilities $(h_i^A)_i$ can be obtained as the minimal solution to the following system of equations (Appendix A.4):

$$\begin{aligned} h_i^A &= 1 && \text{for all } i \in A \\ h_i^A &= \sum_j p_{ij} h_j^A && \text{for all } i \notin A \end{aligned}$$

Applying this procedure to $A = \{q_2 = 100\}$ we obtain the following system of equations:

$$\begin{aligned} h_1^2 &= \frac{\lambda_x}{\lambda_x + \lambda_y} h_2^2 + \frac{\lambda_y}{\lambda_x + \lambda_y} h_3^2 = \frac{\lambda_x}{\lambda_x + \lambda_y} + \frac{\lambda_y}{\lambda_x + \lambda_y} h_3^2 \\ h_2^2 &= 1 \\ h_3^2 &= \frac{\lambda_z}{\lambda_x + \lambda_z} h_5^2 + \frac{\lambda_x}{\lambda_x + \lambda_z} h_7^2 = \frac{\lambda_x}{\lambda_x + \lambda_z} h_7^2 \\ h_4^2 &= \frac{\lambda_y}{\lambda_x + \lambda_y} h_5^2 + \frac{\lambda_x}{\lambda_x + \lambda_y} h_6^2 = 0 \\ h_5^2 &= 0 \\ h_6^2 &= 0 \\ h_7^2 &= \frac{\lambda_{\bar{y}}}{\lambda_{\bar{y}} + \lambda_z} \\ h_8^2 &= 0 \end{aligned}$$

In summary we get

$$\begin{gathered} h_2^2 = 1,\ h_4^2 = h_5^2 = h_6^2 = h_8^2 = 0 \\ \left(h_3^2 - h_1^2\right)\lambda_y + \left(1 - h_1^2\right)\lambda_x = 0 \end{gathered}$$

$$h_3^2\lambda_z + \left(h_3^2 - h_7^2\right)\lambda_x = 0$$
$$h_7^2\lambda_z + \left(h_7^2 - 1\right)\lambda_{\bar{y}} = 0$$

So we have in essence three (nonlinear) equations and six unknowns. It is clear however that since $\lambda_{\bar{x}}$ and $\lambda_{\bar{z}}$ play no role in the dynamics we will not be able to estimate these two parameters. So finally, the vector $\lambda = \left(\lambda_x, \lambda_y, \lambda_{\bar{y}}, \lambda_z\right)^T \in \mathbb{R}^4_{>0}$ is apparently the only set of parameters we can hope to make some inference about. But first we derive analogous equations for the second steady-state $s_6 = 101$. Therefore we set $A = \{s_6\}$ in and obtain:

$$h_2^6 = 0,\ h_4^6 = h_5^6 = h_6^6 = h_8^6 = 1$$
$$h_1^6\lambda_x + \left(h_1^6 - h_3^6\right)\lambda_y = 0$$
$$\left(1 - h_3^6\right)\lambda_z + \left(h_7^6 - h_3^6\right)\lambda_x = 0$$
$$\left(1 - h_7^6\right)\lambda_z + h_7^6\lambda_{\bar{y}} = 0$$

Because of $h_j^6 = 1 - h_j^2$ the two equation systems are equivalent and we end up with three equations for four unknowns:

$$h_2^2 = 1,\ h_4^2 = h_5^2 = h_6^2 = h_8^2 = 0$$
$$\left(h_3^2 - h_1^2\right)\lambda_y + h_1^6\lambda_x = 0$$
$$h_3^2\lambda_z + \left(h_3^2 - h_7^2\right)\lambda_x = 0$$
$$h_7^2\lambda_z - h_7^6\lambda_{\bar{y}} = 0$$

Given that all coefficients are non-zero (in particular $h_3^2 \neq h_1^2$ and $h_3^2 \neq h_7^2$) it is straightforward that this system of equations (under the constraint that $\lambda > 0$) has the one-dimensional solution space given by

$$\left\{\mu\left(1, \frac{h_1^6}{h_1^2 - h_3^2}, \frac{h_7^2}{h_7^6}\frac{h_7^2 - h_3^2}{h_3^2}, \frac{h_7^2 - h_3^2}{h_3^2}\right)^T : \mu > 0\right\}.$$

This means that if some timescale, i.e. one of the parameters $\lambda_x, \lambda_y, \lambda_{\bar{y}}, \lambda_z > 0$, is given a priori, the three other parameters can be inferred from the steady-state absorption frequencies of the system depending on only three starting states, namely states 1, 3 and 7.

In a computational (i.e. theoretical) setting the parameter $\lambda_x > 0$ could easily be estimated by the amount of time the system needs to switch from state 5 to state 8 (see the state transition graph above). Since being at state 5 means that the system will switch to state 8 with probability 1. This gives a direct estimate of λ_x via the inverse of the sample mean of the measured time delays for this transition. How realistic these kinds of considerations are in the

light of real biological data and the involved data sampling capabilities remains to be elucidated.

For statistical models from which one can easily simulate given a parameterization, one can evaluate estimation procedures with simulated data as follows [Bajikar et al. 2014]. Given the true (chosen and therefore known) parameter set $\theta = \{\theta_i\}_i$ one can first simulate $M \in \mathbb{N}$ data sets with the parameters $\theta = \{\theta_i\}_i$ and apply the estimation procedure to be tested in order to obtain M estimates $\hat{\theta}_i^{(j)}$ of parameter θ_i with $j = 1, ..., M$. The mean squared error

$$\mathrm{MSE}\left(\hat{\theta}_i\right) = \mathbb{E}\left[\left(\theta_i - \mathbb{E}\left[\hat{\theta}_i\right]\right)^2\right] = \left[\mathrm{bias}\left(\hat{\theta}_i\right)\right]^2 + \mathrm{Var}\left(\hat{\theta}_i\right)$$ [11]

of the estimator $\hat{\theta}_i$ of the parameter θ_i can then be estimated by estimating the bias of the estimator as $\left(\frac{1}{M}\sum_{j=1}^{M}\hat{\theta}_i^{(j)}\right) - \theta_i$ and the variance as $\frac{1}{M-1}\sum_{j=1}^{M}\left(\hat{\theta}_i^{(j)} - \theta_i\right)^2$.

Figure 4.2.1 (on the next page) essentially shows the development of the MSE of the above proposed estimators for a particular illustrative parameter choice.

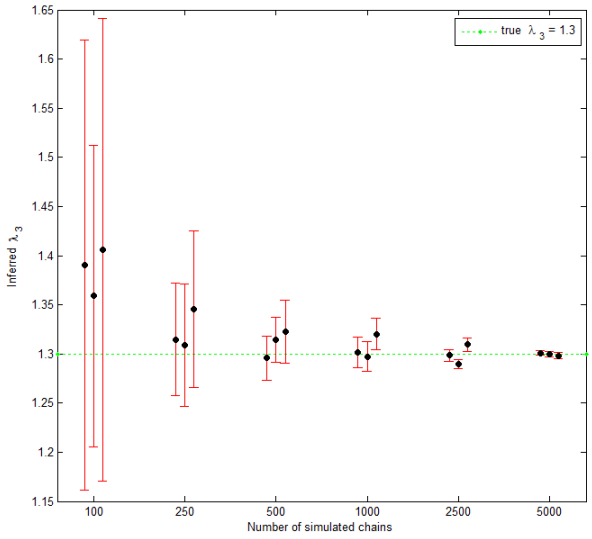
4.2.2 Circuits with more than three elements

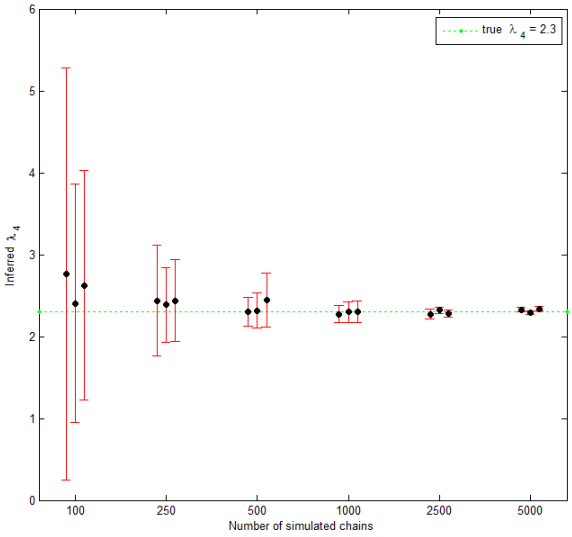
For a general system (with each species having states 1 or 0), define the set of steady states as $\mathcal{S} := \{q_i \in \mathcal{Q} : p_{ii} = 1\} \subset \mathcal{Q}$. For every state $q_j \in \mathcal{Q}$ we have $\sum_{q_i \in \mathcal{S}} h_j^i = 1$ and thus a priori we can hope to use at most[12] $\left(|\mathcal{S}| - 1\right)\left(|\mathcal{Q}| - |\mathcal{S}|\right)$ measured absorption frequencies in order to estimate the parameters as described in Subsection 4.2.1. Note that $\left(|\mathcal{S}| - 1\right)\left(|\mathcal{Q}| - |\mathcal{S}|\right)$ can be very large due to $|\mathcal{Q}| = 2^n$. But how many of the starting states could actually be biochemically implemented (for example by genetic engineering) and evaluated in terms of their absorption frequencies remains open at this point. The actual structure of the network would of course also affect which frequencies have to be measured. In the example from Subsection 4.2.1 for instance one had $h_5^6 = 1$ due to the structure of the network and hence this particular absorption frequency is not very useful for parameter estimation. This topic (which comes down to an identifiability analysis based on network structure) is also left aside here.

---

[11] [Czado, Schmidt 2011] with $\mathrm{bias}\left(\hat{\theta}_i\right) = \mathbb{E}\left[\hat{\theta}_i\right] - \theta_i$ the bias of the estimator.

[12] Note $h_i^i = 1$ for every $q_i \in \mathcal{S}$.





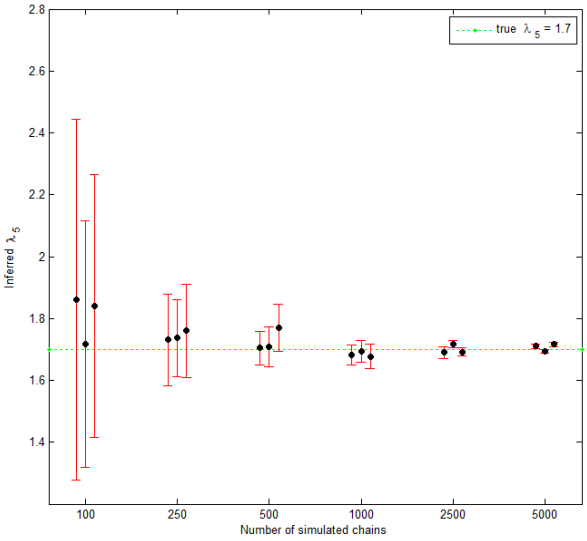


Fig. 4.2.1.

The estimation seems to work in an acceptable manner. But the the plots indicate that sample sizes of 1000 or greater are needed to really get reliable results.

One further remark is that all estimators seem to systematically over estimate the respective parameters.

The basic theory however proceeds as follows. Since there are 2n parameters in the system, one should have $\left(|\mathcal{S}|-1\right)\left(2^n-|\mathcal{S}|\right)\geq 2n$ in order to have chance to identify all parameters.[13] For practical purposes one should at least achieve $\left(|\mathcal{S}|-1\right)\left(F-|\mathcal{S}|\right)\geq 2n$ where $1\leq F\leq 2^n$ is number of experimentally implementable starting states. In terms of the method proposed in the preceding subsection in turns out that all "non-degenerate" starting states, i.e. all $q_k\in\mathcal{Q}$

[13] In the example above $\left(|\mathcal{S}|-1\right)\left(2^n-|\mathcal{S}|\right)=1\cdot(8-2)=6=2n$ but due to $h_4^6=h_5^6=h_8^6=1$ the structurally determined number of potentially useful data entities was reduced to just three absorption frequencies given by h_1^6, h_3^6 and h_7^6. Accordingly we were (luckily) able to at least estimate three parameters.

with $h_k^i \in (0,1)$ have to be measured in order to conduct the described procedure. This is simply due to the reliance of the procedure to solve systems of equations which for every $q_i \in \mathcal{S}^*$ with $\mathcal{S}^* \subset \mathcal{S}$, $|\mathcal{S}^*| = |\mathcal{S}| - 1$ are given by $h_i^i = 1$, $\forall j \neq i$: $h_j^i = \sum_{k=1}^{2^n} p_{jk} h_k^i$.

By remembering the notation from Section 4.l the estimation procedure can now be described as follows. For all steady states $q_i \in \mathcal{S}^*$ one makes the following reformulation of the respective equation system such that for every $j \neq i$ we first get:

$$h_j^i = \sum_{k=1}^{2^n} p_{jk} h_k^i = \sum_{q_k \in \mathcal{C}_{STG}(q_j)} \left(\frac{\lambda\left(\tau_{\phi_1^{(q_j)}(q_k)} \left(\phi_2^{(q_j)}(q_k), \phi_3^{(q_j)}(q_k) \right) \right) h_k^i}{\sum_{q_k \in \mathcal{C}_{STG}(q_j)} \lambda\left(\tau_{\phi_1^{(q_j)}(q_k)} \left(\phi_2^{(q_j)}(q_k), \phi_3^{(q_j)}(q_k) \right) \right)} \right)$$

$$= \frac{\sum_{q_k \in \mathcal{C}_{STG}(q_j)} \lambda\left(\tau_{\phi_1^{(q_j)}(q_k)} \left(\phi_2^{(q_j)}(q_k), \phi_3^{(q_j)}(q_k) \right) \right) h_k^i}{\sum_{q_k \in \mathcal{C}_{STG}(q_j)} \lambda\left(\tau_{\phi_1^{(q_j)}(q_k)} \left(\phi_2^{(q_j)}(q_k), \phi_3^{(q_j)}(q_k) \right) \right)} .$$

This leads to

$$\left[\sum_{q_k \in \mathcal{C}_{STG}(q_j)} \lambda\left(\tau_{\phi_1^{(q_j)}(q_k)} \left(\phi_2^{(q_j)}(q_k), \phi_3^{(q_j)}(q_k) \right) \right) \right] h_j^i = \sum_{q_k \in \mathcal{C}_{STG}(q_j)} \lambda\left(\tau_{\phi_1^{(q_j)}(q_k)} \left(\phi_2^{(q_j)}(q_k), \phi_3^{(q_j)}(q_k) \right) \right) h_k^i$$

and thus finally to

$$\sum_{q_k \in \mathcal{C}_{STG}(q_j)} \left(h_k^i - h_j^i \right) \lambda\left(\tau_{\phi_1^{(q_j)}(q_k)} \left(\phi_2^{(q_j)}(q_k), \phi_3^{(q_j)}(q_k) \right) \right) = 0 .$$

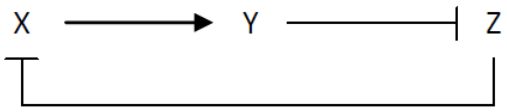
The method now further proceeds by solving the resulting system of $\leq (|\mathcal{S}| - 1)(2^n - |\mathcal{S}|)$ linear equations with unknowns $\lambda_i^{(1)}$, $\bar{\lambda}_i^{(1)} > 0$ for all $i = 1, ..., n$. As we have seen in the preceding subsection the exact number of (non-trivial) equations depends on the number of "non-degenerate" starting states and it may be merely possible to solve the system after certain parameter values are a priori given. In addition one might run into problems if $h_k^i \approx h_j^i$ for some k and j.

A theory which clarifies these issues based on the structure of the respective networks would be nice. For now I close this subsection by remarking that the given derivation of the final equation system is just as well valid for general attractors instead of just fixed points and does also not rely on the restriction to 0-1-valued states. The (anyhow incomplete) considerations about the minimal frequencies needed such that one has a chance to estimate parameters would of course differ but principally proceed along the same lines.

The next subsection looks at a second example. The example again considers only three species but we have seen that the method generalizes (in theory).

4.2.3 Another three element circuit

We have a look at the following three element molecular circuit from (studied for example by [Thomas, 1990]):



Again, I adopt the simplified notation already used in Subsection 4.2.1: The three species are denoted by X, Y and Z and their states by x,y and z respectively. Although it introduces moderate ambiguity, the images of the state variables x,y and z are also denoted by X,Y and Z (i.e. $X = f_1(x, y, z)$, etc.).

The above circuit is formally described by the following image equations:

$$X = \bar{z},\ Y = x,\ Z = \bar{y}.$$

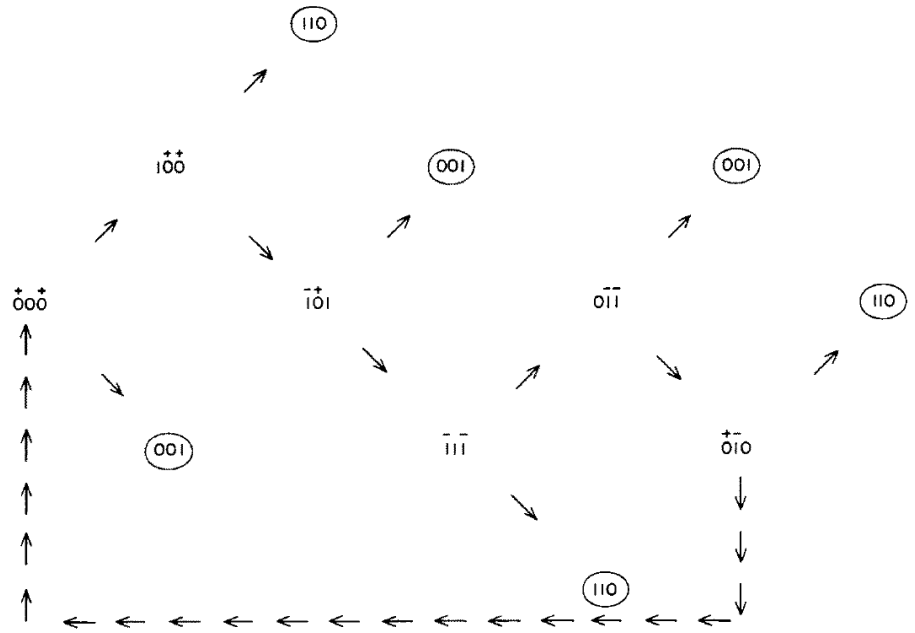
As in Subsection 4.2.1, we obtain the image table:

xyz	XYZ
$\overset{+}{0}0\overset{+}{0}$	101
001	**001**
$\overset{+}{0}1\overset{-}{0}$	100
$0\overset{-}{1}\overset{-}{1}$	000
$\overset{+}{1}0\overset{+}{0}$	111
$\overset{-}{1}0\overset{+}{1}$	011
110	**110**
$\overset{-}{1}1\overset{-}{1}$	010

Again (as in example 4.2.1) there exist two steady states (i.e. states that correspond to their image) and state variables which are about to change in a particular state are marked by a plus or minus sign depending on whether up- or down-regulation is up to occur. The steady states are states four and seven, where we make again the following convention regarding the

indices of the states (the index of a state minus one is given by the respective binary representation): $q_1 = 000,\ q_2 = 100,\ q_3 = 010,\ q_4 = 001,\ q_5 = 011,\ q_6 = 101,\ q_7 = 110,\ q_8 = 111$.

One obtains the following state transition graph (from [Thomas 1990]):



As before we can derive the transition matrix $P = \left(p_{ij}\right)_{1 \le i,j \le 8}$:

$$P = \begin{pmatrix}
0 & \frac{\lambda_x}{\lambda_x + \lambda_z} & 0 & \frac{\lambda_z}{\lambda_x + \lambda_z} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{\lambda_z}{\lambda_y + \lambda_z} & \frac{\lambda_y}{\lambda_y + \lambda_z} & 0 \\
\frac{\lambda_{\bar{y}}}{\lambda_x + \lambda_{\bar{y}}} & 0 & 0 & 0 & 0 & 0 & \frac{\lambda_{\bar{y}}}{\lambda_x + \lambda_{\bar{y}}} & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{\lambda_{\bar{z}}}{\lambda_{\bar{y}} + \lambda_{\bar{z}}} & \frac{\lambda_{\bar{y}}}{\lambda_{\bar{y}} + \lambda_{\bar{z}}} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{\lambda_{\bar{x}}}{\lambda_{\bar{x}} + \lambda_y} & 0 & 0 & 0 & \frac{\lambda_y}{\lambda_{\bar{x}} + \lambda_y} \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & \frac{\lambda_{\bar{x}}}{\lambda_{\bar{x}} + \lambda_{\bar{z}}} & 0 & \frac{\lambda_{\bar{z}}}{\lambda_{\bar{x}} + \lambda_{\bar{z}}} & 0
\end{pmatrix}$$

Given P we can now derive, again analogously to Subsection 4.2.1, the equations for the absorption probabilities $h_i^4 = P\left(\bigcup_{\omega \in \mathbb{N}_0} \{\Xi_\omega = q_4\} | \Xi_0 = q_i \right)$, $i = 1,...,8$, leading to

$$h_4^4 = 1,\ h_7^4 = 0,\ h_i^4 = \sum_{j=1}^{8} p_{ji} h_j^4,\ i \neq 4,7 . \qquad (ß2ß)$$

Again the corresponding equations for the h_i^7's would not give something new because of $h_i^4 = 1 - h_i^7$ for $i = 1,...,8$.

Reformulating (by intuition or according to the principles outlined in the preceding subsection) one arrives at $H\lambda = 0$ with

$$\lambda = \left(\lambda_x, \lambda_{\bar{x}}, \lambda_y, \lambda_{\bar{y}}, \lambda_z, \lambda_{\bar{z}}\right)^T \in \mathbb{R}^6$$

and

$$H = \begin{pmatrix} h_2^4 - h_1^4 & 0 & 0 & 0 & h_1^7 & 0 \\ 0 & 0 & h_2^4 & 0 & h_2^4 - h_6^4 & 0 \\ h_3^4 & 0 & 0 & h_3^4 - h_1^4 & 0 & 0 \\ 0 & 0 & 0 & h_5^7 & 0 & h_3^4 - h_5^4 \\ 0 & h_6^7 & h_8^4 - h_6^4 & 0 & 0 & 0 \\ 0 & h_8^4 - h_5^4 & 0 & 0 & 0 & h_8^4 \end{pmatrix}.$$

Given that all "non-zero"-entries in H are non-zero, the system $H\lambda = 0$ has the following set of positive[14] solutions:

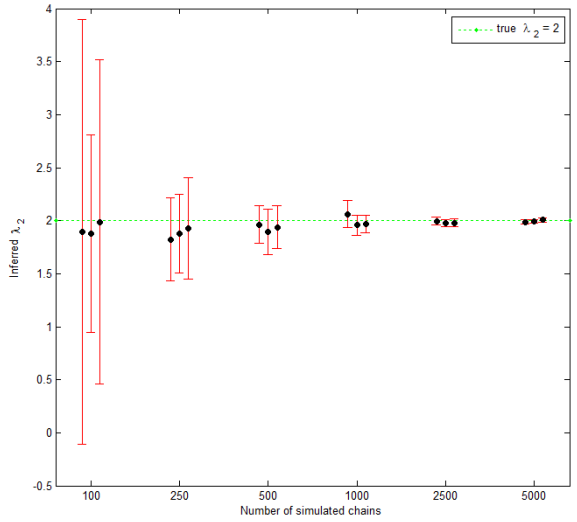
$$\left\langle \left(1,\ \frac{\left(h_6^4 - h_8^4\right)\left(h_6^4 - h_2^4\right)\left(h_1^4 - h_2^4\right)}{h_6^7 h_2^4 h_1^7},\ \frac{\left(h_6^4 - h_2^4\right)\left(h_1^4 - h_2^4\right)}{h_1^7 h_2^4},\ \frac{h_3^4}{h_1^4 - h_3^4},\ \frac{h_1^4 - h_2^4}{h_1^7},\ \frac{h_3^4 h_5^7}{\left(h_1^4 - h_3^4\right)\left(h_5^4 - h_3^4\right)} \right)^T \right\rangle_{pos}$$

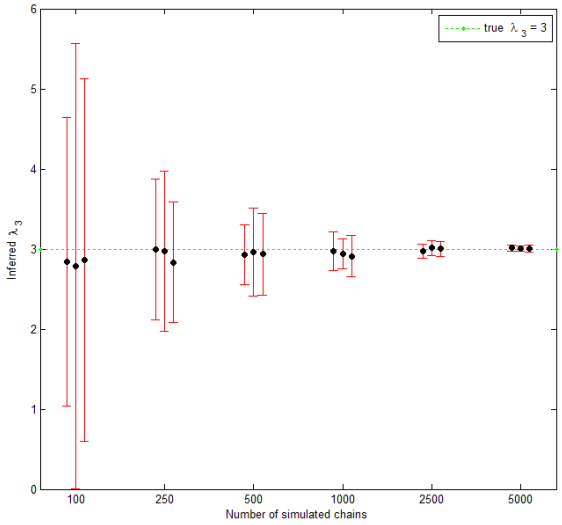
with $\langle x \rangle_{pos} := \{\mu x : \mu > 0\}$ for $x \in \mathbb{R}^n$.

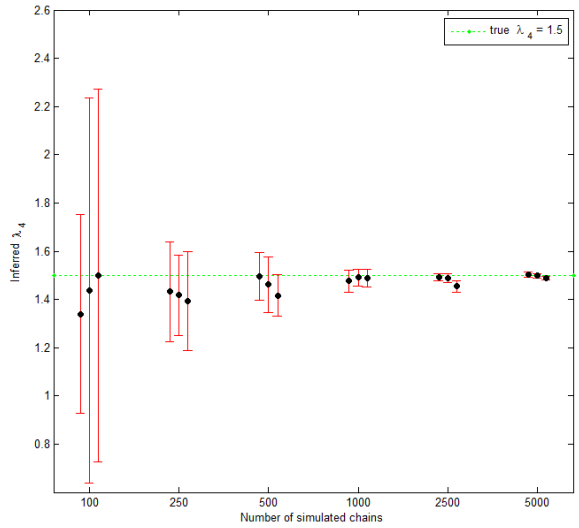
The M-files ExampleTwo.m, EstimExampleTwoSub.m and EstimExampleTwo.m are documented in Appendix A.6, contained on the CD and can be used to estimate the parameters in the setting where λ_x is assumed to be known.

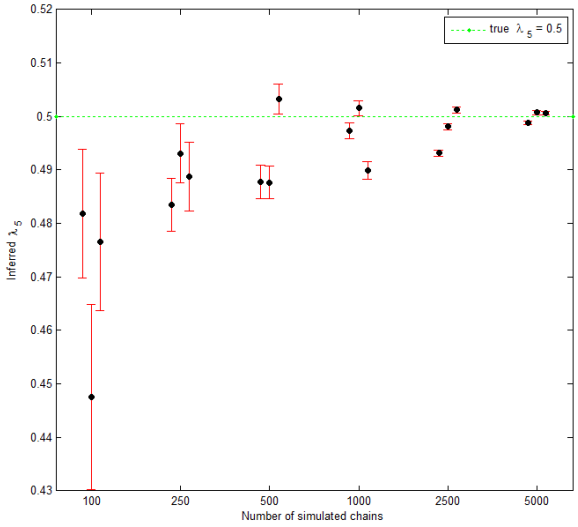
Rudimentarily evaluating the proposed estimation scheme as in Subsection 4.2.1 gives the results shown in figure 4.2.2 (on the next page).

[14] The parameters $\lambda_\cdot$ should be positive since they parametrize exponential distributions.









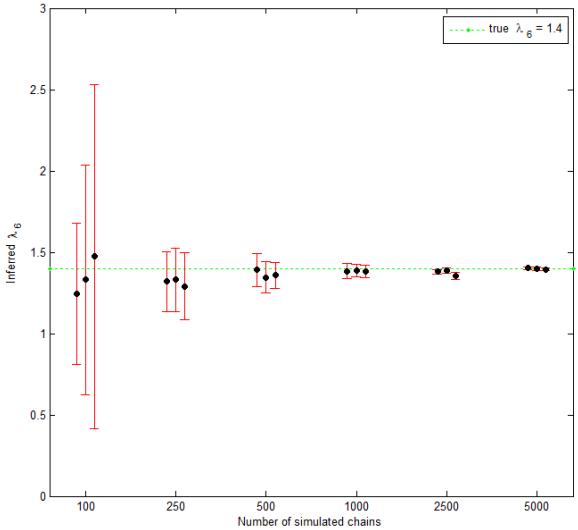


Fig. 4.2.2. In essence the estimators show the same behavior as in Subsection 4.2.1 and figure 4.2.1.

The only thing one could remark is that now instead of overestimating, the estimators seemingly systematically underestimates all (!) the parameters.

4.2.4 Mixture models for absorption frequency based parameter estimation

The approach discussed in the preceding subsections only works if the delays are exponentially distributed. For Weibull delays the resulting process is not a Markov chain and hence the simple theory outlined above does not apply. In addition, the assumption that one can measure the absorption frequencies as a function of every possible "non-degenerate" start state seems, to say it plainly, absurd. This is due to the fact that the number of start states grows exponentially with the number of species in the system and hence the experimental work involved might seem a little bit too overwhelming, set aside the question if it is altogether possible to experimentally implement every needed start state for a particular biochemical system.

In real biological applications it seems more probable that the initial state of the system under study is from a well-defined tentatively small set of biologically plausible start states which are known to be possible. The difficulty then might be that one does not know exactly in which state the system starts, i.e. one might observe an absorption frequency but cannot say to which starting state this frequency has to be associated. This kind of situation seems to be a natural playground for so called *mixture models*.

In the following "(!?)" indicates a crucial assumption which deserved further consideration. Let $\mathcal{F} \subset \mathcal{Q}$ be the set of "plausible" start states. Here, "plausible" (!?) could be defined by a priori expert knowledge or just exclude the states which unambiguously kill the organism and thus cannot be existent in an experimental world. For the set of steady states $\mathcal{S} := \{q_i \in \mathcal{Q} : p_{ii} = 1\}$ we can (!?) now count the respective cells absorbed to the steady states as $N_{q_i} \in \mathbb{N}_0$. Assume w.l.og. $\mathcal{S} = \{q_1, ..., q_s\}$ with $s := |\mathcal{S}|$. Then, with $N_i := N_{q_i} \in \mathbb{N}_0$ and by assuming that cells "decide" independently (!?) from each other to which steady state they evolve, we can identify the likelihood $\mathcal{L}(\theta | N_1, ..., N_s)$ for the parameters θ (the λ's in the $\mathbb{M}_{Exp}$ case, the λ's and k's in the $\mathbb{M}_{WB}$ case) given absorption counts $N_1, ..., N_s$ as the following *mixture of multinoulli distributions*:

$$\mathcal{L}(\theta | N_1, ..., N_s) = \sum_{q_j \in \mathcal{F}} \pi_j \binom{N}{N_1 \cdots N_s} \prod_{i=1}^{s} \left(h_j^i(\theta)\right)^{N_i} .$$

Here, $\pi_j \in [0,1]$ is the probability that a cell starts out with initial state $q_j \in \mathcal{F}$, $h_j^i(\theta)$ is the parameter-dependent absorption probability and N is the number of all cells examined for their fate. The task to estimate such a mixture model would now be to estimate the π_j's and θ based on the absorption counts $N_1, ..., N_s$. In the above formulation it is assumed that all cells in one, say, tube are in the same start state (!?). Accordingly, the entire data would involve different tubes of cells leading to several counts given by $N_1^{(z)}, ..., N_s^{(z)}$ for $z = 1, ..., L \in \mathbb{N}$, say.

For inference in mixture models, the so called *EM* (*expectation maximization*) *algorithm* or one of its extensions like the *Monte Carlo EM* (MCEM) algorithm can be a valuable tool [Robert, Casella 2000: Section 5.3], [Bishop 2006: Chapter 9], [Murphy 2012: Chapter 11]. This is due to the fact that the EM algorithm is especially suited to certain types of so called *latent variable models* which are model where only part of the system determining variables are observed. In the above case the unobserved "data" would be the start state of the cell in a tube of cells. Notice that mixtures of multinoullis are so to speak one of the standard examples of mixture models which can be estimated with the EM algorithm [Bishop 2006: Exercise 9.19]. But in the canonical case there is a slight difference to the above model. While above the parameters θ do not depend on the respective mixture component (i.e. the start state), the standard multinoulli mixture model deals with component-dependent parameters θ_j.

I finally remark that it is relatively straightforward to adapt the EM algorithm to the need of a Bayesian analysis to perform *MAP estimation* [Murphy 2012: Section 11.4]. This would involve the issue of finding a suitable prior. Since the model is highly complex there is certainly no obvious choice for a prior for θ (in the sense of conjugacy, for example). The prior for the mixing probabilities can canonically be chosen to be Dirichlet-distributed. Further approaches for Bayesian inference in finite mixture models also include *mixture posterior simulation* for example [Robert, Casella 2000: Section 9.3].

The applicability of the above mentioned approaches to the setting defined above is not detailed in this thesis.

4.3 Further considerations

The data of the preceding section was reduced to absorption frequencies. One could also consider the estimation task in the light of time series data. Since GKL networks are time-continuous and state-discrete there are the possibilities to base the parameter estimation on the time-continuous process (which would correspond to the most information) or on the jump process, or on a combination thereof. Based on completely observed jump processes for example one could naively consider to simulate the likelihood (which is possible due to the fact that one can actually simulate both model types) over a suitable grid of points in parameter space and then so to speak construct a smooth likelihood surface for example by means of multivariate smoothing splines [Reinsch 1967]. Unfortunately however these kinds of questions are not pursued further here.

Instead, the differences between the models $\mathbb{M}_{\mathrm{WB}}$ and $\mathbb{M}_{\mathrm{Exp}}$ are illustrated by means of some simple simulations. First, consider the network from Subsection 4.2.1. Assume that we start from state 000 and reach state 010. There one notices that the delay for species one is active since the last state and hence in the $\mathbb{M}_{\mathrm{WB}}$ this should lead to a faster switching as in the case where the delay is only activated in the present state. If species one switches this inevitably

leads to the steady state 101 what can easily be seen by having a look at the state transition graph from Subsection 4.2.1. One can ask the question what happens, given scale parameters λ equal for every species and switching direction, to the absorption frequency to steady state 101 or equivalently to 100 (with respect to start state 000) for increasing shape parameters k. If species one, two and three have the same k's for up-switching this should intuitively result in a decreasing frequency of 100-absorption since then at state 000, the active species switch equiprobable as well as in state 010 if the system was without its history. But by taking the history into account, if the second state happens to be 010, the shape parameter responsible for up-switching of species one will drive the system more and more towards 101 if it increases. Note that one has to vary the up-switch k's for species 2 and 3 accordingly in order to balance the switch probabilities to equate to be equiprobable relative to the system evolving without history. The intuition is partially confirmed in the simulations shown in figure 4.3.1.

In general, it is surprisingly non-straightforward to draw intuitive conclusions about the expected behavior of the system given a particular parameter constellation.

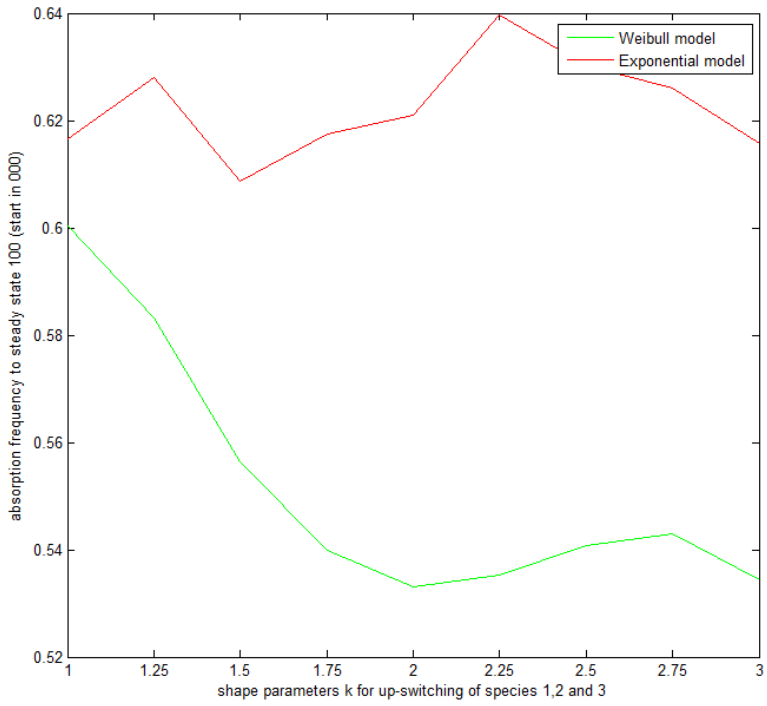


Fig. 4.3.1:

The figure shows simulated absorption frequencies according to the intuitive logic described in the main text.

All scale parameters were set to one, for the Weibull model the shape parameters for up-switching of all three species were increased (simultaneously) from 1 to 3 in steps of 0.25. All other shape parameters were kept to one.

In the exponential model all parameters were fixed to one.

5 PLSDEs and general stochastic hybrid systems (GSHSs)

In this final section I want to give a short impression on stochastic hybrid models and how they possibly relate to the probabilistic timed GKL networks studied in the preceding section. In Section 5.1 I begin by briefly looking at an approach which was proposed to associate transition probabilities to PLDEs in the sense that one asks the question (in some meaningful sense to be defined) what the probability is for a discrete logical state of the PLDE to follow upon another. Section 5.2 then deals with a simple stochastic version of PLDEs termed piecewise linear stochastic differential equations (PLSDEs). They are analogously defined as PLDEs but in contrast the continuous dynamics on each domain is governed by stochastic differential equations (SDEs) instead of ODEs. This kind of model is of course not new and actually an instance of a so called general stochastic hybrid system (GSHS) [Bojuriano, Lygeros 2008] which are shortly outlined in Section 5.3. The overall section is highly superficial, merely presents some examples and gives rough outlines of the broader context involved.

5.1 Probabilistic interpretation of PLDEs

[Chaves, Gouzé 2010] and [Chaves et al. 2013] proposed a method to associate transition probabilities between the discrete states (corresponding to the different regulatory domains) of a PLDE system simply by dividing the domains (polytopes defined by the threshold hyperplanes, see Subsection 3.6) into regions according to the destination of the trajectories which start in the respective regions. Since the dynamics of a PLDE system in every domain is uniquely determined by an initial this can always be done. In a first approach [Chaves et al. 2013] defined the one-step transition probabilities between region A and a neighboring region B as the area of the region of regulatory domain A which contains all the initial values in A which give rise to trajectories which evolve to regulatory domain B. This Markov-type transition system is only a crude approximation since the actually happening transitions in a PLDE system are highly dependent also on domains that were passed in the past before the trajectory finally led to A. So a second approach therefore considered and defined two-step transition probabilities which also take into account the possible regulatory domains prior to regulatory domain A.

The ultimate goal of this kind of analysis (which introduces probabilities to a completely deterministic system) is to relate the parameters of the PLDE system to more abstract notions which only involve the transition between logical states (i.e. domains) according to specific transition probabilities. It seems strange at first to associate probabilities to a deterministic system is such a way but one could intuitively think about this procedure in terms of noise which is introduced whenever the domain switches. Normally, a trajectory of a PLDE system is continuous and just continuously evolves into the next domain (I ignore the possibility of

sliding mode solutions) where it again deterministically evolves according to the new vector field of this new regulatory domain. The probabilistic interpretation of [Chaves et al. 2013] might correspond to the situation where the system evolves just as before in every regulatory domain but behaves randomly in terms of the domain switches. Every time a trajectory meets a switching boundary, the position in the regulatory domain which will normally be continuously entered could now thought to be chosen uniformly from the entire regulatory domain. This would correspond to the area definition of the one-step transition probabilities. The two-step transition probabilities could be interpreted the same way by randomly choosing an initial value in the new regulatory domain, but now not completely independent but dependent in order to reflect the history of the system in some meaningful way. Of course, these models are simply different models than PLDE models but it is nevertheless interesting under which conditions different but related models can inform each other.

If the connection between two such related systems could be made sufficiently strict and consistent the possibility arises to use knowledge inferred with either model to inform the other, for example in term of parameter estimation (also briefly indicated in [Chaves et al. 2013]). This is basically the same logic which is for example applied when inferring the steady states of a continuous model from a simpler Boolean model as described in Chapter 3. However as detailed in [Jamshidi 2012] these matters are already very intricate and non-trivial in a non-stochastic interpretation and it seems probable that the approach of [Chaves et al. 2013] is amenable to some kinds of refinements. For example one could consider stochastic versions of the qualitative simulation approach of [De Jong et al. 2004a].

One particular aspect concerns the fact that the 'probabilistic' interpretation in [Chaves et al. 2013] relates to a completely deterministic system. The next subsection extends the deterministic PLDE framework in a straightforward manner to an inherently stochastic model which ideally could be one starting point for more explorations on the topic how discrete stochastic models (like the probabilistic timed GKL networks from the preceding section) relate to continuous models.

5.2 Piecewise linear stochastic differential equations (PLSDEs)

It appears straightforward to endow the deterministic differential equation evolution of a PLDE system on its respective domain with some kind of stochasticity. The resulting model would then be a piecewise linear stochastic differential equation [Reed, Zwart 2011], [Simpson, Kuske 2012].

Assume that a system with species $\mathcal{S}=\{\mathcal{S}_1,...,\mathcal{S}_n\}$, $n\in\mathbb{N}$ is given. The stochastic chemical kinetics corresponding to a piecewise linear system for a single domain is given by the following 2n reactions:

$$\mathcal{R}_j:\mathcal{S}_j\to\varnothing,\ \mathcal{R}_{j+1}:\varnothing\to\mathcal{S}_j \text{ for } j=1,...,n.$$

The corresponding propensities are $a_j(X) = \gamma_j X_j$ and $a_{j+1}(X) = \beta_j \in [0, \infty)$ for $j = 1, ..., n$ where $X = (X_1, ..., X_n)^T \in \mathbb{N}_0^n$ is supposed to be the state of the system describing the molecule numbers.

Following Subsection 2.2.5, one obtains $A(X) = Sa(X)$ with

$$S = \begin{pmatrix} -1 & 1 & \cdots & 0 & 0 \\ 0 & 0 & \ddots & \vdots & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & \cdots & -1 & 1 \end{pmatrix} \in \mathbb{Z}^{n \times 2n} \text{ and } a(X) = (a_1(X), ..., a_{2n}(X)) \in \mathbb{R}^{2n}$$

and $D(X) = S \cdot \mathrm{diag}^{1/2}[a(X)]$ which leads to the corresponding CLE as

$$\begin{aligned} dX_t &= A(X_t)dt + \left[D(X_t)D(X_t)^T\right]^{1/2} dB_t \\ &= A(X_t)dt + \left(S\,\mathrm{diag}[a(X_t)]S^T\right)^{1/2} dB_t \end{aligned} .$$

Here, $(B_t)_{t \geq t}$ is n-dimensional standard Brownian motion [Oksendal 2010: Section 2.2] and X_t is the continuous-valued stochastic process which is describes the concentrations of the species and which is given as the solution of the above stochastic differential equation (SDE). With $X_t = (X_t^{(1)}, ..., X_t^{(n)})^T$ this can be written component-wise by

$$\begin{aligned} dX_t^{(i)} &= A_i(X_t)dt + \sum_{k=1}^{n} \left[\left(S\,\mathrm{diag}[a(X_t)]S^T\right)^{1/2}\right]_{ik} dB_t^{(k)} \\ &= \left(\beta_i - \gamma_i X_t^{(i)}\right)dt + \left(\beta_i + \gamma_i X_t^{(i)}\right)^{1/2} dB_t^{(i)} \end{aligned} .$$

Now, by making the transition to a multi-domain piecewise linear system we can define a piecewise linear stochastic differential equation system (PLSDE system) with Langevin noise (Langevin PLSDE system) as follows.

A Langevin PLSDE system is given by the following SDEs [Oksendal 2010]:

$$\begin{aligned} dX_t &= \left(\beta^D - \Gamma^D X_t\right)dt + \sigma_{\beta,\Gamma}^D(X_t)dB_t \text{ for } X_t \in D,\ t > 0. \\ X_0 &\sim \mathbb{P}_0 \end{aligned}$$

Here, $X_t = (X_t^{(1)}, ..., X_t^{(n)})^T$ is the n-dimensional stochastic process describing the species concentrations, $\beta^D = (\beta_1^D, ..., \beta_n^D)^T \in \mathbb{R}_{\geq 0}^n$ the vector of production rates corresponding to the domain D, $\Gamma^D = \mathrm{diag}(\gamma_1^D, ..., \gamma_n^D) \in \mathbb{R}_{\geq 0}^{n \times n}$ the diagonal degradation rate matrix (again given

domain D) and the diffusion is given by $\sigma_{\beta,\Gamma}^{D}\left(X_t\right)=\operatorname{diag}\left(\sqrt{\beta_1^D+\gamma_1^D X_t^{(1)}},...,\sqrt{\beta_n^D+\gamma_n^D X_t^{(n)}}\right)$ which finally of course also depends on the domain D. Finally, $\left(B_t\right)_{t\geq 0}$ is a n-dimensional standard Brownian motion process and $\mathbb{P}_0$ is the distribution of the initial condition variable X_0 which has range $\mathbb{R}_{\geq 0}^n$ (and which will almost always be assumed to be a constant in the sequel).

It is $D\in\mathcal{D}$ with the so called (regulatory) domain set $\mathcal{D}:=\left\{D_g : g\in\mathcal{T}\right\}$, $\mathcal{T}:=\prod_{i=1}^{n}\{0,1,...,p_i\}$ while $p_i\in\mathbb{N}$ represents the number of thresholds associated to species i. (g for gene and $\mathcal{T}$ for transcription.) Further define the (regulatory) domains $D_g:=\left\{x\in\mathbb{R}_{\geq}^n : {}^g\theta < x < {}^{g+1}\theta\right\}$ with ${}^g\theta:=\left(\theta_{g_1}^{(1)},...,\theta_{g_n}^{(n)}\right)\in\left(\mathbb{R}_{\geq 0}\cup\{\infty\}\right)^n$, $g+1:=\left(g_1+1,...,g_n+1\right)$ for $g\in\mathcal{T}$ and the $<$-relation understood component-wise. Notice that in terms of the GKL interpretation (see Section 4.1.1) of the thresholds an index can be identified as the corresponding state vector in the discrete GKL model. This will play a role later on. For now one can further state that for all $\ell\in\{1,...,p_i\}$, $\theta_\ell^{(i)}\in\mathbb{R}_+$ represents the ℓ-th threshold of species i while by definition $\theta_0^{(i)}:=0$ and $\theta_{p_i+1}^{(i)}:=\infty$ for all $i=1,...,n$. The thresholds are assumed to be ordered according to $\theta_1^{(i)}<\theta_2^{(i)}<\cdots<\theta_{p_i}^{(i)}$ for every species $i=1,...,n$. The set of threshold of species i is denoted by Θ_i, i.e. one has $\Theta_i=\left\{\theta_1^{(i)},...,\theta_{p_i}^{(i)}\right\}$.

The only difference in the above definitions compared to the usual PLDE systems defined in Section 3.6 is the diffusion term which makes the system a system of piecewise linear SDEs instead of ODEs.

Notice that there is an alternative perspective on the domain-dependent production constants β^D and degradation rates Γ^D. Exactly as in Section 3.6 on PLDEs these numbers can be interpreted as the output of some logic function. Given that the continuous state X of the process (t is suppressed since it does not play a role in the argument) is part the regulatory domain D_g with some $g\in\mathcal{T}=\prod_{i=1}^{n}\{0,1,...,p_i\}$ we can suggestively write $\beta^{D_g}=\beta(g)=\beta^g$ and $\Gamma^{D_g}=\Gamma(g)=\Gamma^g$. What this basically means is that production and degradation terms in a given domain are functions of the logical state (image functions) (as defined in Section 3.5 in the context of GKL networks) associated to that particular domain. This is of course again in analogy to the original PLDE idea [Glass, Kauffman 1973].

Notice further that given a (regulatory) domain $D\in\mathcal{D}$ (according to appendix A.3) the SDE

$$\begin{aligned} dX_t &= \left(\beta^D-\Gamma^D X_t\right)dt+\sigma_{\beta,\Gamma}^{D}\left(X_t\right)dB_t,\ t>0\\ X_0 &\sim \mathbb{P}_0 \end{aligned}$$

is solved by a stochastic process $(X_t)_{t\geq 0}$ if it holds that

$$X_t = X_0 + \int_0^t \left(\beta^D - \Gamma^D X_s\right) ds + \int_0^t \sigma^D_{\beta,\Gamma}(X_s)\, dB_s$$

for all $t > 0$ where the second integral is defined as an Itô integral [Oksendal 2010: Chapter 3]

A solution of a PLSDE system as a whole is not so straightforward to define since the right-hand side is discontinuous. PLSDE system are (in analogy to PLDE systems) not defined on the threshold hyperplanes with the concept of threshold hyperplanes defined the same ways as for PLDEs being the hyperplanes of the state space where one species is located on one of its respective thresholds. Furthermore, in accordance with [De Jong et al. 2004] the domains D_g are also called regulatory domains while the so called switching domains on the other hand are again defined as the subsets of the state space where regulatory domains 'meet'. See Section 3.6 for comparison.

In the following we develop a naïve approach for the simulation of PLSDEs where the threshold hyperplanes do not play a role. Of course, "do not play a role" simply means that they are not adequately incorporated into the simulation procedure, it does by no means imply that the simulated dynamics is not influenced by the discontinuities at the threshold hyperplanes. For deterministic PLDEs there are numerical methods which take into account the Fillipov extensions [Fillipov 1988] at the switching domains, see for example [Stewart 1990].

Naïve PLSDE simulation

Given a Langevin PLSDE system one can naively simulate it as follows. Starting from a particular domain $D \in \mathcal{D}$ one simulates the process on every domain one may encounter in the sequel via the Euler-Maruyama scheme [Fuchs 2013: Subsection 3.3.2] but additionally checks the following two conditions before one accepts the new simulated state of an iteration step.

First it is checked which of the variables are negative and the variables for which it turns out to be so are set to zero since negative concentration are not physical but can arise due to discretization. This procedure is approximately valid in order to keep the state variables non-negative if the discretization step is not too large. More advanced procedures would certainly involve the adaptive choice of variable specific time steps when these variables become smaller and approach zero.

The second condition which has to be checked is a condition which decides if a switch of the domain happens or not. That is clearly important since one has to simulate the SDE corresponding to the domain the state process is situated in. Remember that if any variable passes one of its thresholds one has a corresponding change of the domain. So given a newly simulated state of the process one therefore has to check if one of the variables passed one of

its thresholds and if so use the SDE corresponding to the well-defined new domain for simulation in the next iteration instead of the SDE of the then left behind domain. If two or more variables happen to cross a threshold in a single simulation step one proceeds accordingly. If a variable happens to be located on one of its thresholds the simulation step is discarded and a new one is generated. This is necessary since the PLSDE system is not defined at points where at least one of the variables is equal to one of its thresholds.

It turns out that it is convenient to introduce an additional parameter into the model in order to be able to control the strength of the noise.

Formalizing the above description we arrive at the following algorithm.

ALGORITHM 5.2 [Naïve simulation of Langevin PLSDE systems]

Input: I1. $n \in \mathbb{N}$ # number of species

I2. $p_i \in \mathbb{N} \quad \forall i \in [n]$ # number of thresholds for each species

I3. $\Theta_i = \left\{\theta_1^{(i)},...,\theta_{p_i}^{(i)}\right\} \subset \mathbb{R}_+$ with $\theta_1^{(i)} < \theta_2^{(i)} < \cdots < \theta_{p_i}^{(i)} \quad \forall i \in [n]$

ordered thresholds for every species

I4. $\beta^{(g)} = \left(\beta_1^{(g)},...,\beta_n^{(g)}\right)^T \in \mathbb{R}_{\geq 0}^n \qquad \forall g \in \mathcal{T} = \prod_{i=1}^n \{0,1,...,p_i\}$

production constants for every domain (indexed by logical state)

I5. $\Gamma^{(g)} = \mathrm{diag}\left(\gamma_1^{(g)},...,\gamma_n^{(g)}\right) \in \mathbb{R}_{\geq 0}^{n\times n} \qquad \forall g \in \mathcal{T} = \prod_{i=1}^n \{0,1,...,p_i\}$

degradation rates for every domain (indexed by logical state)

I6. $\Delta > 0$ # step size

I7. $t_{max} > 0$ # maximal simulation time

I8. $\mathbb{P}_0$, a probability measure on $\mathbb{R}_{\geq 0}^n$ # initial value distribution

I9. $\pi \geq 0$ # noise strength

Output: O1. $Y = (Y_0, Y_1,..., Y_m) \in \mathbb{R}_{\geq 0}^{n\times(m+1)}$ with $Y_\tau = \left(Y_\tau^{(1)},..., Y_\tau^{(n)}\right)^T \in \mathbb{R}_{\geq 0}^n,\ \tau = 0,...,m.$

$(m = \rho\Delta$ with $\rho = \lfloor t_{max}/\Delta \rfloor)$

simulated time course of the n-dimensional process

---

(3) simulate Y_0 from $\mathbb{P}_0$

(4) $Y \leftarrow \mathrm{matrix}\left(n, \rho = \lfloor t_{max}/\Delta \rfloor\right)$, $Y[:,1] \leftarrow Y_0$

(5) $g \leftarrow \sum_{\kappa\in\mathcal{T}} \kappa \mathbb{I}_{D_\kappa}(Y_0)$ # initial domain via its gene logic state $g = (g_1,...,g_n)^T$

(6) $\tau \leftarrow 0$

(7) while $(\tau+1)\Delta < t_{max}$:

(7.1) simulate $\Delta B_\tau \sim \mathcal{N}(0, I_n)$

(7.2) # Euler-Maruyama iteration:

$$Y_{\tau+1} \leftarrow Y_\tau + \left(\beta^{(g)} - \Gamma^{(g)} Y_\tau\right)\Delta + \mathrm{diag}\left(\sqrt{\beta_1^{(g)} + \gamma_1^{(g)} Y_\tau^{(1)}}, ..., \sqrt{\beta_n^{(g)} + \gamma_n^{(g)} Y_\tau^{(n)}}\right)\sqrt{\pi}\sqrt{\Delta}\ \Delta B_\tau$$

(7.3) $\mathrm{nonneg} \leftarrow \left(\mathbb{I}_{(-\infty,0)}\left(Y_{\tau+1}^{(1)}\right), ..., \mathbb{I}_{(-\infty,0)}\left(Y_{\tau+1}^{(n)}\right)\right)$ # non-negativity test

(7.4) if $\mathrm{nonneg} = 0$ goto (5.5)

else do for $i = 1, ..., n$ with $\mathrm{nonneg}[i] \neq 0$: # set negative entries to zero

$$Y_{\tau+1}^i \leftarrow 0$$

(7.5) for $i = 1, ..., n$ # determination of switching variables

if $Y_{\tau+1}^{(i)} < \theta_{g_i}^{(i)}$ then $g_i \leftarrow g_i - 1$

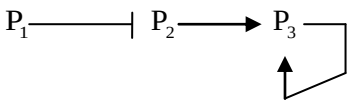
else if $Y_{\tau+1}^{(i)} > \theta_{g_i+1}^{(i)}$ then $g_i \leftarrow g_i + 1$

(7.6) $Y[:, \tau+1] \leftarrow Y_{\tau+1}$

(7.7) $\tau \leftarrow \tau + 1$

A simple example

As in Section 4.2.1 we look at the following simple circuit (from [Thomas 2013]) which involves three proteins P_1, P_2 and P_3:



The circuit is governed by the following logical relationships (where $g_i \in \{0,1\}$, $i = 1,2,3$ are associated Boolean variables describing the state of the genes corresponding to the respective proteins, i.e. $1 \triangleq$ transcription and $0 \triangleq$ no transcription):

$$g_1 = \text{const.} \in \{0,1\},\ g_2 = \neg g_1,\ g_3 = g_2 \vee g_3.$$

In addition proteins are not only produced or not produced by gene expression but they also can possibly be degraded, depending on the situation.

In Section 4.2.1 we saw how to study this system using the (stochastic) GKL framework of Thomas et al. [Thomas, D'Ari 1990]. Here we transform the given circuit information into a PLSDE model and simulate it by means of algorithm 5.2.

In the following we will assume $g_1 = 1$, i.e. the gene of protein P_1 is constantly transcribed no matter what happens otherwise in the system. Biologically this situation could correspond to a gene mutation that accidentally fixes the expression status of a gene to the on-state. It could also be that the gene codes for some very essential protein so that the organism wasn't viable without the respective gene being expressed. In terms of the production constants needed to specify a PL(S)DE system this means $\beta_1^{(g)} = \beta_1$ for all $g \in \mathcal{T} = \{0,1\}^3$ and some production constant $\beta_1 > 0$.

Since the system is classically Boolean with two discrete states for each species it is clear that in terms of PL(S)DEs this means that every species should posses one threshold. Furthermore for simplicity it is assumed that all protein, i.e. P_1, P_2 and P_3, have a constant, domain-independent degradation rate of $\gamma > 0$. Therefore we define $\Gamma^g = \mathrm{diag}(\gamma, \gamma, \gamma)$ for all $g \in \{0,1\}^3$ and for additional simplification we set $\gamma = 1$.

Concerning the production constants $\beta_2^{(g)}$ of protein P_2 we have $\beta_2^{(g)} = \beta_2(\neg g_1)$, i.e. for some $\beta_2 > 0$ we get $\beta_2^{(g)} = \beta_2$ for all $g \in \{0,1\}^3$ with $g_1 = 0$ and $\beta_2^{(g)} = 0$ otherwise in agreement with the logical relationship $g_2 = \neg g_1$. Analogously we have $\beta_3^{(g)} = \beta_3(g_2 \vee g_3)$ for some $\beta_3 > 0$. To fully explore the stochasticity in this simple PLSDE we set $\beta_1 = \beta_2 = \beta_3 = 2$.

It remains to specify the location of the thresholds. Recall that in the PLDE case (see Section 3.6) we have that $\left[0, \frac{\beta_1}{\gamma_1}\right] \times \left[0, \frac{\beta_2}{\gamma_2}\right] \times \left[0, \frac{\beta_3}{\gamma_3}\right] \subset \mathbb{R}^3_{\geq 0}$ is an invariant set and can therefore be interpreted simply as the natural range of the respective protein concentrations. Hence thresholds at $\theta_1^{(i)} = \frac{\beta_i}{2\gamma_i}$ would correspond to the subdivision of the typical concentration range of each species into two equally sized regions. In general one can specify the thresholds according to $\theta_1^{(i)} = \delta_i \frac{\beta_i}{\gamma_i}$ with $\delta_i \in (0,1)$. At this point every threshold value is equally reasonable but since we only want to conceptually indicate the viability of the PLSDE approach the choice equidistant choice, i.e. $\delta_i = 1/2$, is perfectly valid. When dealing with the stochastic PLSDEs, $\left[0, \frac{\beta_1}{\gamma_1}\right] \times \left[0, \frac{\beta_2}{\gamma_2}\right] \times \left[0, \frac{\beta_3}{\gamma_3}\right] \subset \mathbb{R}^3_{\geq 0}$ is of course no longer invariant.

The R-script PLSDE_ExampleOne.R implements algorithm 5.2 specifically for the described example. The script is documented in Appendix A.5 and can also be found on the CD.

Example 5.2.1 Steady states and repressilator with PLSDEs

First recall what should happen in the usual PLSDE scenario. To this end one can run

PLSDE_ExampleOne.R by setting the noise-argument to the zero vector. Plotting the time courses of the concentrations gives (by adjusting production or degradation rates, or thresholds) the following two (out of three, see below) deterministic scenarios which correspond to the two steady state possibilities 100 and 101 which were already encountered in the context of the GLK model from Section 4.2.1 and which by Snoussi's theorem (Subsection 3.6.1) correspond to steady states of the associated PLSDE system:

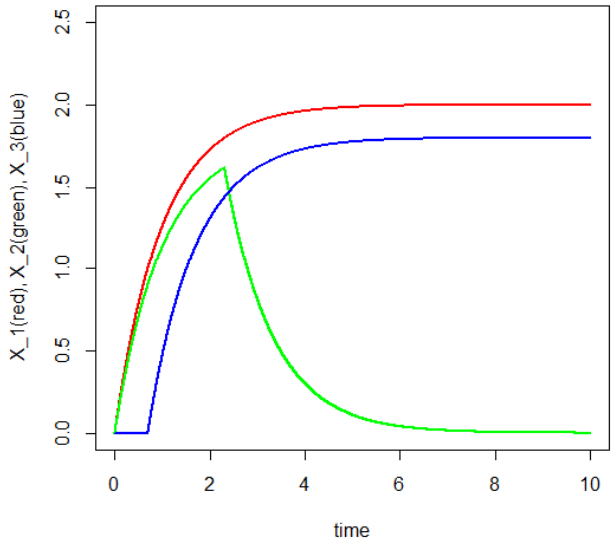


Fig. 5.2.1: Steady state corresponding to 101.
(parameters: noise=c(0,0,0), β=c(2,1.8,1.8),thres1=0.9)

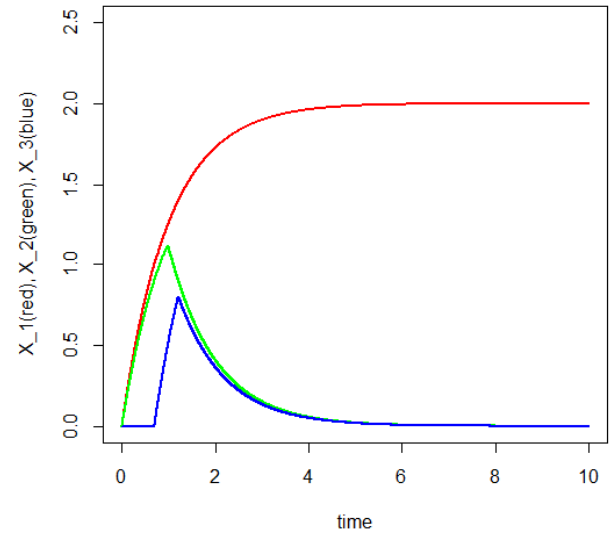


Fig. 5.2.2: Steady state corresponding to 100.
(parameters: noise=c(0,0,0), β=c(2,1.8,2))

What these scenarios mean in terms of the phase space is shown in figures 5.2.3 and 5.2.4 on the next page. What you can see there are just the linear evolutions of the trajectories corresponding to particular domains and domain switches are obviously indicated by the bends in the trajectories.

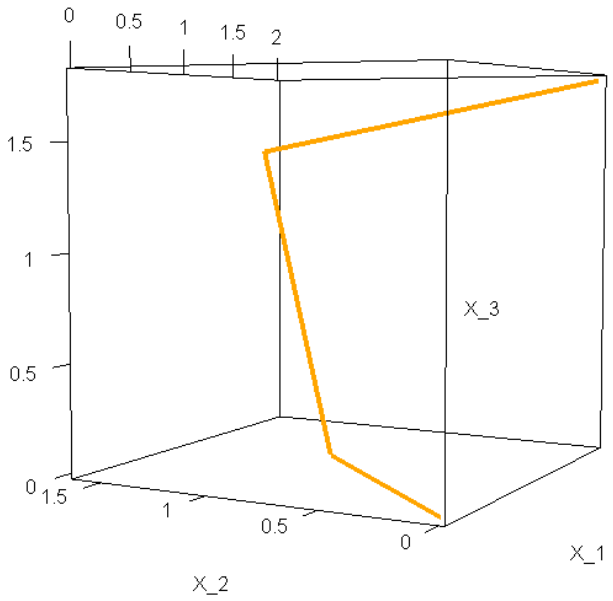


Fig. 5.2.3: Phase space trajectories corresponding to the 101 steady state.
(parameters: noise=c(0,0,0), β=c(2,1.8,1.8),thres1=0.9)

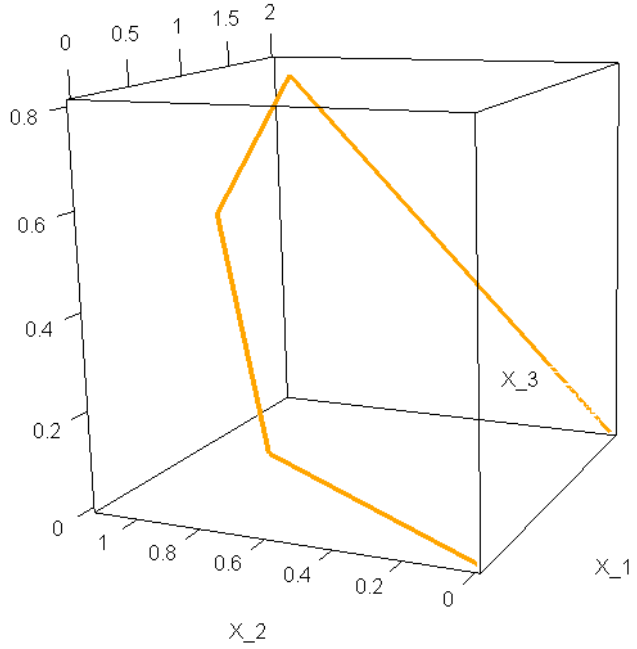
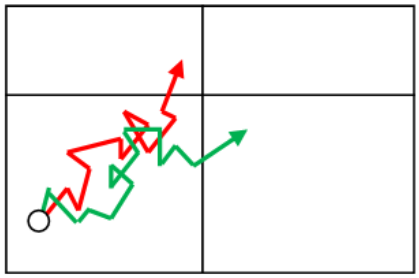


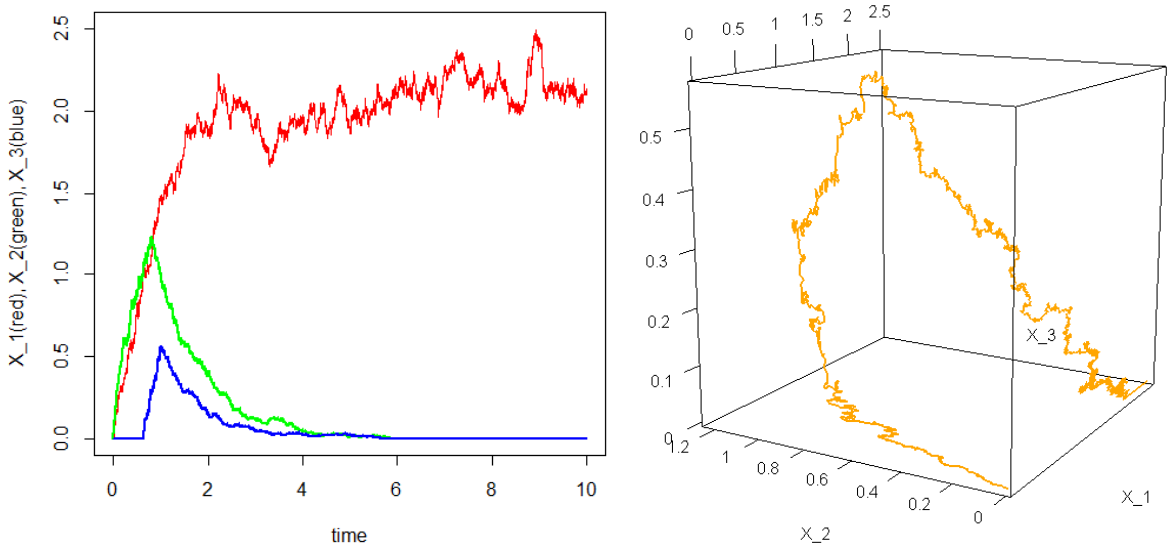
Fig. 5.2.4: Phase space trajectories corresponding to the 100 steady state.
(parameters: noise=c(0,0,0), β=c(2,1.8,2))

Now, as outlined above we want to finally check that the PLSDE formalism is capable of modelling intrinsic noise. The reason for this is the following canonical schematic scenario showing the noisy situation in two dimensions:

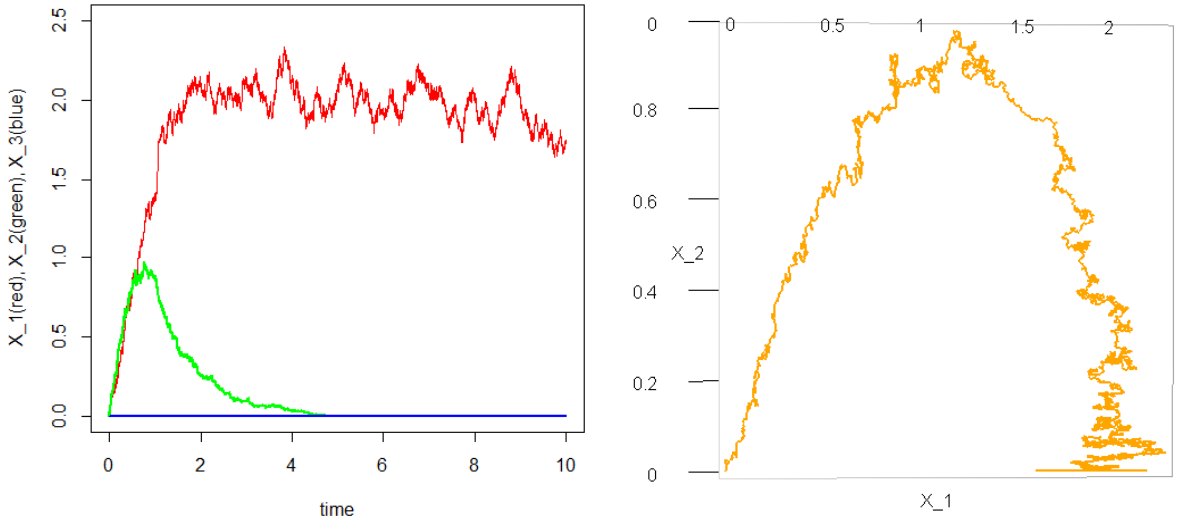


The above scheme shows four boxes corresponding to four domains and although the two trajectories start at the same point in phase space (indicated by the circle) they evolve into different neighbouring domains because of the presence of noise.

By running the script PLSDE_ExampleOne.R at default (see Appendix A.5) multiple times one arrives at three different probabilistic outcomes differing in transient and long-term behaviour. The first one possibility obviously corresponds to the 100 steady state:

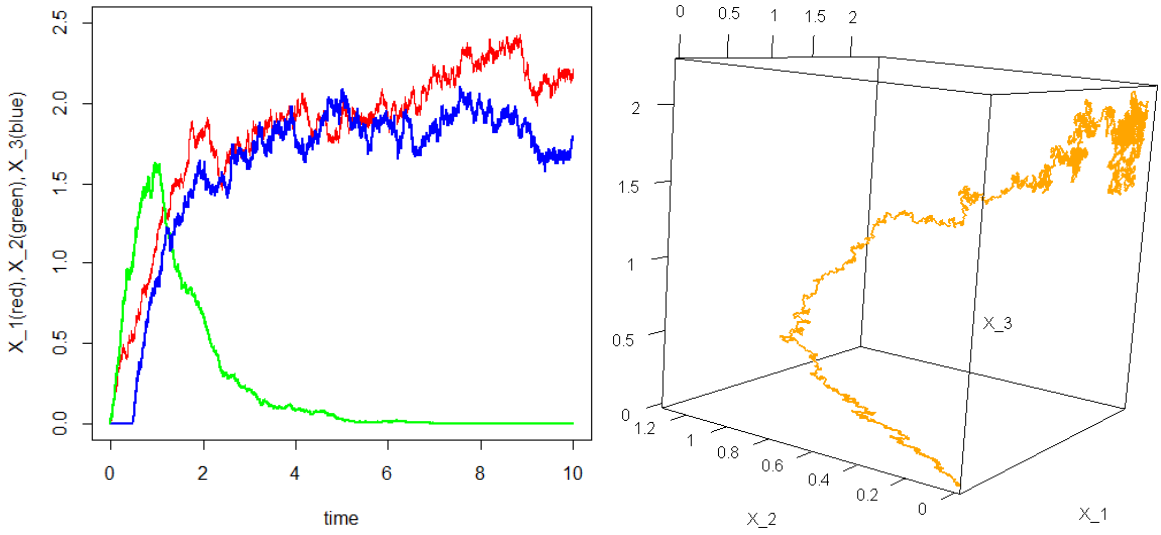


The second (relatively rare) possibility has the same long-term behaviour as the first one but differs from it with respect to the transient behaviour of the third variable (the blue one). Whereas in the first scenario the blue species shows some short-lived peak in its concentration in the second scenario its concentration level always remains zero. Accordingly the dynamics is located on the appropriate 2-dimensional hyperplane. The long-term behaviours on the other hand agree again and correspond to the 100 state.



Of course, by adjusting the parameters in the deterministic PADE system the second scenario would also be possible and given the defining regulatory logic of the system this dynamic

possibility can be interpreted as the case in which the second species never reaches the threshold beyond which species three would be activated, i.e. the height of the green peak is below the threshold of species 2, here this is one. Correspondingly the first scenario above represents the case where species two crosses its threshold and thereby transiently activates species three before both converge to zero. They both go to zero because species two is inactivated as soon species one surpasses its threshold (and this happens always) and species three wasn't able to cross its threshold such that the positive auto-regulatory loop of species three would have been activated. The third possibility which can be observed is the one corresponding to the case where species two crosses its threshold thereby activates species three while this time species two is active (above its threshold) long enough such that species three also passes its threshold and thereafter activates itself. This then leads to the steady state 101 and is shown next:



So, basically the same dynamic range is covered by both the deterministic PLDEs and the stochastic PLSDEs but with the difference that the dynamic ranges are all possible stochastically with one and the same production-, degradation- and threshold parameters, i.e. PLSDEs model noise given by intrinsic molecular fluctuations (what was clear a priori). A PLDE system on the other hand can only model noise with respect to fluctuations of the parameters in different cells while PLSDEs potentially model noise at a single cell level.

Of course, by experimenting with parameters and noise strength many more possibilities show up in the PLSDE framework. For example, in correspondence with the logic described above, repeated peaks of species two show up if one raises the threshold of species one near to its steady state mean value, see figure 5.2.5. Here this mean is equal to two and the threshold can thus be specified to 1.8 for example. The phenomenon of multiple peaks is explicable by the now occurring fluctuations of species one around its threshold concentration. However, in the system described here, the repeated peaks of species two have no further influence on the other two species. But imagine for example a situation where species two regulates a some other species according to some associated threshold which should be below the average peak height. Then, these species would presumably be influenced by the shifted peaks and given the overall dynamic of the enlarged system this could surely have some interesting

consequences and can be seen as a first semi-concrete indication that PLSDE models can naturally have more enriched dynamics than PLDE systems.

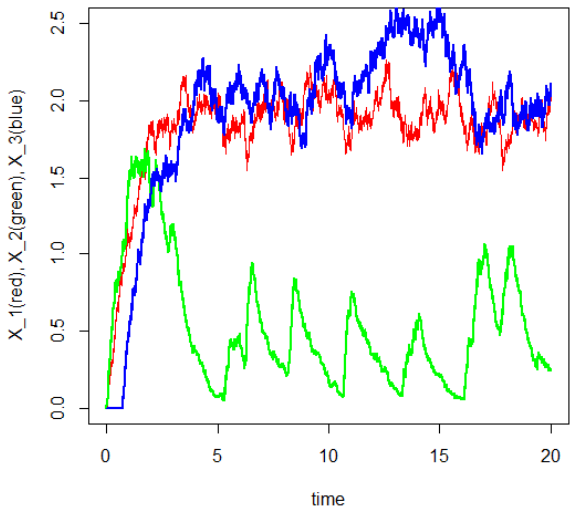
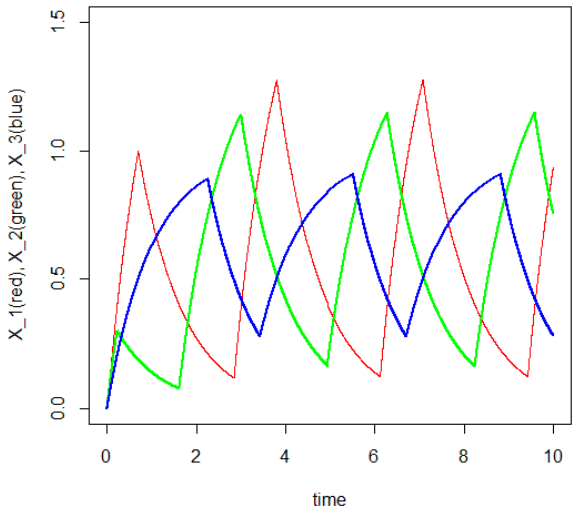


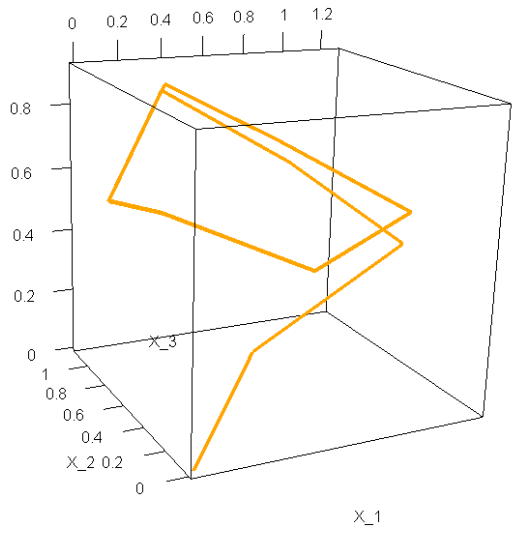
Fig. 5.2.5: **Multiple noise-induced peaks of species 2.**
(parameters: thres1=0.9, t_max =20)

The Repressilator

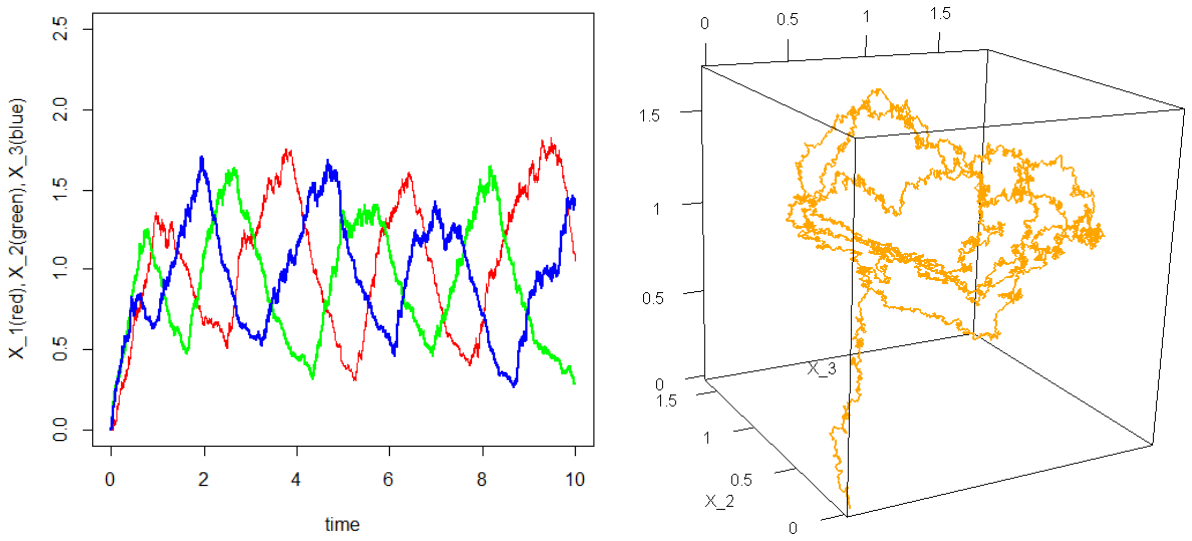
Now that I have illustrated some aspects of the PLSDE approach in the case of an example involving convergence to alternative steady-states I want shortly look at the repressilator circuit (see Subsection 1.3.6). *PLSDE_Repressilator.R* does the job, see Appendix A.5.

If one runs deterministically run the script *PLSDE_Repressilator.R* (i.e. with $\text{noise} = \text{c}(0,0,0)$ and default otherwise) one will notice the *occurrence of sliding mode solutions* manifesting in the state variables ending up being fixed to their respective threshold values. In particular, no oscillations occur. However, adjusting the production parameters and thresholds gives rise to *deterministic oscillations* as is expected for the repressilator model. The following simulations for example can be reproduced by setting $\text{noise} = \text{c}(0,0,0)$, $\beta = \text{c}(2,1.5,1)$ and $\text{thres1} = 0.2$:





Not very surprisingly the oscillatory behaviour can also be obtained in the SPADE case but interestingly, running *PLSDE_Repressilator.R* at default already produces oscillations (in contrast to the deterministic case where sliding mode solutions show up):



In turns out that PLSDEs are a (very, very simple) special case of a more general model calls: general stochastic hybrid systems. These are now shortly indicated in the following subsection.

5.3 General stochastic hybrid systems (GSHS)

Generally speaking, *hybrid systems* are systems (or rather models of systems) which incorporate discrete and continuous states [Ahmad et al. 2007], [Lygeros et al. 2003], [Lygeros 2004], [Lunze, Lamnabhi-Lagarrigue (eds.) 2009]. PLDEs for example are hybrid systems in the sense that a continuous differential evolution represented by the continuous protein concentrations is governed by discrete states indicating the respective domains. The discrete states switch so to speak if the continuous trajectory reaches a switching threshold (and nothing fancy like a sliding mode solution for example shows up). Analogously the PLSDEs briefly examined in the preceding section are also hybrid systems in this sense and because they are governed by some kind of stochasticity one could call a PLSDE a *stochastic hybrid system*.

In the case of PL(S)DEs the discrete switching dynamics is completely determined by the continuous evolution in the sense that switching only occurs when the continuous dynamics reaches another domain. The respective domain then in turn determines the continuous dynamics, either deterministically or stochastically. In general one could also imagine hybrid systems where the switching is governed by a random process independent of the continuous

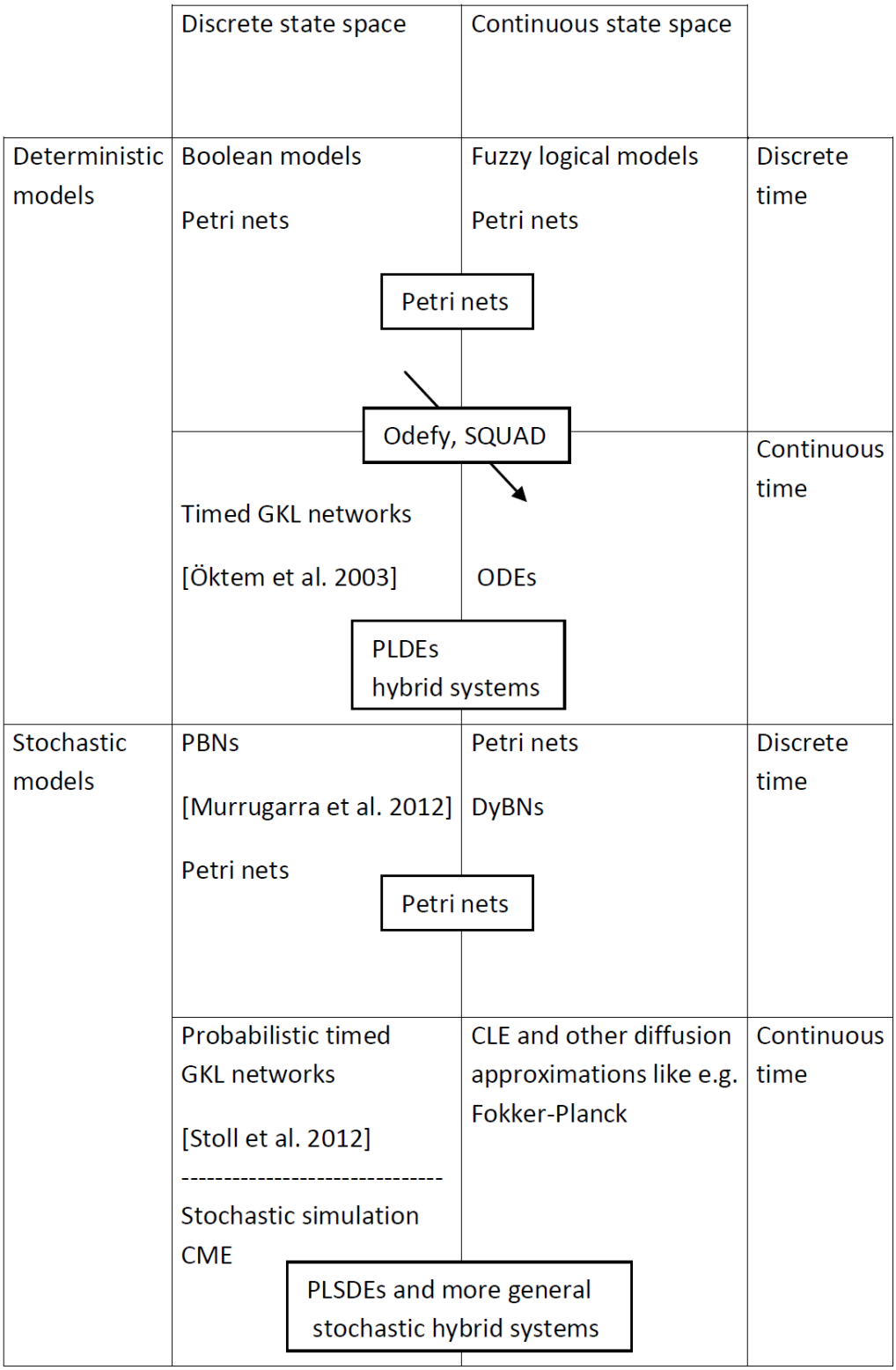
evolution or depends on it in some more intricate way than in PL(S)DEs. One can add even more subtleties, for example by stochastically choosing a respective initial value every time a switch forces a new continuous dynamics. In PLDE models, this would correspond to randomly (according to some law) choosing concentration values within a new domain whenever a trajectory reaches that domain, see Subsection 5.1.3. In principle, this choice could then depend on the history of the continuous trajectory or just on its last position or it could be independent from the continuous trajectory and just depend on the new domain, and so on and so forth.

Of course, such systems are already subject of study in seemingly all possible kinds of variations. According to [Pola et al. 2003], concerning stochastically influenced hybrid systems [Cassandras, Lygeros (eds.) 2007], [Blom, Lygeros (eds.) 2006], researchers invented for example (!) so called *piecewise deterministic Markov processes* (PDMPs) [Davis 1993], *switched diffusion processes* (SDPs) [Ghosh et al. 1997], [Yin, Zhu 2010], models just termed *stochastic hybrid systems* (SHSs) [Hu et al. 2000] or so called *stochastic timed automata* [Kwiatkowska et al. 2000]. As reviewed in [Pola et al. 2003] these model classes are different but nonetheless related under circumstances depending on the respective model specifications. All these model classes formalize (in one way or another) the notion of systems which possess both a discrete as well as a continuous state space component and which are governed by some kind of stochasticity. [Bujorianu, Lygeros 2008] proposed a framework which unifies some of the approaches taken in the literature and termed their general model class *general stochastic hybrid systems* (GSHSs).

Stochastic hybrid systems (in the sense of models which possess both a discrete as well as a continuous state space component and which are governed by some kind of stochasticity) have been also been applied to biochemical networks: [Singh, Hespanha 2010] is a general review on the topic while [Kouretas et al. 2006], [Cinquemani et al. 2007], [Crudu et al. 2009] or [Crudu et al. 2011] may represent (!) more specialized studies on stochastic hybrid systems in systems biology.

6 Summary and Outlook

The following table basically summarizes the conceptual part of the thesis (and accordingly more or less ignores spatial models):



[Jamshidi 2012] provides an interesting detailed study on the relation between PLDEs and different kinds of discrete models. While the situation is already very intricate for the considered deterministic systems, it might nonetheless be worthwhile to explore possible stochastic extensions. This would ultimately lead to the study of stochastic hybrid systems as outlined in Subsection 5.3. Related to these issues is the conception that the established qualitative simulation approach of [De Jong et al. 2004a,b] based on discrete abstractions of PLDEs should also be accessible to stochastic extensions (at least in theory by using PLSDEs) and it would certainly be interesting to be able to conduct *stochastic qualitative simulation*.

Concerning the estimation of parameters in probabilistic timed GKL networks it is (although they are biologically not extremely plausible) interesting to try to obtain moderately general results which relate the structure of the network with the corresponding parameter estimation task as outlined in Subsections 4.2.1 to 4.2.3. In addition, it would certainly be useful to apply much heavier statistical machinery as basic Markov chain theory in order for example to meaningfully be able to deal with the mixture models proposed in Subsection 4.2.4. The same holds of course even more for time series based estimation. Furthermore, one could look out for further "feasible" delay distributions and also consider mixed systems where the delay can come from different distribution families. It would also be nice to find a non-trivial application for one of the approaches.

A long term theoretical goal would be to formally relate as many modeling approaches as possible by adopting the idea that it could "somehow" be possible to "construct" a "continuum" of models connecting even the simplest ones via a chain of intricate mathematically strictly backed up transformations such that certain properties remain invariant under these transformations. This could also be used potentially to guide parameter estimation by clever "model jumping".

While the last two or three sentences might sound (although philosophically acceptable) a little bit spongy a clear-cut short-term goal could be to apply the PLSDE approach to a PLDE model of [Chaves et al. 2009] which models the interplay of the NFκB pathway and apoptosis. There, noise was modeled in terms of parameter fluctuations from cell to cell and it would be interesting to examine the same system in terms of a modeling framework which is intrinsically stochastic. I already tried my luck but ran into numerical problems which can almost surely be attributed to the size of the system and my "spontaneous" numerical capabilities. In principle, it seems a worthwhile goal to extend numerical approaches which exist for piecewise linear models [Stewart 1990] to PLSDE systems. Considering the huge time span from 1990 to 2014 it might seem also worthwhile to look for already existing solutions first.

A.1 Exponential and Weibull distribution

This appendix provides the facts on the exponential and the Weibull distribution needed for the main part of the text. Throughout, random variables X are implicitly given as measurable functions $X:\Omega\to\mathbb{R}^d$ where $(\Omega,\mathcal{F},\mathbb{P})$ is a probability space and $\left(\mathbb{R}^d,\mathcal{B}_{\mathbb{R}^d}\right)$ is the measurable space given by the d-dimensional Euclidean space with Borel-σ-algebra $\mathcal{B}_{\mathbb{R}^d}$ [Klenke 2008]. Random variables with range $A\subset\mathbb{R}^n$ are understood to be defined with respect to corresponding induced σ-algebra $(A,\mathcal{B}_A)$. The induced probability measure (the distribution) of X is denoted by $\mathbb{P}_X := \mathbb{P}\circ X^{-1}$ [Klenke 2008].

Exponential distribution

A random variable T is said to be exponentially distributed with parameter $\lambda>0$ if the law $\mathbb{P}_T$ of T has the probability density function (pdf) $f_T(t)=\mathbb{I}(t\geq 0)\lambda e^{-\lambda t}$. The corresponding cumulative distribution function (cdf) is given by $F_T(t)=\mathbb{I}(t\geq 0)\left(1-e^{-\lambda t}\right)$. We write $X\sim \mathrm{Exp}(\lambda)$. The following result shows that the exponential distribution is characterized by its so called *memoryless property*.

Result A.1.1 (**Memoryless property of the exponential distribution**) [Norris 1998]

> A random variable T with range $(0,\infty]$ has an exponential distribution if and only if the following memoryless property holds:
>
> $$\forall s,t\geq 0:\ \mathbb{P}\left(T>t+s\middle|T>t\right)=\mathbb{P}(T>s).$$

Proof [Norris 1998]:

If $T\sim\mathrm{Exp}(\lambda)$ it follows for all $s,t\geq 0$ (since $\mathbb{P}(T>t)>0$ for all t):

$$\mathbb{P}\left(T>t+s\middle|T>t\right)=\frac{\mathbb{P}(T>s+t)}{\mathbb{P}(T>t)}=\frac{e^{-\lambda(t+s)}}{e^{-\lambda t}}=e^{-\lambda s}=\mathbb{P}(T>s).$$

Reversely, suppose $\forall s,t\geq 0:\ \mathbb{P}\left(T>t+s\middle|T>t\right)=\mathbb{P}(T>s)$, then see [Norris 1998] p. 71. ■

The memoryless property can be slightly reformulated as follows:

$$T \sim \mathrm{Exp}(\lambda) \qquad \Leftrightarrow \qquad \forall s \geq 0 \colon \ (T - s \,|\, T > s) \sim \mathrm{Exp}(\lambda).$$

Further, notice that $\mathbb{E}[T] = \frac{1}{\lambda}$ for $T \sim \mathrm{Exp}(\lambda)$.

Result A.1.2

$$X_j \sim \mathrm{Exp}(\lambda_j) \text{ independent, } j = 1, \ldots, n \ \Rightarrow \ \mathbb{P}\left(\min(X_1, \ldots, X_n) = X_i\right) = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}$$

Proof:

$$\begin{aligned}
\mathbb{P}\left(\min(X_1, \ldots, X_n) = X_i\right) &= \mathbb{P}\left(\bigcap_{j \neq i} \{X_j > X_i\}\right) \\
&= \int_0^\infty \mathbb{P}\left(\bigcap_{j \neq i} \{X_j > X_i\} \,|\, X_i = t\right) f_{X_i}(t)\, dt \\
&\underset{\{X_j\} \text{ independent}}{=} \int_0^\infty \lambda_i e^{-\lambda_i t} \prod_{j \neq i} \mathbb{P}(X_j > t)\, dt \\
&= \int_0^\infty \lambda_i e^{-\lambda_i t} \prod_{j \neq i} e^{-\lambda_j t}\, dt \\
&= \lambda_i \int_0^\infty \exp\left(-t \cdot \sum_{j=1}^{n} \lambda_j\right) dt = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}
\end{aligned}$$

■

Weibull distribution

A random variable T is said to be Weibull distributed with parameters $\mu, k > 0$ when its law $\mathbb{P}_T$ possesses the pdf $g_T(t) = \mathbb{I}(t \geq 0) \frac{k}{\mu} \left(\frac{t}{\mu}\right)^{k-1} e^{-(t/\mu)^k}$. The corresponding cdf is given by $G_T(t) = \mathbb{I}(t \geq 0)\left(1 - e^{-(t/\mu)^k}\right)$. We write $X \sim \mathcal{WB}(\lambda, k)$. The following result concerns the pdf f_ξ of the random variable $\xi := (T - t \,|\, T > t)$ for $t > 0$ and $T \sim \mathcal{WB}(\lambda, k)$. The result is needed in Section 4.2 in order to sample from $\xi := (T - t \,|\, T > t)$.

Result A.1.3

$$f_{\xi}(s) = \mathbb{I}(t \geq 0) \cdot \frac{k}{\mu}\left(\frac{t+s}{\mu}\right)^{k-1} \cdot \exp\left(\left(\frac{t}{\mu}\right)^{k} - \left(\frac{t+s}{\mu}\right)^{k}\right)$$

Proof:

Since $\mathbb{P}(T > t) > 0$ for all t is follows:

$$\begin{aligned}\mathbb{P}(\xi > s) = \mathbb{P}(T > t+s \mid T > t) &= \frac{\mathbb{P}(T > t+s)}{\mathbb{P}(T > t)} \\ &= \frac{\mathbb{I}(t \geq 0)\exp\left(-\left(\frac{t+s}{\mu}\right)^{k}\right)}{\mathbb{I}(t \geq 0)\exp\left(-\left(\frac{t}{\mu}\right)^{k}\right)} \\ &= \mathbb{I}(t \geq 0)\exp\left(\left(\frac{t}{\mu}\right)^{k} - \left(\frac{t+s}{\mu}\right)^{k}\right) =: R(s)\end{aligned}$$

Therefore, $f_{\xi}(s) = -R'(s) = \mathbb{I}(t \geq 0) \cdot \frac{k}{\mu}\left(\frac{t+s}{\mu}\right)^{k-1} \cdot \exp\left(\left(\frac{t}{\mu}\right)^{k} - \left(\frac{t+s}{\mu}\right)^{k}\right)$. ■

A.2 Sampling from probability distributions

This appendix is based on Chapter 2 of [Robert, Casella 2000] and Chapter 3 of [Rizzo 2008] and is fitted to the needs of this thesis and is definitely not meant to be any kind of even semi-extensive overview. Throughout, random variables X are implicitly given as measurable functions $X : \Omega \to \mathbb{R}^d$ where $(\Omega, \mathcal{F}, \mathbb{P})$ is a probability space and $\left(\mathbb{R}^d, \mathcal{B}_{\mathbb{R}^d}\right)$ is the measurable space given by the d-dimensional Euclidean space with Borel-σ-algebra $\mathcal{B}_{\mathbb{R}^d}$ [Klenke 2008]. Random variables with range $A \subset \mathbb{R}^n$ are understood to be defined with respect to corresponding induced σ-algebra $(A, \mathcal{B}_A)$. The induced probability measure (the distribution) of X is denoted by $\mathbb{P}_X := \mathbb{P} \circ X^{-1}$.

The basis of drawing samples from a given probability distribution is the ability to draw reliable sample from a uniform distribution $U \sim \mathcal{U}[0,1]$. Algorithms which generate such sample are called *uniform pseudo-random number generators* [Robert, Casella 2000]. Such

algorithms generate deterministic (!) sequences $(u_i)=\left(D^i(u_0)\right)\in[0,1]^{\mathbb{N}_0}$ starting from a deterministic (!) initial value $u_0\in[0,1]$ and with $D:[0,1]\to[0,1]$ being a 'suitable' deterministic (!) transformation. If the deterministic sequences $(u_1,...,u_n)$ for $n\in\mathbb{N}$ can be reliably seen as an uniform i.i.d. sample $(V_1,...,V_n)$, i.e. $V_i\sim\mathcal{U}[0,1]$ independently, has to be assessed by testing the null hypothesis $H_0:\ \forall n\in\mathbb{N}: U_1,...,U_n\sim_{iid}\mathcal{U}[0,1]$ which can for example be done by *Kolmogorov-Smirnov tests* [Rizzo 2008: Chapter 8] or by utilizing concepts from *time series analysis* as indicated by [Robert, Casella 2000: p.37]. One famous uniform pseudo-random number generator is the so called *Kiss generator*, see for example [Robert, Casella 2000: Subsection 2.1.2].

Theoretically every random variable $X:\Omega\to\mathbb{R}$ can be simulated based on an i.i.d. sample of uniformly distributed random variables. This is due to the fact sometimes known as *probability integral transform* [Robert, Casella 2000: p.36]. Given the cumulative distribution function (cdf) $F_X:\mathbb{R}\to[0,1]$, $F_X(x)=\mathbb{P}(X\le x)$ of X one defines the *generalized inverse* of F_X as the function $F_X^-:[0,1]\to\mathbb{R}$ via $F_X^-(u):=\inf\left\{x\in\mathbb{R}:F_X(x)\ge u\right\}$. For absolutely continuous distributions the generalized inverse is just the inverse of the cdf. The probability integral transform theorem can then be stated as follows: If $U\sim\mathcal{U}[0,1]$, then $F_X^-(U)\sim\mathbb{P}_X$.

Practically the probability integral transform can only be used to generate samples of a random variable X if the corresponding generalized inverse is easily computable. This is however most often not case. The exponential distribution and the Weibull distribution are two examples where the integral transform method works.

As given in the preceding appendix, the cdf of a random variable $X\sim\mathcal{WB}(\lambda,k)$ with scale parameter $\lambda>0$ and shape parameter $k>0$ is given by $F_X(x)=\mathbb{I}(x\ge0)\left(1-\exp\left\{-\left(\frac{x}{\lambda}\right)^k\right\}\right)$.

Since X is absolutely continuous one can set for $u\in[0,1]$

$$u=F_X\left(F_X^-(u)\right)=1-\exp\left\{-\left(\frac{F_X^-(u)}{\lambda}\right)^k\right\}$$

and arrive at $F_X^-(u)=\lambda\left(-\ln(1-u)\right)^{1/k}$. Specializing to $k=1$ gives the result for $X\sim\mathrm{Exp}\left(\frac{1}{\lambda}\right)$. So to generate a (pseudo) sample $X_1,...,X_n\sim_{iid}\mathcal{WB}(\lambda,k)$ from generate $U_1,...,U_n\sim_{iid}\mathcal{U}[0,1]$ with a uniform pseudo-random number generator and set

$$X_i=\lambda\left(-\ln(1-U_i)\right)^{1/k},\ i=1,...,n.$$

An application of the inverse transform for a discrete random variable is given by the following method to generate a sample $X_1,...,X_n \sim_{iid} \mathrm{Cat}(\pi)$ from a categorical m-valued random variable with $\pi \in S_m$ for some $m \in \mathbb{N}$ (see Appendix A.X). To do this, you can sample $U_1,...,U_n \sim_{iid} \mathcal{U}[0,1]$ and set $X_i = k\mathbb{I}\left(\sum_{j=1}^{k-1}\pi_j \le U_i < \sum_{j=1}^{k}\pi_j, k<m\right) + m\mathbb{I}\left(\sum_{j=1}^{m-1}\pi_j \le U_i \le 1\right)$ for $i=1,...,n$.

To draw a sample $X_1,...,X_n \sim_{iid} \mathcal{N}_d(\mu,\Sigma)$ of a d-dimensional multivariate normal distribution with mean $\mu \in \mathbb{R}^d$ and symmetric positive definite covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$ (see the preceding appendix) simple inverse transform methods are not applicable. The classical *Box-Muller method* could be used to draw a sample if $d=2$ and $\Sigma \in \mathbb{R}^{2\times 2}$ diagonal [Robert, Casella 2000: p.46].

In the general case, the basic principle is given as follows. One first samples $\tilde{X}_i \sim \mathcal{N}_d(0, I_d)$ and then, based on a suitable decomposition $\Sigma = C^T C$ with some matrices C, obtain the sample $X_i := C\tilde{X}_i + \mu \sim \mathcal{N}_d(\mu,\Sigma)$. Suitable decompositions include the spectral decomposition, the Cholesky decomposition or the singular value decomposition (SVD) [Rizzo 2008: Subsection 3.6.1].

For extensive treatments of the topic it is referred to [Robert, Casella 2000] and [Rizzo 2008].

For all practical purposes it seems advisable to use the many built-in sampling algorithms of R or MATLAB.

A.3 Adaptive rejection sampling of Weibull random variables T conditioned on events of the form $\{T>t\}$ for $t>0$

In order to be able to simulate the probabilistic timed GKL networks with Weibull distributed time delays introduced in Section 4.1, it is necessary to sample from conditioned random variables $T|T>t$ with $T \sim \mathcal{WB}(\lambda,k)$, $t,\lambda>0$ and $k \ge 1$. If $k=1$, then $T \sim \mathrm{Exp}(\lambda)$ and the task of sampling from $T|T>t$ simply reduces to sampling from $T \sim \mathrm{Exp}(\lambda)$ because of the memoryless property of the exponential distribution. Some basic definitions and results on the exponential and the Weibull distribution can be found in Appendix A.1. If $k>1$, it turns out that the density of $T|T>t$ (or equivalently of $\xi = (T-t|T>t)$) is *log-concave* and is thus predestined to be simulated with *adaptive rejection sampling*.

This section is subdivided into two subsections. Subsection A.3.1 shortly outlines adaptive rejection sampling (ARS) for log-concave densities while Subsection A.3.2 applies these concepts to sampling from $T|T>t$ with $T\sim\mathcal{WB}(\lambda,k)$, $t,\lambda>0$ and $k>1$.

A.3.1 Adaptive rejection sampling (ARS) for log-concave densities

Adaptive rejection sampling (ARS) is an instance of the general class of so called *accept-reject sampling methods* [Robert, Casella 2000: Section 2.3]. The basic accept-reject algorithm to sample from a (continuous) density $f:D_f\to\mathbb{R}, D_f\subset\mathbb{R}^d$ (called *target density*) relies on the existence of some so called *instrumental density* $g:D_g\to\mathbb{R}, \operatorname{supp}(f)\subset D_g\subset\mathbb{R}^d$ such that there is an $M>0$ with $f(x)\le Mg(x)$ for all $x\in\operatorname{supp}(f):=\{x\in D_f: f(x)>0\}$. In addition, it is important that it is comparatively easy to sample from the instrumental density g. A convenient general characteristic of accept-reject methods is that the target density needs only be known up to a constant, i.e. is suffices to know $\tilde{f}:D_{\tilde{f}}\to\mathbb{R}, \operatorname{supp}(f)\subset D_{\tilde{f}}\subset\mathbb{R}^d$ with $\tilde{f}=c\cdot f$ with unknown instrumental density f and unknown normalization constant $c>0$. This is very useful in a Bayesian setting where one wants to sample from a posterior distribution which is only known to be proportional to the prior times the likelihood.

Algorithm A.3.1 [**Accept-reject method**]

Input: I1. $\tilde{f}:D_f\to\mathbb{R}$ with $\tilde{f}=c\cdot f$ for $c>0$ and a density $f:D_f\to\mathbb{R}$, $D_f\subset\mathbb{R}^d$
target density
I2. A density $g:D_g\to\mathbb{R}, \operatorname{supp}(f)\subset D_g$ with $\tilde{f}(x)\le Mg(x)$
for some $M>0$ and all $x\in\operatorname{supp}(f)$ # instrumental density
I3. $n\in\mathbb{N}$ # sample size

Output: O1. $Y_1,...,Y_n\sim_{iid} f$

(1) $k\leftarrow 0$
(2) while $k<n$:
(2.1) sample $U\sim\mathcal{U}[0,1]$
(2.2) sample $X\sim g$
(2.3) if $U\le\dfrac{\tilde{f}(X)}{Mg(X)}$:

(2.3.1) $Y_{k+1} \leftarrow X$

(2.3.2) $k \leftarrow k+1$

Lemma A.3.1 (**Accept-reject method**)

The sample $Y_1,...,Y_n$ generated with algorithm 5.1 is i.i.d. according to f.

Proof [Robert, Casella 2000: Section 2.3]:

$Y_1,...,Y_n$ are independent because U and X are independently sampled anew in each iteration and hence independence follows for example from Satz 2.26 in [Klenke 2008].

If one denotes the joint density of U and X by $\varphi(x,u) = \mathbb{I}\left(u \in [0,1]\right) g(x)$ (U and X are independent) one obtains for $y = (y_1,...,y_d)^T \in D_g$:

$$\mathbb{P}(Y \le y) = \mathbb{P}\left(X \le y \middle| U \le \frac{\tilde{f}(X)}{Mg(X)} \right)$$

$$= \frac{\mathbb{P}\left(X \le y, U \le \frac{\tilde{f}(X)}{Mg(X)} \right)}{\mathbb{P}\left(U \le \frac{\tilde{f}(X)}{Mg(X)} \right)}$$

$$= \frac{\int_{\prod_{i=1}^{d}(-\infty,y_i]} \int_0^{\tilde{f}(x)/Mg(x)} \phi(x,u)\,du\,dx}{\int_{D_g} \int_0^{\tilde{f}(x)/Mg(x)} \phi(x,u)\,du\,dx}$$

$$= \frac{\int_{\prod_{i=1}^{d}(-\infty,y_i]} \int_0^{\min\{\tilde{f}(x)/Mg(x),1\}} du\; g(x)\,dx}{\int_{D_g} \int_0^{\min\{\tilde{f}(x)/Mg(x),1\}} du\; g(x)\,dx}$$

$$\underset{\substack{\min\{\tilde{f}(x)/Mg(x),1\}=\tilde{f}(x)/Mg(x) \\ \text{since } \tilde{f}(x) \le Mg(x)}}{=} \frac{\frac{1}{M}\int_{\prod_{i=1}^{d}(-\infty,y_i]} \tilde{f}(x)\,dx}{\frac{1}{M}\int_{D_g} \tilde{f}(x)\,dx}$$

$$= \frac{\int_{\prod_{i=1}^{d}(-\infty, y_i]} c f(x)\,dx}{\int_{D_f} c f(x)\,dx}$$

$$= \int_{\prod_{i=1}^{d}(-\infty, y_i]} f(x)\,dx = F_f(y)$$

■

The probability of acceptance p_{accept} in algorithm 5.1 given by

$$p_{accept} := \mathbb{P}\left(U \le \frac{\tilde{f}(X)}{Mg(X)} \right) = \int_{D_g} \int_0^{\tilde{f}(x)/Mg(x)} \phi(x,u)\,du\,dx = \frac{c}{M}$$

represents the "efficiency" of a given accept-reject method. The higher p_{accept} the more efficient is the algorithm in the sense that it needs (on average) less iterations to produce a sample of a given length.

Even if one has a suitable instrumental distribution for a given target density it can be the case that the target f (i.e. $\tilde{f}$) is difficult to evaluate. In this case, algorithm 5.1 come into trouble since in every iteration, $\tilde{f}$ has to be evaluated at the simulated X (step (2.3) in algorithm A.3.1). One way to overcome this bottleneck is by means of so called *envelope accept-reject methods* [Robert, Casella 2000: Subsection 2.3.2]. While the instrumental density g bounds the target density from above (scaled with some constant M), envelope accept-reject methods additionally introduce a further instrumental function h (which does not need to be a density) which bounds the target density from below. g is also called *rejection envelope* and h is also known as *squeezing function* [Gilks, Wild 1992].

Algorithm A.3.2 [***Envelope accept-reject method***]

Input: I1. $\tilde{f}: D_f \to \mathbb{R}$ with $\tilde{f} = c \cdot f$ for $c > 0$ and a density $f: D_f \to \mathbb{R}$, $D_f \subset \mathbb{R}^d$
target density

I2. A density $g: D_g \to \mathbb{R}$, $\text{supp}(f) \subset D_g$ with $\tilde{f}(x) \le Mg(x)$
for some $M > 0$ and all $x \in \text{supp}(f)$ # instrumental density

I3. $h: D_h \to \mathbb{R}$, $\text{supp}(f) \subset D_h$ such that $h(x) \le \tilde{f}(x)$ for all $x \in \text{supp}(f)$
instrumental lower bound on $\tilde{f}$

I4. $n \in \mathbb{N}$ # sample size

Output: O1. $Y_1, ..., Y_n \sim_{iid} f$

---

(1) $k \leftarrow 0$

(2) while $k < n$:

(2.1) sample $U \sim \mathcal{U}[0,1]$

(2.2) sample $X \sim g$

(2.3) if $U \le \frac{h(X)}{Mg(X)}$:

(2.3.1) $Y_{k+1} \leftarrow X$

(2.3.2) $k \leftarrow k+1$

(2.4) else:

(2.4.1) if $U \le \frac{\tilde{f}(X)}{Mg(X)}$:

(2.4.1.1) $Y_{k+1} \leftarrow X$

(2.4.1.2) $k \leftarrow k+1$

Lemma A.3.2 (**Envelope accept-reject method**)

The sample $Y_1,...,Y_n$ generated with algorithm 5.2 is i.i.d. according to f.

Proof:

Independence follows by the same argument as in the proof of lemma A.3.1.

Concerning the distribution of the sample $Y_1,...,Y_n$ it suffices to notice that a proposal is accepted if and only if $U \le \frac{\tilde{f}(X)}{Mg(X)}$ since $h \le \tilde{f}$ on $\text{supp}(f)$. Hence the same computation as in the proof of lemma 5.1.1 shows that $Y_1,...,Y_n$ are indeed identically distributed according to the target density f. ■

The difference between algorithm A.3.1 and A.3.2 is simply that in algorithm A.3.2 the target $\tilde{f}$ is only evaluated, if $U \le \frac{\tilde{f}(X)}{Mg(X)}$ cannot be ascertained by noticing the stronger condition $U \le \frac{h(X)}{Mg(X)} \le \frac{\tilde{f}(X)}{Mg(X)}$. So, in order to avoid many evaluations of $\tilde{f}$ it is administrable if h is as tight a lower bound of $\tilde{f}$ as possible. Of course, h should be easy to evaluate.

A short look at algorithms A.3.1 and A.3.2 and the respective proofs of their correctness reveals that independence does not depend on g or h. In addition, the derivation that the single samples Y_k follow the desired distribution is viable for every iteration in isolation and hence one could possibly use different instrumental functions g and h for different iterations given that they fulfill the demanded properties for instrumental functions. This is now, to a first approximation, one of the key ideas of adaptive rejection sampling, the other one being to choose the instrumental functions in an adaptive manner in order to achieve a successively tighter envelope. In fact, ARS even changes g and h after samples were rejected and hence the just described logic

Adaptive rejection sampling was invented by [Gilks, Wild 1992] in the context of analyzing monoclonal antibody data with Bayesian methods. In order to be able to conduct the desired Bayesian analysis it was necessary to use a Gibbs sampling approach and in order to be able to sample from the involved conditional distributions [Gilks, Wild 1992] devised adaptive rejection sampling.

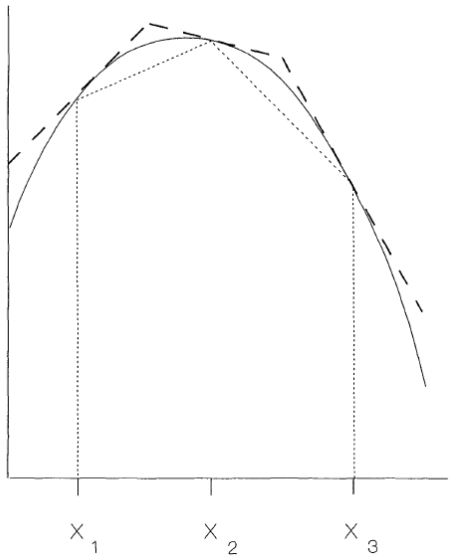
Adaptive rejection sampling is universal in the sense that it provides a automatic method to sample from any log-concave density. A function $f : D \to \mathbb{R}$ with convex $D \subset \mathbb{R}^d$ is said to be log-concave if its logarithm is concave, i.e. if

$$\forall x, y \in D,\ \theta \in [0,1]:\ \log f\left(\theta x + (1-\theta) y\right) \geq \theta \log f\left(x\right) + (1-\theta) f\left(y\right).$$

In the next subsection it will be shown that the density of $\left(T - t \middle| T > t\right)$ with $T \sim \mathcal{WB}(\lambda, k)$, $t, \lambda > 0$ and $k > 1$ is log-concave what leads to the possibility to sample from $\left(T - t \middle| T > t\right)$ (and hence $T \middle| T > t$) by means of ARS. For twice differentiable functions log-concavity is equivalent to the negativity of its logarithm due to basic facts from analysis.

The main idea of adaptive rejection sampling is best explained with figure A.3.1 in mind.

Fig. A.3.1 (from [Gilks, Wild 1992])



What is shown, is a log-concave density function and its so called lower and upper hulls. In the following I follow [Robert, Casella 2000]. Given a log-concave density $f : D \to \mathbb{R}$ and a set of support points $\mathcal{S}_n := \{x_0, , x_1, ..., x_{n+1}\} \subset D$ such that the log of f is known up to some constant at the points in $\mathcal{S}_n$. Now, ARS proceeds by defining a piecewise linear rejection envelope by taking the respective tangents at the support points and joining them in the canonical way. Analogously piecewise linear squeezing function is constructed by taking the secants from support point to support point. After consistently defining the envelope at the boundaries of the domain of f (see [Robert,

Casella 2000: Subsection 2.3.3]), the basic ARS algorithm can be formulated as follows. Given a set of support points construct the squeezing function and the rejection envelope as outlined and perform a respective envelope-rejection step (steps (2.3) and (2.4) in algorithm A.3.2). If you have to go to step (2.4) take the respective sampled x and join it to the support points (notice, since you had to go to step (2.4) you already evaluated f at x and therefore the logarithm is also given.). And so on and so forth. After the proofs of Lemmas A.3.1 and A.3.2 it should be clear that ARS works. The overall logic is such that whenever you have to evaluate f, you join the respective sample to the support points in order to tighten up the envelope at the encountered "leaky" point. As a last remark, I point out that sampling from the (transformed) upper hull is relatively easy [Robert, Casella 2000: Subsection 2.3.3].

A.3.2 Sampling from the conditioned Weibull distribution with ARS

This subsection shows that $\xi = \left(T - t \middle| T > t\right)$ with $T \sim \mathcal{WB}(\lambda, k)$, $t, \lambda > 0$ and $k > 1$ can be sampled by means of ARS by showing that the pdf (probability density function) f_ξ given by

$$f_\xi(s) = \mathbb{I}(s \geq 0) \cdot \frac{k}{\mu}\left(\frac{t+s}{\mu}\right)^{k-1} \cdot \exp\left(\left(\frac{t}{\mu}\right)^k - \left(\frac{t+s}{\mu}\right)^k\right) \quad \text{(see Appendix A.1, result A.1.2)}$$

is log-concave and hence accessible by ARS. To show that f_ξ is log-concave it suffices to show that the second derivative of the log of f_ξ is negative on its domain (see the preceding subsection). Let $s \geq 0$, then one has

$$\ell_\xi(s) := \log f_\xi(s) = \log\frac{k}{\mu} + (k-1)\log\left(\frac{t+s}{\mu}\right) + \left(\left(\frac{t}{\mu}\right)^k - \left(\frac{t+s}{\mu}\right)^k\right)$$

and therefore

$$\frac{d}{dt}\ell_\xi(s) = \frac{(k-1)}{t+s} - \frac{k}{\mu}\left(\frac{t+s}{\mu}\right)^{k-1}$$

i.e. finally

$$\frac{d^2}{dt^2}\ell_\xi(s) = \frac{(1-k)}{(t+s)^2} - \frac{k(k-1)}{\mu^2}\left(\frac{t+s}{\mu}\right)^{k-2} < 0$$

since $t, \lambda > 0$, $k > 1$ and $s \geq 0$.

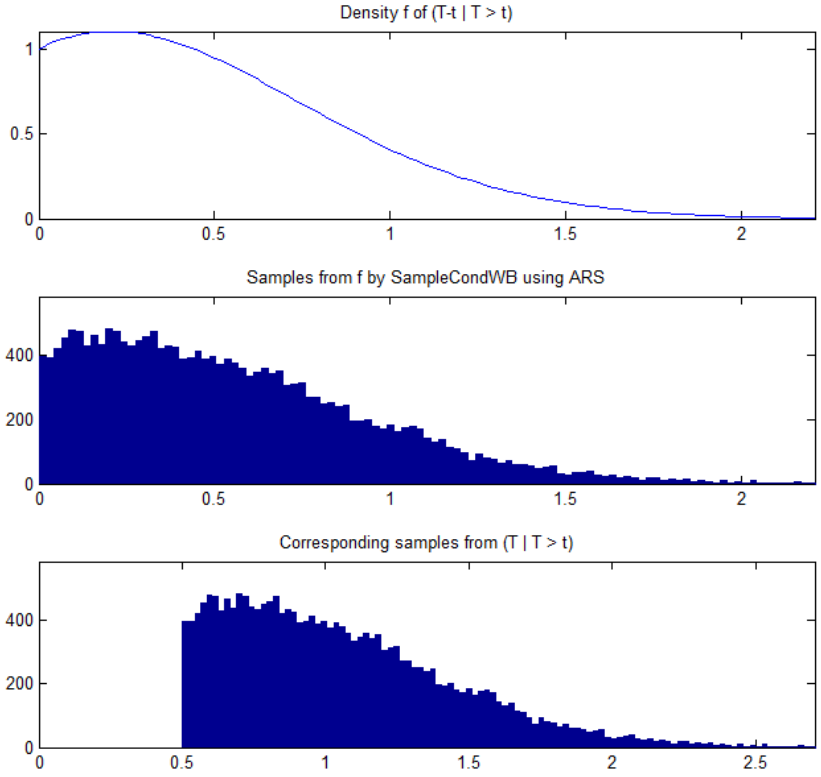
In summary, f_ξ is log-concave and can thus be simulated via ARS.

The M-file SampleCondWB.m implements the simulation scheme based on an existing ARS-implementation (ars.m) by [Eaton 2006]. See Appendix A.4 for details on SampleCondWB.m and the CD in the back of the thesis for the actual M-files SampleCondWB.m and ars.m.

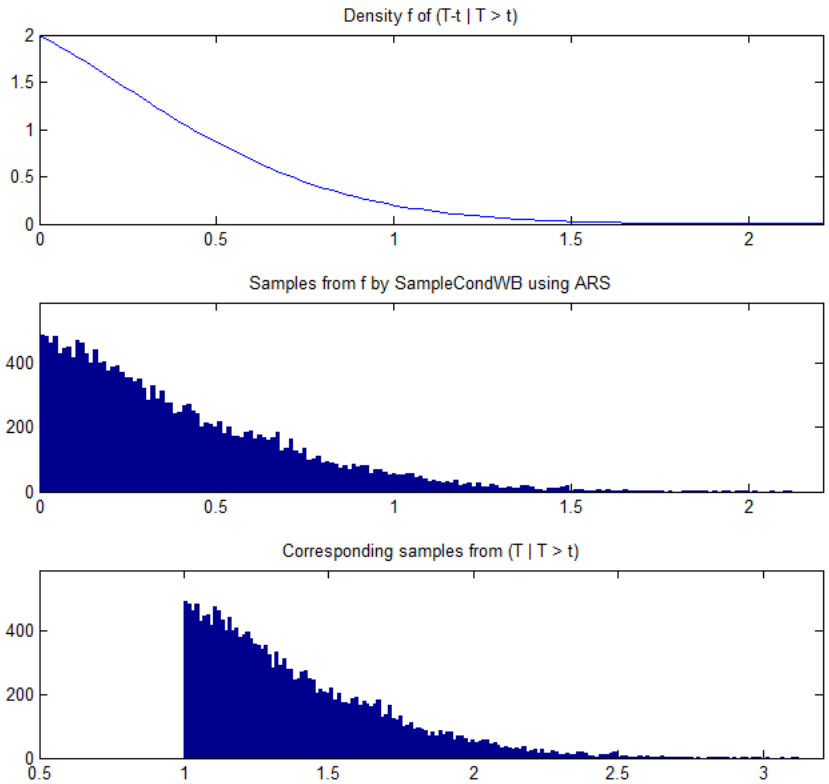
SampleCondWB.m

```
function [samples, samples2] = SampleCondWB(scale,shape,t,nSamples)

% Sampling from conditioned Weibull distributions (Appendix A.3)
% scale = scale parameter (lambda) > 0
% shape = shape parameter (k) >= 1
% t = condition time --> samples are from
%                          xi = WB(scale,shape)-t |  WB(scale,shape) > t
% nSamples = number of samples one wants to draw
% output: samples = samples from xi
%         samples2 = samples from WB(scale,shape) | WB(scale,shape) > t
%
% see SampleCondWBdemo.m on the CD

    if shape < 1
        error('shape parameter must be greater or equal than 1');
    end

    func = @(T,t,scale,shape) log(shape./scale) - (shape-1).*log(scale) +
          (t./scale).^shape + (shape-1).*log(T+t) - ((T+t)./scale).^shape;
    domain = [0 inf];
    a = domain(1);
    b = 1 + scale.*nthroot(1-(1./shape),shape);

    samples = ars(func, a, b, domain, nSamples, t, scale, shape);
    samples2 = samples + t;
end
```

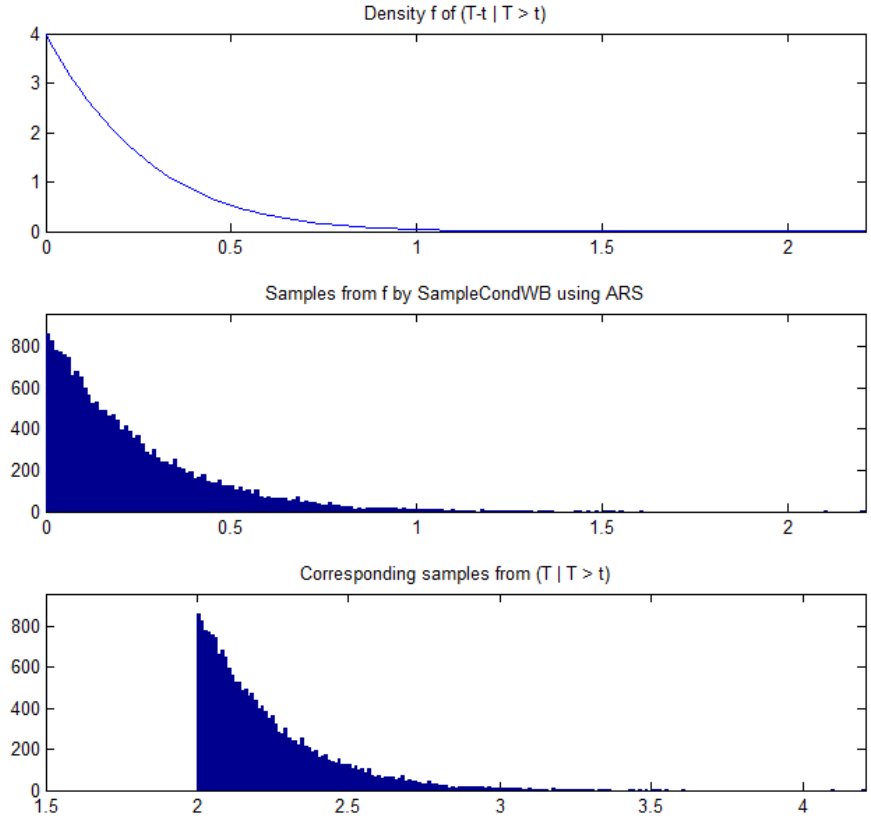
The following diagrams can be generated by SampleCondWBdemo.m (see CD) and together they exemplify the basic property of the Weibull distribution in contrast to the exponential distribution: In the case of Weibull waiting times and due to $k>1$ it becomes more and more probable that the described event happens when more and more time goes by. Formally, $\forall s,t\geq 0\colon\ \mathbb{P}\left(T>t+s\middle|T>t\right)<\mathbb{P}\left(T>s\right)$ (compare with Appendix A.1). The three (times three) diagrams show the density of f_ξ the samples from $\xi=\left(T-t\middle|T>t\right)$ and the corresponding samples from $T\middle|T>t$ for a Weibull variable $T\sim\mathcal{WB}(1,2)$ for $t=0.5,\ 1,\ 2$:


Density f of (T-t | T > t)
Samples from f by SampleCondWB using ARS
Corresponding samples from (T | T > t)

t = 0.5


Density f of (T-t | T > t)
Samples from f by SampleCondWB using ARS
Corresponding samples from (T | T > t)

t = 1



Intuitively spoken, the for our purposes decisive property of the Weibull distribution with shape parameter greater than one is the fact that the longer one already waits the more probable it is that the described event will happen soon. You can nicely see this property for the plotted examples by comparing the respective heights in the histograms.

The plots above can be produced by SampleCondWBdemo.m which can be found on the CD in the back of the thesis.

A.4 M-files for Subsections 4.2.1 and 4.2.3

The M-file ExampleOne.m can be used to simulate the model from Subsection 4.2.1:

```
function [N2, N6, tc, TimeEvol] = ExampleOne(Delay, Ncells, start, lambda,
shape)

% input: Delay = 'exp' --> Ncells = number of simulated chains/cells
%                          lambda = scale parameters of the respective
%                                   exp. dist., 1x6 vector:
%                                               [x_on,x_off,...,z_off]
%        Delay = 'WB' --> in addition specify the shape parameters of the
%                         Weibull dist., 1x6 vector:
%                                        [kx_on,kx_off,...,kz_off]

% output: N2: absorption count to steady state 2 (100)
%         N6:  -------------- " -------------- 6 (101)
%         tc: switching times (in absolute continuous time)
%         TimeEvol: times series of jump process according to tc

    if nargin == 4 || nargin == 5


    N2 = 0; N6 = 0;

    TimeEvol = zeros(Ncells,100);
    for i=1:Ncells
        TimeEvol(i,1) = start;
    end

    tc = zeros(Ncells,100);
    for i=1:Ncells
        tc(i,1) = 0;
    end

    len = ones(1,Ncells);

    if strcmp(Delay, 'exp') == 1

        for i=1:Ncells

            state = start;
            t = 1;

            while state ~= 2 && state ~= 6

                if state == 1
                    tau1 = exprnd(1/lambda(1));
                    tau2 = exprnd(1/lambda(3));
                    tau = min(tau1,tau2);
                    if tau == tau1
                        state = 2;
                        N2 = N2 + 1;
                    else
                        state = 3;
                    end
                    t = t + 1;
```

```
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    elseif state == 3
        tau1 = exprnd(1/lambda(1));
        tau2 = exprnd(1/lambda(5));
        tau = min(tau1,tau2);
        if tau == tau1
            state = 7;
        else
            state = 5;
        end
        t = t + 1;
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    elseif state == 4
        tau1 = exprnd(1/lambda(1));
        tau2 = exprnd(1/lambda(3));
        tau = min(tau1,tau2);
        if tau == tau1
            state = 6;
            N6 = N6 + 1;
        else
            state = 5;
        end
        t = t + 1;
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    elseif state == 5
        tau = exprnd(1/lambda(1));
        state = 8;
        t = t + 1;
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    elseif state == 7
        tau1 = exprnd(1/lambda(4));
        tau2 = exprnd(1/lambda(5));
        tau = min(tau1,tau2);
        if tau == tau1
            state = 2;
            N2 = N2 + 1;
        else
            state = 8;
        end
        t = t + 1;
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    else
        tau = exprnd(1/lambda(4));
        state = 6;
        N6 = N6 + 1;
        t = t + 1;
```

```
                tc(i,t) = tc(i,t-1) + tau;
                TimeEvol(i,t) = state;
                len(i) = len(i) + 1;
            end

        end

    end

end

if strcmp(Delay, 'WB') == 1

    for i=1:Ncells

        state = start;
        t = 1;
        delta = 0;
        deltaStar = 0;
        kap = 0;

        while state ~= 2 && state ~= 6

            if state == 1
                tau1 = SampleCondWB(lambda(1),shape(1),delta,1);
                tau2 = SampleCondWB(lambda(3),shape(3),0,1);
                tau = min(tau1,tau2);
                if tau == tau1
                    state = 2;
                    N2 = N2 + 1;
                else
                    state = 3;
                end
                t = t + 1;
                delta = tau;
                tc(i,t) = tc(i,t-1) + tau;
                TimeEvol(i,t) = state;
                len(i) = len(i) + 1;

            elseif state == 3
                tau1 = SampleCondWB(lambda(1),shape(1),delta,1);
                tau2 = SampleCondWB(lambda(5),shape(5),0,1);
                tau = min(tau1,tau2);
                if tau == tau1
                    state = 7;
                else
                    state = 5;
                    kap = 2;
                end
                t = t + 1;
                delta = tau;
                deltaStar = tc(i,t-1) - tc(i,t-2);
                tc(i,t) = tc(i,t-1) + tau;
                TimeEvol(i,t) = state;
                len(i) = len(i) + 1;

            elseif state == 4
                tau1 = SampleCondWB(lambda(1),shape(1),0,1);
                tau2 = SampleCondWB(lambda(3),shape(3),0,1);
                tau = min(tau1,tau2);
```

```
            if tau == tau1
                state = 7;
                N7 = N7 + 1;
            else
                state = 5;
                kap = 1;
            end
            t = t + 1;
            delta = tau;
            tc(i,t) = tc(i,t-1) + tau;
            TimeEvol(i,t) = state;
            len(i) = len(i) + 1;

        elseif state == 5
            if kap == 1
                tau = SampleCondWB(lambda(1),shape(1),delta,1);
            else
                tau = SampleCondWB(lambda(1),shape(1),deltaStar,1);
            end
            state = 8;
            kap = 0;
            t = t + 1;
            delta = tau;
            tc(i,t) = tc(i,t-1) + tau;
            TimeEvol(i,t) = state;
            len(i) = len(i) + 1;

        elseif state == 7
            tau1 = SampleCondWB(lambda(4),shape(4),0,1);
            tau2 = SampleCondWB(lambda(5),shape(5),delta,1);
            tau = min(tau1,tau2);
            if tau == tau1
                state = 2;
                N2 = N2 + 1;
            else
                state = 8;
                kap = 1;
            end
            t = t + 1;
            delta = tau;
            tc(i,t) = tc(i,t-1) + tau;
            TimeEvol(i,t) = state;
            len(i) = len(i) + 1;

        else
            if kap == 1
                tau = SampleCondWB(lambda(4),shape(4),delta,1);
            else
                tau = SampleCondWB(lambda(2),shape(2),0,1);
            end
            state = 6;
            N6 = N6 + 1;
            t = t + 1;
            delta = tau;
            tc(i,t) = tc(i,t-1) + tau;
            TimeEvol(i,t) = state;
            len(i) = len(i) + 1;
        end

    end
```

```
        end

    end



    tcCell = cell(Ncells,1);
    for i=1:Ncells
        tcCell{i,1} = tc(i,1:len(i));
    end
    tc = tcCell;

    TimeEvolCell = cell(Ncells,1);
    for i=1:Ncells
        TimeEvolCell{i,1} = TimeEvol(i,1:len(i));
    end
    TimeEvol = TimeEvolCell;

    end

end
```

The M-file EstimExampleOne.m can be used to estimate the parameters based on simulated data:

```
function estimator = EstimExampleOne(Ncells, scale)

% estimation of Example 1 according to Subsection 4.2.1
% relies on EstimExampleOneSub.m, see CD

% input: Ncells = number of simulated chains/cells
%        scale = true chosen scale parameter vector
%                1x6 vector = [x_on,x_off,...,z_off]
%
% output: estimator = estimator of scale

        f2 = zeros(1,8);
        f6 = zeros(1,8);

        for i=[1,3,4,5,7,8]
            N2 = ExampleOne('exp',Ncells,i,scale,0);
            f2(i) = N2/Ncells;
            f6(i) = 1-f2(i);
        end

        estimator = EstimExampleOneSub(f2,f6,scale(1));
        estimator =
[estimator(1),scale(2),estimator(2),estimator(3),estimator(4),scale(6)];

end
```

Concerning Subsection 4.2.3 the M-file ExampleTwo.m can be used to simulate the network for both model types:

```
function [N4, N7, tc, TimeEvol] = ExampleTwo(Delay, Ncells, start, lambda,
shape)

% input: Delay = 'exp' --> Ncells = number of simulated chains/cells
%                          lambda = scale parameters of the respective
%                                   exp. dist., 1x6 vector:
%                                               [x_on,x_off,...,z_off]
%        Delay = 'WB' --> in addition specify the shape parameters of the
%                         Weibull dist., 1x6 vector:
%                                        [kx_on,kx_off,...,kz_off]

% output: N4: absorption count to steady state 4 (001)
%         N7:  -------------- " -------------- 7 (110)
%         tc: switching times (in absolute continuous time)
%         TimeEvol: times series of jump process according to tc

if nargin == 4 || nargin == 5


    N4 = 0; N7 = 0;

    TimeEvol = zeros(Ncells,100);
    for i=1:Ncells
        TimeEvol(i,1) = start;
    end

    tc = zeros(Ncells,100);
    for i=1:Ncells
        tc(i,1) = 0;
    end

    len = ones(1,Ncells);

    if strcmp(Delay, 'exp') == 1

        for i=1:Ncells

            state = start;
            t = 1;

            while state ~= 4 && state ~= 7

                if state == 1
                    tau1 = exprnd(1/lambda(1));
                    tau2 = exprnd(1/lambda(5));
                    tau = min(tau1,tau2);
                    if tau == tau1
                        state = 2;
                    else
                        state = 4;
                        N4 = N4 + 1;
                    end
                    t = t + 1;
                    tc(i,t) = tc(i,t-1) + tau;
                    TimeEvol(i,t) = state;
                    len(i) = len(i) + 1;
```

```
elseif state == 2
    tau1 = exprnd(1/lambda(3));
    tau2 = exprnd(1/lambda(5));
    tau = min(tau1,tau2);
    if tau == tau1
        state = 7;
        N7 = N7 + 1;
    else
        state = 6;
    end
    t = t + 1;
    tc(i,t) = tc(i,t-1) + tau;
    TimeEvol(i,t) = state;
    len(i) = len(i) + 1;

elseif state == 3
    tau1 = exprnd(1/lambda(1));
    tau2 = exprnd(1/lambda(4));
    tau = min(tau1,tau2);
    if tau == tau1
        state = 7;
        N7 = N7 + 1;
    else
        state = 1;
    end
    t = t + 1;
    tc(i,t) = tc(i,t-1) + tau;
    TimeEvol(i,t) = state;
    len(i) = len(i) + 1;

elseif state == 5
    tau1 = exprnd(1/lambda(4));
    tau2 = exprnd(1/lambda(6));
    tau = min(tau1,tau2);
    if tau == tau1
        state = 4;
        N4 = N4 + 1;
    else
        state = 3;
    end
    t = t + 1;
    tc(i,t) = tc(i,t-1) + tau;
    TimeEvol(i,t) = state;
    len(i) = len(i) + 1;

elseif state == 6
    tau1 = exprnd(1/lambda(2));
    tau2 = exprnd(1/lambda(3));
    tau = min(tau1,tau2);
    if tau == tau1
        state = 4;
        N4 = N4 + 1;
    else
        state = 8;
    end
    t = t + 1;
    tc(i,t) = tc(i,t-1) + tau;
    TimeEvol(i,t) = state;
    len(i) = len(i) + 1;
```

```
            else
                tau1 = exprnd(1/lambda(2));
                tau2 = exprnd(1/lambda(6));
                tau = min(tau1,tau2);
                if tau == tau1
                    state = 5;
                else
                    state = 7;
                    N7 = N7 + 1;
                end
                t = t + 1;
                tc(i,t) = tc(i,t-1) + tau;
                TimeEvol(i,t) = state;
                len(i) = len(i) + 1;
            end

        end

    end

end

if strcmp(Delay, 'WB') == 1

    for i=1:Ncells

        state = start;
        t = 1;
        delta = 0;

        while state ~= 4 && state ~= 7

            if state == 1
                tau1 = SampleCondWB(lambda(1),shape(1),delta,1);
                tau2 = SampleCondWB(lambda(5),shape(5),0,1);
                tau = min(tau1,tau2);
                if tau == tau1
                    state = 2;
                else
                    state = 4;
                    N4 = N4 + 1;
                end
                t = t + 1;
                delta = tau;
                tc(i,t) = tc(i,t-1) + tau;
                TimeEvol(i,t) = state;
                len(i) = len(i) + 1;

            elseif state == 2
                tau1 = SampleCondWB(lambda(3),shape(3),0,1);
                tau2 = SampleCondWB(lambda(5),shape(5),delta,1);
                tau = min(tau1,tau2);
                if tau == tau1
                    state = 7;
                    N7 = N7 + 1;
                else
                    state = 6;
                end
                t = t + 1;
                delta = tau;
```

```
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    elseif state == 3
        tau1 = SampleCondWB(lambda(1),shape(1),0,1);
        tau2 = SampleCondWB(lambda(4),shape(4),delta,1);
        tau = min(tau1,tau2);
        if tau == tau1
            state = 7;
            N7 = N7 + 1;
        else
            state = 1;
        end
        t = t + 1;
        delta = tau;
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    elseif state == 5
        tau1 = SampleCondWB(lambda(4),shape(4),0,1);
        tau2 = SampleCondWB(lambda(6),shape(6),delta,1);
        tau = min(tau1,tau2);
        if tau == tau1
            state = 4;
            N4 = N4 + 1;
        else
            state = 3;
        end
        t = t + 1;
        delta = tau;
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    elseif state == 6
        tau1 = SampleCondWB(lambda(2),shape(2),0,1);
        tau2 = SampleCondWB(lambda(3),shape(3),delta,1);
        tau = min(tau1,tau2);
        if tau == tau1
            state = 4;
            N4 = N4 + 1;
        else
            state = 8;
        end
        t = t + 1;
        delta = tau;
        tc(i,t) = tc(i,t-1) + tau;
        TimeEvol(i,t) = state;
        len(i) = len(i) + 1;

    else
        tau1 = SampleCondWB(lambda(2),shape(2),delta,1);
        tau2 = SampleCondWB(lambda(6),shape(6),0,1);
        tau = min(tau1,tau2);
        if tau == tau1
            state = 5;
        else
            state = 7;
            N7 = N7 + 1;
```

```
                    end
                    t = t + 1;
                    delta = tau;
                    tc(i,t) = tc(i,t-1) + tau;
                    TimeEvol(i,t) = state;
                    len(i) = len(i) + 1;
                end

            end

        end

    end

    tcCell = cell(Ncells,1);
    for i=1:Ncells
        tcCell{i,1} = tc(i,1:len(i));
    end
    tc = tcCell;

    TimeEvolCell = cell(Ncells,1);
    for i=1:Ncells
        TimeEvolCell{i,1} = TimeEvol(i,1:len(i));
    end
    TimeEvol = TimeEvolCell;

end

end
```

The estimation procedure described in Subsection 4.2.3 can be conducted for simulated data via the M-file EstimExampleTwo.m:

```
function estimator = EstimExampleTwo(Ncells, scale)

% input: Ncells = number of simulated chains/cells
%        scale = true chosen scale parameter vector
%                1x6 vector = [x_on,x_off,...,z_off]
%
% output: estimator = estimator of scale

        f4 = zeros(1,6);
        f7 = zeros(1,6);

        for i=[1,2,3,5,6,8]
            N4 = ExampleTwo('exp',Ncells,i,scale,0);
            f4(i) = N4/Ncells;
            f7(i) = 1-f4(i);
        end

        estimator = EstimExampleTwoSub(f4,f7,scale(1));

end
```

A.5 R scripts for Subsection 5.2.2

This appendix documents the R-scripts referred to in Section 5.2.

PLSDE_ExampleOne.R

```
# Naive SPADE simulation of simple circuit from [Thomas 2013]
# needs rgl package

# input

# gam: degradation rates, a 3-dimensional vector
# beta: production rates, a 3-dimensional vector
# 0 <= thresi <= 1, i=1,2,3: thresholds corredponding to delta_i (see main text)
# delta > 0: step size
# t_max > 0: maximal simulation time
# start: initial concentrations, a 3-dimensional vector
# noise: noise strength, a 3-dimensional vecotr (noise[i] = noise for species i)

# ouput

# plot of concentration time coursees,
# species 1 = red, species 2 = green, species 3 = blue

# 3D portrait of the trajectory in state space

# X_i: conentration values of species i over time, i=1,2,3
# time: time
# logic: logical trajectory given by a sequences of domains
#        (3-dimensional 01-  vectors)



PLSDE_ExampleOne <-
function(gam=c(1,1,1),beta=c(2,2,2),thres1=.625,thres2=.5,thres3=.5,delta=.001,
         t_max=10,start=c(0,0,0),noise=c(.02,.02,.02)){

       Y <- start                          # initial value

       X_1 <- c(); X_1[1] <- Y[1]
       X_2 <- c(); X_2[1] <- Y[2]
       X_3 <- c(); X_3[1] <- Y[3]

       theta <- c()
       theta[1] <- thres1*(beta[1]/gam[1])    # definition of continuous thresholds
       theta[2] <- thres2*(beta[2]/gam[1])
       theta[3] <- thres3*(beta[3]/gam[1])

       g_1 <- c(); g_2 <- c(); g_3 <- c()     # logical states
       g_1[1] <- 1*(Y[1] >= theta[1])
       g_2[1] <- 1*(Y[2] >= theta[2])
       g_3[1] <- 1*(Y[3] >= theta[3])

  logic <- matrix(,nrow=length(seq(0,t_max,by=delta)),ncol=3)
  # logical trajectory
       logic[1,1] <- g_1[1]
       logic[1,2] <- g_2[1]
       logic[1,3] <- g_3[1]

       p <- c()                    # production rates
       p[1] <- beta[1]*1
       p[2] <- beta[2]*(1-g_1[1])
       p[3] <- beta[3]*(g_2[1]|g_3[1])
```

```
diff <- (sqrt(p[1]+gam[1]*Y[1]),sqrt(p[2]+gam[2]*Y[2]),sqrt(p[3]+gam[3]*Y[3]))
diff <- diag(diff)                                  # diffusion matrix

     z <- 2

     for (t in seq(delta,t_max,by=delta)){

            for (i in 1:3){
                  b <- rnorm(1,0,1)
                  Y[i] <- Y[i] + delta*(p[i] - gam[i]*Y[i]) +
                          diff[i,i]*(sqrt(delta*noise[i])*b)
            # Euler-Maruyama iteration
                  }

            for(i in 1:3){
                  if(Y[i] <= 0){Y[i] <- 0}
                  }

            X_1[z] <- Y[1]
            X_2[z] <- Y[2]
            X_3[z] <- Y[3]

            g_1[z] <- 1*(Y[1] >= theta[1])
            g_2[z] <- 1*(Y[2] >= theta[2])
            g_3[z] <- 1*(Y[3] >= theta[3])

            logic[z,1] <- g_1[z]
            logic[z,2] <- g_2[z]
            logic[z,3] <- g_3[z]

            p[1] <- beta[1]*1
            p[2] <- beta[2]*(1-g_1[z])
            p[3] <- beta[3]*(g_2[z]|g_3[z])

  diff <- c(sqrt(p[1]+gam[1]*Y[1]),sqrt(p[2]+gam[2]*Y[2]),sqrt(p[3]+gam[3]*Y[3]))
  diff <- diag(diff)                    # update of diffusion matrix

            z <- z + 1
            }



     time <- seq(0,t_max,by=delta)

     plot(time,X_1,type="l",col="red",
      ylim=c(0,.5 + max(beta[1]/gam[1],beta[2]/gam[2],beta[3]/gam[3])),
      ylab="X_1(red), X_2(green), X_3(blue)",lwd=1.5)

     lines(time,X_2,type="l",col="green",lwd=2)
     lines(time,X_3,type="l",col="blue",lwd=2)

     library(rgl)

     plot3d(X_1,X_2,X_3,type="l",col="orange",lwd=2)

     #plot(time,g_1,type="l")
     #plot(time,g_2,type="l")
     #plot(time,g_3,type="l")

     #plot(time,logic[,1],type="l")
     #plot(time,logic[,2],type="l")
     #plot(time,logic[,3],type="l")

     out <- list(X_1,X_2,X_3,time,logic)

     return(out)
      }
```

PLSDE_Repressilator.R

```
# Naive SPADE simulation of the repressilator
# needs rgl package

# input

# gam: degradation rates, a 3-dimensional vector
# beta: production rates, a 3-dimensional vector
# 0 <= thresi <= 1, i=1,2,3: thresholds corredponding to delta_i (see main text)
# delta > 0: step size
# t_max > 0: maximal simulation time
# start: initial concentrations, a 3-dimensional vector
# noise: noise strength, a 3-dimensional vecotr (noise[i] = noise for species i)

# ouput

# plot of concentration time coursees,
# species 1 = red, species 2 = green, species 3 = blue

# 3D portrait of the trajectory in state space

# X_i: conentration values of species i over time, i=1,2,3
# time: time
# logic: logical trajectory given by a sequences of domains
# (3-dimensional 01-vectors)

PLSDE_Repressilator <-
function(gam=c(1,1,1),beta=c(2,2,2),thres1=.5,thres2=.5,thres3=.5,delta=.001,
         t_max=10,start=c(0,0,0),noise=c(.02,.02,.02)){

       Y <- start                          # initial value

       X_1 <- c(); X_1[1] <- Y[1]
       X_2 <- c(); X_2[1] <- Y[2]
       X_3 <- c(); X_3[1] <- Y[3]

       theta <- c()
       theta[1] <- thres1*(beta[1]/gam[1])    # definition of continuous thresholds
       theta[2] <- thres2*(beta[2]/gam[1])
       theta[3] <- thres3*(beta[3]/gam[1])

       g_1 <- c(); g_2 <- c(); g_3 <- c()     # logical states
       g_1[1] <- 1*(Y[1] >= theta[1])
       g_2[1] <- 1*(Y[2] >= theta[2])
       g_3[1] <- 1*(Y[3] >= theta[3])

  logic <- matrix(,nrow=length(seq(0,t_max,by=delta)),ncol=3)
  # logical trajector
       logic[1,1] <- g_1[1]
       logic[1,2] <- g_2[1]
       logic[1,3] <- g_3[1]

       p <- c()                           # production rates
       p[1] <- beta[1]*(1-g_3[1])
       p[2] <- beta[2]*(1-g_1[1])
       p[3] <- beta[3]*(1-g_2[1])

  diff <- c(sqrt(p[1]+gam[1]*Y[1]),sqrt(p[2]+gam[2]*Y[2]),sqrt(p[3]+gam[3]*Y[3]))
  diff <- diag(diff)                                        # diffusion matrix
```

```
      z <- 2               # time counter

      for (t in seq(delta,t_max,by=delta)){

            for (i in 1:3){
                  b <- rnorm(1,0,1)
                  Y[i] <- Y[i] + delta*(p[i] - gam[i]*Y[i]) +
                        diff[i,i]*(sqrt(delta*noise[i])*b)
            # Euler-Maruyama iteration
                  }

            for(i in 1:3){
                  if(Y[i] <= 0){Y[i] <- 0}
                  }

            X_1[z] <- Y[1]
            X_2[z] <- Y[2]
            X_3[z] <- Y[3]

            g_1[z] <- 1*(Y[1] >= theta[1])
            g_2[z] <- 1*(Y[2] >= theta[2])
            g_3[z] <- 1*(Y[3] >= theta[3])

            logic[z,1] <- g_1[z]
            logic[z,2] <- g_2[z]
            logic[z,3] <- g_3[z]

            p[1] <- beta[1]*(1-g_3[z])
            p[2] <- beta[2]*(1-g_1[z])
            p[3] <- beta[3]*(1-g_2[z])

  diff <- c(sqrt(p[1]+gam[1]*Y[1]),sqrt(p[2]+gam[2]*Y[2]),sqrt(p[3]+gam[3]*Y[3]))
  diff <- diag(diff)                        # update of diffusion matrix

            z <- z + 1
            }

      time <- seq(0,t_max,by=delta)

#plot(time,X_1,type="l",col="red",ylim=c(0,1.5),ylab="X_1(red), X_2(green),
      X_3(blue)",lwd=1.5)

      plot(time,X_1,type="l",col="red",
            ylim=c(0,.5 + max(beta[1]/gam[1],beta[2]/gam[2],beta[3]/gam[3])),
            ylab="X_1(red), X_2(green), X_3(blue)",lwd=1.5)
      lines(time,X_2,type="l",col="green",lwd=2)
      lines(time,X_3,type="l",col="blue",lwd=2)

      library(rgl)

      plot3d(X_1,X_2,X_3,type="l",col="orange",lwd=2)

      #plot(time,g_1,type="l")
      #plot(time,g_2,type="l")
      #plot(time,g_3,type="l")

      #plot(time,logic[,1],type="l")
      #plot(time,logic[,2],type="l")
      #plot(time,logic[,3],type="l")

      out <- list(X_1,X_2,X_3,time,logic)

      return(out)

      }
```

B Bibliography

[Abou-Jaoudé et al. 2009] W. Abou-Jaoudé, D.A. Ouattara, M. Kaufman: *From structure to dynamics: Frequency tuning in the p53-Mdm2 network I. Logical approach.* Journal of Theoretical Biology, 2009, 258, 561-577

[Ahmad et al. 2007] J. Ahmad, G. Bernot, J.-P. Comet, D. Lime, O. Roux: *Hybrid Modelling and Dynamical Analysis of Gene Regulatory Networks with Delays.* ComPlexUs, 2007, 3, 231-251

[Aigner 2006] M. Aigner: *Diskrete Mathematik.* 2006, 6. Aufl., Vieweg + Teubner

[Aittokallio, Schwikowski 2006] T. Aittokallio, V. Schwikowski: *Graph-based methods for analysing networks in cell biology.* Briefings in Bioinformatics, 2006, 7, 243-255

[Albert, Wang 2009] R. Albert, R.-S. Wang: *Discrete dynamic modeling of cellular signaling networks.* Methods in Enzymology, 2009, 467, 281-306

[Albert et al. 2013] R. Albert, J.J. Collins, L. Glass: *Introduction to Focus Issue: Quantitative approaches to genetic networks.* Chaos, 2013, 23, 025001

[Aldana et al. 2003] M. Aldana, S. Coppersmith, L.P. Kadanoff: *Boolean dynamics with random couplings*. Perspectives and Problems in Nonlinear Science, 2003, 23-89

[Aldridge et al. 2009] B.B. Aldridge, J. Saez-Rodriguez, J.L. Muhlich, P.K. Sorger, D.A. Lauffenburger: *Fuzzy Logic Analysis of Kinase Pathway Crosstalk in TNF/EGF/Insulin-Induced Signaling.* PLoS Computational Biology, 5 (4) e1000340

[Alla, David 2005] H. Alla, R. David: Discrete, *Continuous and Hybrid Petri nets.* 2005, 1st ed., Springer

[Alon 2007] U. Alon: *An introduction to systems biology: Design principles of biological circuits*. 2007, 1st ed., Chapman & Hall/CRC

[Alvarez-Buylla et al. 2008] E.R. Alvarez-Buylla, A. Chaos, M. Aldana, M. Benitez, Y. Cortes-Poza, C. Espinosa-Soto, D.A. Hartasánchez, R.B. Lotto, D. Malkin, G.J. Escalera Santos, P. Padilla-Longoria. *Floral Morphogenesis: Stochastic Explorations of a Gene Network Epigenetic Landscape.* PLoS One, 2008, 3 (11) e3626

[Aracena et al. 2009] J. Aracena, E. Goles, A. Moreira, L. Salinas: *On the robustness of update schedules in Boolean networks.* BioSystems, 2009, 97, 1-8

[Arkin et al. 1998] A. Arkin, J. Ross, H.H. McAdams: *Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected Escherichia coli cells.* Genetics, 1998, 149, 1633-1648

[Ashyraliyev et al. 2009] M. Ashyraliyev, Y. Fomekong-Nanfack, J.A. Kaandorp, J.G. Blom: *Systems biology: parameter estimation for biochemical models.* FEBS Journal, 2009, 276, 886-902

[Bajikar et al. 2014] S.S. Bajikar, C. Fuchs, A. Roller, F.J. Theis, K.A. Janes: *Parameterizing cell-to-cell regulatory heterogeneities via stochastic transcriptional profiles.* PNAS, 2014, Published online before print January 21 2014, doi: 10.1073

[Barkai, Shilo 2007] N. Barkai, B.-Z. Shilo: *Variability and Robustness in Biomolecular Systems.* Molecular Cell, 2007, 28, 755-760

[Bause 2002] F. Bause: *Stochastic Petri nets.* 2002, 2nd ed., Vieweg + Teubner

[Bernot et al. 2004] G. Bernot, J.-P. Comet, A. Richard, J. Guespin: *Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic.* Journal of Theoretical Biology, 2004, 229, 339-347

[Bintu et al. 2005a] L. Bintu, N.E. Buchler, H.G. Garcia, U. Gerland, T. Hwa, J. Kondev, R. Philips: *Transcriptional regulation by the numbers: models.* Current Opinions in Genetics and Development, 2005, 15 (2) 116-124

[Bintu et al. 2005b] L. Bintu, N.E. Buchler, H.G. Garcia, U. Gerland, T. Hwa, J. Kondev, T. Kuhlman, R. Philips: *Transcriptional regulation by the numbers: applications.* Current Opinions in Genetics and Development, 2005, 15 (2) 125-135

[Bishop 2006] C.M. Bishop: *Pattern Recognition and Machine Learning.* 2006, 1st ed., Springer

[Blom, Lygeros (eds.) 2006] H.A.P. Blom, J. Lygeros: *Stochastic Hybrid Systems: Theory and Safety Critical Applications.* 2006, 1st ed., Springer

[Bujorianu, Lygeros 2008] M.L. Bojurianu, J. Lygeros: *Toward a General Theory of Stochastic Hybrid Systems.* Chapter 1 in [Blom, Lygeros (eds.) 2006]

[Bolouri 2008] H. Bolouri: *Compuational Modeling of Gene Regulatory Networks. A Primer.* 2008, 1st ed., World Scientific

[Bornholdt 2008] S. Bornholdt: *Boolean network models of cellular regulation: prospects and limitations.* Journal of the Royal Society Interface, 2008, 5, S85-S94

[Boys et al. 2008] R.J. Boys, D.J. Wilkinson, T.B.L. Kirkwood: *Bayesian inference for a discretely observed stochastic kinetic model.* Statistical Computing, 2008, 18, 125-135

[Bulyk, Walhout 2013] M.L. Bulyk, A.J.M. Walhout: *Gene Regulatory Networks.* Chapter 4 in [Walhout et al. (ed.) 2013]

[Casey et al. 2006] R. Casey, H. de Jong, J.-L. Gouzé: *Piecewise-linear models of genetic regulatory networks: Equilibria and their stability.* Journal of Mathematical Biology, 2006, 52 (1) 27-56

[Cassandras, Lygeros (eds.) 2007] C.G. Cassandras, J. Lygeros: *Stochastic Hybrid Systems.* 2007, 1st ed., CRC Taylor & Francis

[CellSignalingTechnology 2014]: http://www.cellsignal.com/common/content/content.jsp?id=science-pathways [22.08.2014]

[CellSignalingTechnology NF-κB 2014]: http://www.cellsignal.com/common/content/content.jsp?id=pathways-nfkb [22.08.2014]

[CellSignalingTechnology Warburg 2014]: http://www.cellsignal.com/common/content/content.jsp?id=pathways-warburg [23.08.2014]

[Chaouiya 2007] C. Chaouiya: *Petri net modeling of biological networks.* Briefings in Bioinformatics, 8 (4) 210-219

[Chaouyia et al. 2008] C. Chaouiya, E. Remy, D. Thieffry: *Petri net modeling of biological regulatory networks.* Journal of Discrete Algorithms, 2008, 6, 165-177

[Chaouyia, Remy 2013] C. Chaouyia, E. Remy: *Logical Modelling of Regulatory Networks, Methods and Applications.* Bulletin of Mathematical Biology, 2013, 75, 891-895

[Chaves et al. 2005] M. Chaves, R. Albert, E. Sontag: *Robustness and fragility of Boolean models for genetic regulatory networks.* Journal of Theoretical Biology, 2005, 235, 431-449

[Chaves et al. 2009] M. Chaves, T. Eissing, F. Allgöwer: *Regulation of apoptosis via the NFκB pathway: Modeling and Analysis.* In: N. Ganuly, A. Deutsch, A. Mukherjee (eds.): *Dynamics on and of complex networks. Applications to biology, computer science, and the social sciences.* 2009, Birkhäuser

[Chaves et al. 2010] M. Chaves, L. Tournier, J.-L. Gouzé: *Comparing Boolean and Piecewise Affine Differential Models for Genetic Networks.* Acta Biotheoretica, 2010, 58, 217-232

[Chaves, Gouzé 2010] M. Chaves, J.-L. Gouzé: *Piecewise affine models of regulatory genetic networks: review and probabilistic interpretation.* pp. 241-253 in *Advances in the Theory of Control, Signals and Systems, with Physical Modelling.* (L. Lévine, P. Müllhaupt (eds.))

[Chaves et al. 2013] M. Chaves, E. Farcot, J.-L. Gouzé: *Probabilistic Approach for Predicting Periodic Orbits in Piecewise Affine Differential Models.* Bulletin of Mathematical Biology, 2013, 75, 967-987

[Cinquemani et al. 2007] E. Cinquemani, R. Porreca, G. Ferrari-Trecate, J. Lygeros: *Subtilin Production by Bacillus Subtilis: Stochastic Hybrid Models and Parameter Identification.* IEEE Transactions On Automatic Control, 2007, 53, 38-50

[Conrad, Tyson 2006] E.D. Conrad, J.J. Tyson: *Modeling Molecular Interaction Networks with Nonlinear Ordinary Differential Equations.* Chapter 6 in [Szallasi et al. (ed.) 2006]

[Cornish-Bowden 2012] A. Cornish-Bowden: *Fundamental of Enzyme Kinetics.* 2012, 4th ed., Wiley-VHC

[Covert et al. 2008] M.W. Covert, N.Xiao, T.J. Chen, J.R. Karr: *Integrating metabolic, transcriptional regulatory and signal transduction models in Escherichia coli.* Bioinformatics, 2008, 24 (18) 2044-2050

[Crudu et al. 2009] A. Crudu, A. Debussche, O. Radulesu: *Hybrid stochastic simplifications for multiscale gene networks.* BMC Systems Biology, 2009, 3:89

[Crudu et al. 2011] A. Crudu, A. Debussche, A. Muller, O. Radulescu: *Convergence of stochastic gene networks to hybrid piecewise deterministic processes.* arXiv:1101.1431

[Czado, Schmidt 2011] C. Czado, T. Schmidt: *Mathematische Statistik.* 2011, 1st ed., Springer

[Danial, Korsmeyer 2004] N.N. Danial, S.J. Korsmeyer: *Cell Death: Critical Control Points.* Cell, 2004, 116, 205-219

[Davidich, Bornholdt 2008a] M. Davidich, S. Bornholdt: *Boolean network model predicts cell cycle sequence of fission yeast.* PLoS One, 2008, 3 (2) e1672

[Davidich, Bornholdt 2008b] M. Davidich, S. Bornholdt: *The transition from differential equations to Boolean networks: A case study in simplifying a regulatory network model.* Journal of Theoretical Biology, 2008, 255, 269-277

[Davidson 2006] E.H. Davidson: *The Regulatory Genome. Gene Regulatory Networks in Development and Evolution.* 2006, 1st ed., Academic Press

[Davis 1993] M.H.A. Davis: *Markov processes and optimization.* 1993, 1st ed., Springer

[Dee, Ghil 1984] D. Dee, M. Ghil: *Boolean difference equations, I: formulation and dynamics behavior.* SIAM Journal of Applied Mathematics, 1984, 44 (1) 111-126

[Dee, Mullhaupt 1985] D. Dee, A. Mullhaupt: *Boolean delay equations. II. Periodic and aperiodic solutions.* Journal of Statistical Physics, 1985, 41 (1,2) 125-173

[Deevlo et al. 2003] V. Deevlo, P. Hansen, M. Labbé: *Identification of all steady states in large networks by logical analysis.* Bulletin of Mathematical Biology, 2003, 65, 1025-1051

[Dekker, van Steensel 2013] J. Dekker, B. van Steensel: *The spatial architecture of chromosomes.* Chapter 7 in [Walhout et al. (eds) 2013]

[De Jong 2002] H. de Jong: *Modeling and simulation of genetic regulatory networks: A literature review.* Journal of Computational Biology, 2002, 9 (1) 67-103

[De Jong et al. 2004a] H. de Jong, J.-L. Gouzé, C. Hernandez, M. Page, T. Sari, J. Geiselmann: *Qualitative Simulation of Genetic Regulatory Networks Using Piecewise-Linear Models.* Bulletin of Mathematical Biology, 2004, 66, 301-340

[De Jong et al. 2004b] H. de Jong, J. Geiselmann, G. Batt, C. Hernandez, M. Page: *Qualitative Simulation of the Initiation of Sporulation in Bacillus subtilis.* Bulletin of Mathematical Biology, 2004, 66, 261-299

[Deng et al. 2007] X. Deng, H. Geng, M.T. Matache: *Dynamics of asynchronous random Boolean networks with asynchrony generated by stochastic processes.* Biosystems, 2007, 88, 16-34

[Deuflhard, Bornemann 2008] P. Deuflhard, F. Bornemann: *Numerische Mathematik 2: Gewöhnliche Differentialgleichungen.* 2008, 3. Aufl., de Gruyter

[Di Cara et al. 2007] A. Di Cara, A. Garg, B. De Micheli, I. Xenarios, L. Mendoza: D*ynamic simulation of regulatory networks using SQUAD.* BMC Bioinformatics, 2007, 8:462

[Dong, Golden 2008] G. Dong, S.S. Golden: *How a cyanobacterium tells time.* Current Opinion in Microbiology, 2008, 11 541-546

[Dümcke et al. 2014] S. Dümcke, J. Bräuer, B. Anchang, R. Spang, N. Beerenwinkel, A. Tresch: *Exact likelihood computation in Boolean networks with probabilistic time delays, and ist application in signal network reconstruction.* Bioinformatics, 2014, 30 (3) 414-419

[Eaton 2006] D. Eaton: *ars.m.* 2006, pmtk3 https://code.google.com/p/pmtk3/source/browse/trunk/toolbox/Algorithms/mcmc/ars.m?r=2678

[Edelstein-Keshet 2005] L. Edelstein-Keshet: *Mathematical Models in Biology.* 2005, 1st ed., SIAM

[Edwards 2000] R. Edwards: *Analysis of continuous-time switching networks.* Physica D, 2000, 146 (1-4) 165-199

[Edwards et al. 2001] R. Edwards, H.T. Siegelmann, K. Aziza, L. Glass: *Symbolic dynamics and computation in model gene networks.* Chaos, 2001, 11 (1) 160-169

[Elowitz, Leibler 2000] M.B. Elowitz, S. Leibler: *A synthetic oscillatory network of transcriptional regulators.* Nature, 2000, 403 (20) 335-338

[Elowitz et al. 2002] M.B. Elowitz, A.J. Levine, E.D. Siggia, P.S. Swain: *Stochastic Gene Expression in a Single Cell.* Science, 2002, 297, 1183-1186

[Engblom 2009] S. Engblom: *Spectral approximation of solutions to the chemical master equation.* Journal of Computational and Applied Mathematics, 2009, 229, 208-221

[en.wikipedia DoubleHelix 2014] http://en.wikipedia.org/wiki/Double_helix [22.08.2014]

[en.wikipedia Entner-Doudoroff 2014] http://en.wikipedia.org/wiki/Entner%E2%80%93Doudoroff_pathway [22.08.2014]

[en. Wikipedia EssentialAA 2014] http://en.wikipedia.org/wiki/Essential_amino_acid [23.08.2014]

[en.wikipedia Glycolysis 2014] http://en.wikipedia.org/wiki/Glycolysis [22.08.2014]

[en.wikipedia List of biodatabases 2014] http://en.wikipedia.org/wiki/List_of_biological_databases [22.08.2014]

[en.wikipedia NFκB 2014] http://en.wikipedia.org/wiki/NF-%CE%BAB [22. 08. 2014]

[en.wikipedia OHWarburg 2014] http://en.wikipedia.org/wiki/Otto_Heinrich_Warburg [23.08.2014]

[Farcot 2006] E. Farcot: *Geometric properties of a class of piecewise affine biological network models.* Journal of Mathematical Biology, 2006, 52, 373-418

[Fauré, Thieffry 2009] A. Fauré, D. Thieffry: *Logical modeling of cell cycle control in eukaryotes: A comparative study.* Molecular BioSystems, 2009, 5, 1569-1581

[Fell 1997] D. Fell: *Understanding the Control of Metabolism.* 1997, 1st ed., Portland Press

[Fillipov 1988] A.F. Fillipov: *Differential equations with discontinuous right-hand sides.* 1988, 1st ed., Springer

[Friedman et al. 2000] N. Friedman, M. Linial, I. Nachman, D. Pe'er: Using Bayesian networks to analyze expression data. Journal of Computational Biology, 2000, 7 (3) 601-620

[Friedman, Koller 2003] N. Friedman, D. Koller: *Being Bayesian about Network Structure: A Bayesian Approach to Structure Discovery in Bayesian Networks.* Machine Learning, 2003, 50, 95-126

[Fuchs 2013] C. Fuchs: *Inference for Diffusion Processes. With Applications in Life Sciences.* 2013, 1st ed., Springer

[Garg et al. 2009] A. Garg, K. Mohanram, A. Di Cara, G. De Micheli, I. Xenarios: *Modeling stochasticity and robustness in gene regulatory networks.* Bioinformatics, 2009, 25, i101-i109

[Gasca, Sauer 2000] M. Gasca, T. Sauer: *On the history of multivariate polynomial interpolation.* Journal of Computational and Applied Mathematics, 2000, 122 (1,2) 23-35

[Gershenson 2002] C. Gershenson: *Classification of random Boolean networks.* Proceedings of the Eighth International Conference on Artificial Life, 2002, 1-8

[Gershenson 2004] C. Gershenson: Introduction to Random Boolean Networks. arXiv:nlin/0408006v3

[Ghosh et al. 1997] M.K. Ghosh, A. Arapostathis, S.I. Marcus*: Ergodic control of switching diffusions.* SIAM Journal on Control Optimization, 1997, 35 (6) 1952-1988

[Gibson, Bruck 2000] M.A. Gibson, J. Bruck: *Efficient exact stochastic simulation of chemical systems with many species and many channels.* Journal of Physical Chemistry A, 2000, 104, 1876-1889

[Gilks, Wild 1992] W.R. Gilks, P. Wild: *Adaptive rejection sampling for Gibbs sampling.* Journal of the Royal Statistical Society, Series C, 1992, 42, 337-348

[Gillespie 1977] D.T. Gillespie: *Exact stochastic simulation of coupled chemical reactions.* Journal of Physical Chemistry, 1977, 81, 2340-2361

[Gillespie 1992] D.T. Gillespie: *A rigorous derivation of the chemical master equation.* Physica A, 1992, 188, 404-425

[Gillespie 2000] D.T. Gillespie: *The chemical Langevin equation.* Journal of Chemical Physics, 2000, 113, 297-306

[Gillespie 2001] D.T. Gillespie: Approximate accelerated stochastic simulation of chemically reacting systems. Journal of Chemical Physics, 2001, 115, 1716-1733

[Gillespie, Petzold 2003] D.T. Gillespie, L. Petzold: *Improved leap-size selection for accelerated stochastic simulation.* Journal of Chemical Physics, 2003, 119, 8229-8234

[Gillespie, Petzold 2006] D.T. Gillespie, L. Petzold: *Numerical Simulation for Biochemical Kinetics.* Chapter 16 in [Szallasi et al. (ed.) 2006]

[Glass, Kauffman 1973] L. Glass, S. Kauffman: *The logical analysis of continuous, non-linear biochemical control networks.* Journal of Theoretical Biology, 1973, 39, 103-129

[Glass, Siegelmann 2010] L. Glass, H.T. Siegelmann: *Logical and symbolic analysis of robust biological dynamics.* Current Opinion in Genetics and Development, 2010, 20, 644-649

[Golightly, Wilkinson 2005] A. Golightly, D.J. Wilkinson: *Bayesian inference for stochastic kinetic models using a diffusion approximation.* Biometrics, 2005, 61, 781-788

[Goncalves et al. 2013] E. Goncalves, J. Bucher, A. Ryll, J. Niklas, K. Mauch, S. Klamt, M. Rocha, J. Saez-Rodriguez: *Bridging the layers: towards integration of signal transduction, regulation and metabolism into mathematical models.* Molecular Systems Biology, 2013, 9, 1576-1583

[Gouzé, Sari 2002] J.-L. Gouzé, T. Sari: *A class of piecewise linear differential equations arising in biological models.* Dynamical Systems, 2002, 17 (4) 299-316

[Guckenheimer, Holmes 2002] J. Guckenheimer, P. Holmes: *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields.* 2002, corrected 7th printing, Springer

[Haas 2002] P.J. Haas: *Stochastic Petri nets: Modeling, Stability, Simulation.* 2002, 1st ed., Springer

[Harris et al. 2002] S.E. Harris, B.K. Sawhill, A. Wuensche, S. Kauffman: *A model of transcriptional regulatory networks based on biases in the observed regulation rules.* Complexity, 2002, 7 (4) 23-40

[Harvey, Bossomaier 1997] I. Harvey, T. Bossomaier: *Time out of joint: attractors in asynchronous random Boolean networks.* Proceedings of the Fourth European Conference on Artifical Life, 1997, 67-75

[Hasenauer et al. 2014] J. Hasenauer, V. Wolf, A. Kazeroonian, F.J. Theis: *Method of conditional moments (MCM) for the Chemical Master Equation. A unified framework for the method of moments and hybrid stochastic-deterministic models.* Journal of Mathematical Biology, 2014, 69 (3) 687-735

[Haggart et al. 2011] C.R. Haggart, J.A. Bartlett, J.J. Saucerman, J.A. Papin: *Whole-genome metabolic network reconstruction and constraint-based modeling.* Methods in Enzymology, 2011, 500, 411-433

[Hardy, Robillard 2004] S. Hardy, P.N. Robillard: *Modeling and simulation of molecular biology systems using Petri nets: Modeling goals of various approaches.* Journal of Bioinformatics and Computational Biology, 2 (4) 595-613

[Hefzi et al. 2013] H. Hefzi, B.O. Palsson, N.E. Lewis: *Reconstruction of genome-scale metabolic networks.* Chapter 12 in [Walhout et al. (ed.) 2013]

[Heiner et al. 2004] M. Heiner, I. Koch, J. Will: *Model validation of biological pathways using Petri nets – demonstrated for apoptosis.* Biosystems, 2004, 75, 15-28

[Heinrich, Schuster 1996] R. Heinrich, S. Schuster: *The Regulation of Cellular Systems.* 1996, 1st ed., Springer

[Henzinger et al. 2010] T.A. Henzinger, L. Mikeev, M. Mateescu, V. Wolf: *Hybrid numerical solution of the chemical master equation.* Proceedings of the 8th International Conference on Computational Methods in Systems Biology, 55-65

[Hock 2010] S. Hock: *Spatial modeling of differentiation of mid- and hindbrain.* 2010, Diplomarbeit, München http://push-zb.helmholtz-muenchen.de/frontdoor.php?source_opus=28684&la=en

[Hofstädt, Thelen 1998] R. Hofstädt, S. Thelen: *Quantitative modeling of biochemical networks.* In Silicio Biology, 1998, 1, 39-53

[Hu et al. 2000] J. Hu, J. Lygeros, S. Sastry: *Towards a theory of stochastic hybrid systems.* In: N. Lynch, B.H. Krogh (eds.): *Hybrid Systems: Computation and Control.* 2000, 1st ed., Springer LNCS 1790

[Huang, Hahn 2009] Z. Huang, J. Hahn: *Fuzzy modeling of signal transduction networks.* Chemical Engineering Science, 2009, 64, 2044-2056

[Huang et al. 2009] Y. Huang, I.M. Tienda-Luna, Y. Wang: *Reverse Engineering Gene Regulatory Networks.* IEEE Signal Processing Magazine, January 2009, 76-97

[Ivanov, Dougherty 2006] I. Ivanov, E.R. Dougherty: *Modeling genetic regulatory networks: continuous or discrete?* Journal of Biological Systems, 2006, 14 (2) 219-229

[Jaeger 2009] J. Jaeger: *Modeling the Drosophila embryo.* Molecular BioSystems, 5 (12) 1549-1568

[Jamshidi 2012] S. Jamshidi: *Comparing discrete, continuous and hybrid modeling approaches of gene regulatory networks.* PhD thesis, 2012 Berlin http://www.diss.fu-berlin.de/diss/receive/FUDISS_thesis_000000094522

[Jamshidi et al. 2013] S. Jamshidi, H. Siebert, A. Bockmayr: *Preservation of Dynamic Properties in Qualitative Modeling Frameworks for Gene Regulatory Networks.* Biosystems, 2013, 112 (2) 171-179

[Joshi-Tope et al. 2004] G. Joshi-Tope, M. Gillespie, I. Vastrik, P. d'Eustachio, E. Schmidt, B. De Bono, B. Jassal, G. Gopinath, G. Wu, L. Matthews, S. Lewis, E. Birney, L. Stein: *Reactome: A knowledgebase of biological pathways.* Nucleic Acids Research, 2004, 33 D428-D432

[Jaqaman, Danuser 2006] K. Jaqaman, G. Danuser: *Linking data to models: data regression.* Nature Reviews Molecular Cell Biology, 2006, 7, 813-819

[Kaplan et al. 2008] S. Kaplan, A. Bren, A. Zaslaver, E. Dekel, U. Alon: *Diverse two-dimensional input functions control bacterial sugar genes.* Molecular Cell, 2008, 29, 786-792

[Karpfinger, Meyberg 2010] C. Karpfinger, K. Meyberg: *Algebra: Gruppen – Ringe – Körper.* 2010, 2. Aufl., Spektrum Akademischer Verlag

[Kauffman 1969] S. Kauffman: *Metabolic stability and epigenesist in randomly connected nets.* Journal of Theoretical Biology, 1969, 22, 437-467

[Kauffman 2000] S. Kauffman: *A proposal for using the ensemble approach to understand genetic regulatory networks.* Journal of Theoretical Biology, 2000, 230, 581-590

[Kauffman 2004] S. Kauffman: *A proposal for using the ensemble approach to understand genetic regulatory networks*. Journal of Theoretical Biology, 2004, 230, 581-590

[Kauffman et al. 2003] S. Kauffman, C. Peterson, B. Samuelsson, C. Troein: *Random Boolean network models and the yeast transcriptional network.* PNAS, 2003, 100 (25) 14796-14799

[Kauffman et al. 2004] S. Kauffman, C. Peterson, B. Samuelsson, C. Troein: Genetic networks with canalyzing Boolean rules are always stable. PNAS, 2004, 101 (49) 17102-17107

[Kazemzadeh et al. 2012] L. Kazemzadeh, M. Cvijovic, D. Petranovic: *Boolean model of yeast apoptosis as a tool to study yeast and human apoptotic regulations.* Frontiers in Physiology, 2012, 3, Article 446

[Keener, Sneyd 2009] J. Keener, J. Sneyd: *Mathematical Physiology. I: Cellular Physiology.* 2009, 2nd ed., Springer

[Kell, Knowles 2006] D.B. Kell, J.D. Knowles: *The role of modeling is systems biology.* Chapter 1 in [Szallasi et al. (ed.) 2006]

[Kerr et al. 1972] J.F. Kerr, A.H. Wyllie, A.R. Currie: *Apoptosis: a basic biological phenomenon with wide-ranging implications in tissue kinetics.* British Journal of Cancer, 1972, 26 (4) 239-257

[Kim et al. 2002] S. Kim, H. Li, E.R. Dougherty, N. Cao, Y. Chen, M. Bittner, E.B. Suh: *Can Markov chain models mimic biological regulation?.* Journal of Biological Systems, 2002, 10 (4) 337-357

[Kitano 2004] H. Kitano: *Biological Robustness.* Nature Reviews Genetics, 2004, 5, 826-837

[Kitano 2007] H. Kitano: *Towards a theory of biological robustness.* Molecular Systems Biology, 2007, 3:137

[Klamt et al. 2006] S. Klamt, J. Saez-Rodriguez, J.A. Lindquist, L. Simeoni, E.D. Gilles: *A methodology for the structural and functional analysis of signaling and regulatory networks.* BMC Bioinformatics, 2006, 7:56

[Klamt et al. 2007] S. Klamt, J. Saez-Rodriguez, E.D. Gilles: *Structural and functional analysis od cellular networks with CellNetAnalyzer.* BMC Systems Biology, 2007, 1:2

[Klamt et al. 2009] S. Klamt, UU. Haus, F. Theis: *Hypergraphs and Cellular Networks.* PLoS Computational Biology, 2009, 5 (5) e1000385

[Klamt, Stelling 2006] S. Klamt, J. Stelling: *Stochiometric and Constraint-based Modeling.* Chapter 5 in [Szallasi et al. (ed.) 2006]

[Klenke 2008] A. Klenke: *Wahrscheinlichkeitstheorie.* 2008, 2. Aufl., Springer

[Klipp et al. 2009] E. Klipp, W. Liebermeister, C. Wierling, A. Kowald, H. Lehrach, R. Herwig: *Systems Biology: A textbook*. 2009, 1st ed., Wiley-VHC

[Koch et al. (eds.) 2004] I. Koch, W. Reisig, F. Schreiber (eds.): *Modeling in systems biology: the Petri net approach.* 2004, 1st ed., Springer

[Kouretas et al. 2006] P. Kouretas, K. Koutroumpas, J. Lygeros, Z. Lygerou: *Stochastic Hybrid Modeling of Biochemical Processes.* Chapter 9 in [Cassandras, Lygeros (eds.) 2007]

[Kremling 2012] A. Kremling: *Kompendium Systembiologie. Mathematische Modellierung und Modellanalyse.* 2012, 1. Aufl., Vieweg + Teubner

[Krumsiek et al. 2010] J. Krumsiek, S. Pölsterl, D.M. Wittmann, F.J. Theis: *Odefy – From discrete to continuous models.* BMC Bioinformatics, 2010, 11:233

[Kruse, Elf 2006] K. Kruse, J. Elf: *Kinetics in Spatially Extended Systems.* Chapter 9 in [Szallasi et al. (eds.) 2006]

[Kulkarni, Perrimon 2013] M.M. Kulkarni, N. Perrimon: *Analyzing the Structure, Function and Information Flow in Signaling Networks using Quantitative Cellular Signatures.* Chapter 5 in [Walhout et al. (ed.) 2013]

[Kwiatkowska et al. 2000] M. Kwiatowska, G. Norman, R. Segala, J. Sprosto: *Verifying quantitative properties of continuous probabilistic timed automata.* 2000, 1st ed., Springer LNCS 1877

[Lähdesmäki et al. 2006] H. Lähdesmäki, S. Hautaniemi, I. Shmulevich, O. Yli-Harja: *Relationships between probabilistic Boolean networks and dynamic Bayesian networks as models of gene regulatory networks.* Signal Processing, 2006, 86, 814-834

[Lewin 2008] B. Lewin: *Genes IX.* 2008, 9th ed., Jones and Bartlett

[Li et al. 2007] C. Li, Q.W. Ge, M. Nakata, H. Matsuno, S. Miyano: *Modelling and simulation of signal transductions in an apoptosis pathway by using timed Petri nets.* Journal of Biosciences, 2007, 32 (1) 113-127

[Liang et al. 1998] S. Liang, S. Fuhrman, R. Somogyi: *REVEAL, a general reverse engineering algorithm for inference of genetic network architectures.* Pacific Symposium on Biocomputing, 1998, 3, 18-29

[Lunze, Lamnabhi-Lagarrigue (eds.) 2009] J. Lunze, F. Lamnabhi-Lagarrigue: *Handbook of Hybrid Systems Control: Theory, Tools, Applications.* 2009, 1st ed., Cambridge University Press

[Lygeros et al. 2003] J. Lygeros, K.H. Johansson, S.N. Simic, J. Zhang, S. Sastry: *Dynamical properties of hybrid automata.* IEEE Transactions on Automatic Control, 2003, 48 (1) 2-17

[Lygeros 2004] J. Lygeros: Lecture Notes on Hybrid Systems. http://robotics.eecs.berkeley.edu/~sastry/ee291e/lygeros.pdf

[Maamar et al. 2007] H. Maamar, A. Raj, D. Dubnau: *Noise in gene expression determines cell fate in Bacillus subtilis.* Science, 2007, 317, 526-529

[Macía et al. 2009] J. Macía, S. Widder, R. Solé: *Why are cellular switches Boolean? General conditions for multistable genetic circuits.* Journal of Theoretical Biology, 2009, 261, 126-135

[Mai, Liu 2009] Z. Mai, H. Liu: *Boolean network-based analysis of the apoptosis network: Irreversible apoptosis and stable surviving*. Journal of Theoretical Biology, 259, 760-769

[Markowetz, Spang 2007] F. Markowetz, R. Spang: *Inferring cellular networks – a review.* BMC Bioinformatics, 8 (Suppl 6):S5

[Mariottini, Iyengar 2013] C. Mariottini, R. Iynegar: *System Biology of Cell Signaling.* Chapter 16 in [Walhout et al. (ed.) 2013]

[Marsan et al. 1995] M.A. Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis: *Modeling with generalized stochastic Petri nets.* 1995, 1st ed., Wiley

[Matsuno et al. 2000] H. Matsuno, A. Doi, M. Nagasaki, S. Miyano: *Hybrid Petri net representation of gene regulatory networks.* Proceedings Pacific Symposium Biocomputing 2000, 341-352

[Mbodj et al. 2013] A. Mbodj, G. Junion, C. Brun, E.E.M. Furlong, D. Thieffry: *Logical modeling of Drosophila signaling pathways.* Molecular BioSystems, 2013, 9, 2248

[Mendoza et al. 1999] L. Mendoza, D. Thieffry, E.R. Álvarez-Buylla: *Genetic control of flower morphogenesis in Arabidopsis thaliana: A logical analysis.* Bioinformatics, 1999, 15 (7,8) 593-606

[Mendoza, Xenarios 2006] L. Mendoza, I. Xenarios: *A method for the generation of standardized qualitative dynamical systems of regulatory networks.*
Theoretical Biology and Medical Modelling, 2006, 3:13

[Mesot, Teuscher 2003] B. Mesot, C. Teuscher: *Critical Values in Asynchronous Random Boolean Networks.*
In: W. Banzafh, J. Ziegler, T. Christaller, P. Dittrich, J.T. Kim (eds.) *Advances in Artifical Life.*
2003, 1st ed., Springer LNCS 2801

[Michaelis, Menten 1913] L. Michaelis, M.L. Menten: *Die Kinetik der Invertinwirkung.*
Biochemische Zeitschrift 1913, 49, 333-369

[Michal, Schomburg 2012] G. Michal, D. Schomburg: *Biochemical Pathways: An Atlas of Biochemistry and Molecular Biology.* 2012, 2nd ed., John Wiley & Sons
see also http://biochemical-pathways.com/#/map/1

[Miranda-Saavedra, Göttgens 2008] D. Miranda-Saavedra, B. Göttgens:
Transcriptional regulatory networks in haematopoiesis.
Current Opinions in Genetics & Development, 2008, 18 (6) 530-535

[Morris et al. 2010] M.K. Morris, J. Saez-Rodriguez, P.K. Sorger, D.A. Lauffenburger: *Logic-Based Models for the Analysis of Cell Signaling Networks.* Biochemistry, 2010, 49, 3216-3224

[Mugler et al. 2009] A. Mugler, A.M. Walczak, C.H. Wiggins: *Spectral solutions to stochastic models of gene expression with bursts and regulation.* Physical Review E, 2009, 80, 041921

[Munk et al. 2008] herausgegeben von K. Munk: *Biochemie-Zellbiologie.* 2008, 1. Aufl., Thieme Verlag

[Munsky, Kammash 2006] B. Munsky, M. Kammash: *The finite state projection algorithm for the solution of the chemical master equation.* Journal of Chemical Physics, 124, 044104

[Murata 1989] T. Murata: *Petri nets: Properties, analysis and applications.*
Proceedings IEEE, 1989, 77, 541-580

[Murphy 2012] K.P. Murphy: *Machine Learning. A Probabilistic Perspective.* 2012, 1st ed., MIT Press

[Murphy, Mian 1999] K. Murphy, S. Mian: *Modeling gene expression data using dynamics Bayesian networks.*
Technical Report, Computer Science Division, University of California, Berkeley

[Murray 2004] J.D. Murray: *Mathematical Biology I: An Introduction.* 2004, 3rd ed., Springer

[Murray 2008] J.D. Murray: Mathematical Biology II: Spatial Models and Biomedical Applications.
2008, 3rd ed., Springer

[Murrugarra et al. 2012] D. Murrugarra, A. Veliz-Cuba, B. Aguilar, S. Arat, R. Laubenbacher:
Modeling stochasticity and variability in gene regulatory networks.
Journal on Bioinformatics and Systems Biology, 2012, 2012:5

[Nakajima et al. 2005] M. Nakajima, K. Imai, H. Ito, T. Nishiwaki, Y. Murayama, H. Iwasaki, T. Oyama , T. Kondo: *Reconstitution of circadian oscillation of cyanobacterial KaiC phosphorrylation in vitro*. Science, 2005, 308 (5720) 414–415

[Nikolajewa et al. 2007] S. Nikolajewa, M. Friedel, T. Wilhelm: *Boolean networks with biologically relevant rules show ordered behavior.* Biosystems, 2007, 90, 40-47

[Norris 1998] J.R. Norris: *Markov chains.* 1998, 1st ed., Cambridge University Press

[Novak et al. 2001] B. Novak, Z. Pataki, A. Ciliberto, J.J. Tyson: *Mathematical model of the cell division cycle of fission yeast.* Chaos, 2001, 11 (1) 277-286

[Oksendal 2010] B. Oksendal: *Stochastic Differential Equations. An Introduction with Applications.* 2010, 6th ed., Springer

[Öktem et al. 2002] H. Öktem, R. Pearson, O. Yli-Harja, D. Nicorici, K. Egiazarian, J. Astola: *A computational model for simulating continuous time Boolean networks.* Proceedings of IEEE international Workshop on Genomic Signal Processing and Statistics 2002

[Öktem et al. 2003] H. Öktem, R. Pearson, K. Egiazarian: An *adjustable aperiodic model class* of *genomic interactions using continuous time Boolean networks* (*Boolean delay equations*). Chaos, 2003, 13 (4) 1167-1174

[Oppenheim et al. 2005] A.B. Oppenheim, O. Kobiler, J. Stavans, D.L. Court, S. Adhya: *Switches in Bacteriophage Lambda Development.* Annual Review of Genetics, 2005, 39, 409-429

[Orphanides, Reiberg 2002] G. Orphanides, D. Reinberg: *A unified theory of gene expression.* Cell, 2002, 108 (4) 439-451

[Pedraza, van Oudengaarden 2005] J.M. Pedraza, A. van Oudengaarden: *Noise Propagation in Gene Networks.* Science, 2005, 307, 1965-1969

[Petri 1962] C.A. Petri: *Kommunikation mit Automaten.* Institut für Instrumentelle Mathematik Bonn, Schriften des IIM 3, 1962

[Petri 1963] C.A. Petri: *Fundamental of a theory of asynchronous information flow.* Proceedings, International Federation for Information Processing 62, 1962, 386-390

[Plathe et al. 1998] E. Plathe, T. Mestl, S.W. Omholt: *A methodological basis for description and analysis os systems with complex switch-like interactions.* Journal of Mathematical Biology, 1998, 36, 321-348

[Pola et al.] G. Pola, M.L. Bujorianu, J. Lygeros, M.D. Di Benedetto: *Stochastic hybrid models: An overview.* In: S. Engell, H. Gueguen, J. Zaytoon (eds.): *Analysis and Design of Hybrid Systems 2003.* 2003, 1st ed., Elsevier IFAC

[Ptashne 2004] M. Ptashne: *A genetic switch: Phage lambda revisited.* 2004, 3rd ed., CSH Press

[Purves et al. 2006] W. K. Purves, D. Sadava, G. H. Orians, H. C. Heller, herausgegeben von J. Markl: *Biologie.* 2006, 7. Aufl., Elsevier Spektrum Akademischer Verlag

[Quateroni et al. 2002] A. Quateroni, R. Sacco, F. Saleri: *Numerische Mathematik 2.* 2002, 2. Aufl., Springer

[Raeymaekers 2002] L. Raeymaekers: *Dynamics of Boolean networks controlled by biologically meaningful functions.* Journal of Theoretical Biology, 2002, 218 (3) 331-341

[Raj, van Oudengaarden 2008] A. Raj, A. van Oudengaarden: *Nature, Nurture, or Chance: Stochastic Gene Expression and its Consequences.* Cell, 2008, 135, 216-226

[Raue et al. 2009] A. Raue, C. Kreutz, T. Maiwald, J. Bachmann, M.Schilling, U. Klingmüller, J. Timmer: *Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood.* Bioinformatics, 2009, 25 (15) 1923-1929

[Raue et al. 2014] A. Raue, J. Karlsson, M.P. Saccomani, M. Jirstrand, J. Timmer: *Comparison of approaches for parameter identifiability of biological systems.* Bioinformatics, 2014, 30 (10) 1440-1448

[Reed, Zwart 2011] J. Reed, B. Zwart: *A piecewise linear stochastic differential equation driven by a Lévy process.* Journal of Applied Probability, 2011, Special Volume 48A, 109-119

[Reinsch 1967] C. Reinsch: *Smoothing by Spline Functions.* Numerische Mathematik, 1967, 10, 177-183

[Rizzo 2008] M.L. Rizzo: *Statistical Computing with R.* 2008, 1st ed., Chapman & Hall/CRC

[Robert, Casella 2000] C.P. Robert, G. Casella: *Monte Carlo Statistical Methods.* 2000, 1st ed., Springer

[Rohlf, Bornholdt 2002] T. Rohlf, S. Bornholdt: *Criticality in random Boolean threshold networks: annealed approximation and beyond.* Physica A, 2002, 310 (1,2) 245-259

[Saadatpour et al. 2011] A. Saadatpour, R.-S. Wang, A. Liao, X. Liu, T.P. Loughran, I. Albert, R. Albert: *Dynamical and structural analysis of a T cell survival network identifies novel candidate therapeutic targets for large granular lymphocyte leukemia.* PLoS Computational Biology, 2011, 7 (11) e1002267

[Saadatpour et al. 2010] A. Saadatpour, I. Albert, R. Albert: *Attractor analysis of asynchronous Boolean models of signal transduction networks.* Journal of Theoretical Biology, 2010, 266, 641-656

[Sackmann et al. 2006] A. Sackmann, M. Heiner, I. Koch: *Application of Petri net based analysis techniques to signal transduction pathways.* BMC Bioinformatics, 2006, 7:482

[Saeed et al. 2012] M. Saeed, M. Ijaz, K. Javed, H.A. Babri: *Reverse Engineering Boolean Networks: From Bernoulli Mixture Models to Rule Based Systems.* PLoS One, 2012, 7 (12) e51006

[Saez-Rodriguez et al. 2004] J. Saez-Rodriguez, A. Kremling, H. Conzelmann, K. Bettenbrock, E.D. Gilles: *Modular analysis of signal transduction networks.* Control Systems IEEE, 2004, 24 (4) 35-52

[Saez-Rodriguez et al. 2007] J. Saez-Rodriguez, L. Simeoni, J.A. Lindquist, R. Hemenway, U. Bommhardt, B. Arndt, U.-U. Haus, R. Weismantel, E.D. Gilles, S. Klamt, B. Schraven: *A logical model provides insights into T cell receptor signaling.* PLoS Computational Biology, 2007, 3 (8) e163

[Samaga, Klamt 2013] R. Samaga, S. Klamt: *Modeling approaches for qualitative and semi-quantitative analysis of cellular signaling networks.* Cell Communication and Signaling, 2013, 11:43

[Sánchez, Thieffry 2001] L. Sánchez, D. Thieffry: *A logical analysis of the Drosophila gap-gene system.* Journal of Theoretical Biology, 2001, 211, 115-141

[Sánchez, Thieffry 2003] L. Sánchez, D. Thieffry: *Segmenting the fly embryo: a logical analysis of the pair-rule cross-regulatory module.* Journal of Theoretical Biology, 2003, 224, 517-537

[Sánchez et al. 2008] L. Sánchez, C. Chaouyia, D. Thieffry: *Segmenting the fly embryo: a logical analysis of the segment polarity cross-regulatory module.* International Journal of Developmental Biology, 2008, 52 (8) 1059-1075

[Savageau 2009] M. A. Savageau: *Biochemical Systems Analysis. A study of function and design in molecular biology.* 2009, 40th anniversary ed., M. A. Savageau

[Schlatter et al. 2009] R. Schlatter, K. Schmich, I.A. Vizcarra, P. Scheurich, T. Sauter, C. Borner, M. Ederer, I. Merfort, O. Sawodny: *ON/OFF and beyond – A Boolean model of apoptosis.* PLoS Computational Biology, 2009, 5 (12) e1000595

[Setty et al. 2003] Y. Setty, A.E. Mayo, M.G. Surette, U. Alon: *Detailed map of a cis-regulatory input function.* PNAS, 2003, 100 (13) 7702-7707

[Shmulevich et al. 2002a] I. Shmulevich, E.R. Dougherty, W. Zhang: *From Boolean to probabilistic Boolean networks as models of genetic regulatory networks.* Proceedings IEEE, 2002, 90, 1778-1792

[Shmulevich et al. 2002b] I. Shmulevich, E.R. Dougherty, S. Kim, W. Zhang: *Probabilistic Boolean networks: A rule-based uncertainty modelf for gene regulatory networks.* Bioinformatics, 2002, 18, 261-274

[Shmulevich, Dougherty 2010] I. Shmulevich, E.R. Dougherty: *Probabilistic Boolean Networks. The Modeling and Control of Gene Regulatory Networks.* 2010, 1st ed., SIAM

[Siebert, Bockmayr 2009] H. Siebert, A. Bockmayr: *Temporal constraints in the logical analysis of regulatory networks.* Theoretical Computer Science, 2009, 391, 258-275

[Simpson, Kuske 2012] D.J.W. Simpson, R. Kuske: *Stochastically Perturbed Sliding Motion in Piecewise-Smooth Systems.* arXiv:1204.5792v1

[Singh, Hespanha 2010] A. Singh, J.P. Hespanha: *Stochastic hybrid systems for studying biochemical processes.* Philosophical Transactions of the Royal Society A, 2010, 368, 4995-5011

[Singh, Weinberger 2009] A. Singh, L.S. Weinberger: *Stochastic gene expression as a molecular switch for viral latency.* Current Opinions in Microbiology, 2009, 12 (4) 460-466

[Snoussi 1989] E.H. Snoussi: *Qualitative dynamics of piecewise-linear differential equations: a discrete mapping approach.* Dynamics and Stability of Systems, 1989, 4 (3,4) 189-207

[Snoussi, Thomas 1993] E.H. Snoussi, R. Thomas: *Logical identification of all steady states: The concept of feedback loop characteristic states.* Bulletin of Mathematical Biology, 1993, 55 (5) 973-991

[St. Pierre, Endy 2008] F. St. Pierre, D. Endy: *Determination of cell fate selection during phage lambda infection.* PNAS, 2008, 105 (52) 20705-20710

[Steggles et al. 2006] L.J. Steggles, R. Banks, O. Shaw, A. Wipat: *Qualitatively modeling and analyzing genetic regulatory networks: A Petri net approach.* Bioinformatics, 23, 336-343

[Stelling et al. 2004] J. Stelling, U. Sauer, Z. Szallasi, F.J. Doyle III, J. Doyle: *Robustness of Cellular Functions.* Cell, 2004, 118, 675-685

[Stewart 1990] D. Stewart: *A high accuracy method for solving ODEs with discontinuous right-hand sides.* Numerische Mathematik, 1990, 58, 299-328

[Stoll et al. 2012] G. Stoll, E. Viara, E. Barillot, L. Calzone: *Continuous time Boolean modeling for biological signaling: application of the Gillespie algorithm.* BMC Systems Biology, 6:116

[Sugeno, Yasukawa 1993] M. Sugeno, T. Yasukawa: *A fuzzy-logic-based approach to qualitative modeling.* IEEE Transactions in Fuzzy Systems, 1993, 1 (1) 7-31

[Szallasi et al. (ed.) 2006] Z. Szallasi, J. Stelling, V. Periwal (editors): *System Modeling in Cellular Biology.* 2006, 1st ed., MIT press

[Szallasi et al. 2006] Z. Szallasi, V. Periwal, J. Stelling: *On modules and modularity.* Chapter 3 in [Szallasi et al. (ed.) 2006]

[Szejka et al. 2008] A. Szejka, T. Mihaljev, B. Drossel: *The phase diagram of random threshold networks.* New Journal of Physics, 2008, 10, 063009

[Teraguchi et al. 2011] S. Teraguchi, Y. Kumagai, A. Vandenbon, S. Akira, D.M. Standley: *Stochastic binary modeling of cells in continuous time as an alternative to biochemical reaction equations.* Phyiscal Review E, 2011, 84, 062903

[Thakar et al. 2007] J. Thakar, M. Pilione, G. Kirimanjeswara, E.T. Harvill, R. Albert: *Modeling systems-level regulation of host-immune responses.* PLoS Computational Biology, 2007, 3, e109

[Thieffry, Thomas 1995] D. Thieffry, R. Thomas: *Dynamical behavior of biological regulatory networks -II. Immunity control in bacteriophage lambda.* Bulletin of Mathematical Biology, 1995, 57 (2) 277-297

[Thomas, D'Ari 1990] R. Thomas, R. D'Ari: *Biological feedback.* 1990, 1st ed., CRC Press

[Thomas 1973] R. Thomas: *Boolean formalization of genetic control circuits.* Journal of Theoretical Biology, 1973, 42, 565-583

[Thomas (ed.) 1979]: R. Thomas (ed.): Kinetic Logic: A Boolean Approach to the Analysis of Complex Regulatory Systems. 1979, 1st ed., Springer

[Thomas 1979] R. Thomas: *Integration-excision in lambdoid phages: its relation with immunity.* In [Thomas (ed.) 1979], 366-379

[Thomas 1991] R. Thomas: *Regulatory networks seen as asynchronous automata: A logical description.* Journal of Theoretical Biology, 1991, 153, 1-23

[Thomas et al. 1995] R.Thomas, D. Thieffry, M. Kaufman: *Dynamical behavior of biological regulatory networks-I. Biological role of feedback loops and practical use of the concept of the loop-characteristic state.* Bulletin of Mathematical Biology, 1995, 57 (2) 247-276

[Thomas 2013] R. Thomas: *Remarks on the respective roles of Logical Parameters and Time Delays in asynchronous logic: An homage to El Houssine Snoussi.* Bulletin of Mathematical Biology, 2013, 75, 896-904

[Torres, Voit 2002] N.V. Torres, E.O. Voit: *Pathway Analysis and Optimization in Metabolic Engineering.* 2002, 1st ed., Cambridge University Press

[Tournier, Chaves 2009] L. Tournier, M. Chaves: *Uncovering operational interactions in genetic networks using asynchronous Boolean dynamics.* Journal of Theoretical Biology, 2009, 260, 196-209

[Tyson et al. 2003] J.J. Tyson, K.C. Chen, B. Novak: *Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell.* Current Opinion in Cell Biology, 2003, 15, 221-231

[Ullah, Wolkenhauer 2011] M. Ullah, O. Wolkenhauer: *Stochastic Approaches for Systems Biology.* 2011, 1st ed., Springer

[Van Ham 1973] P. Van Ham: *How to deal with variables with more than two levels.* In [Thomas (ed.) 1979], 326-343

[Veliz-Cuba et al. 2012] A. Veliz-Cuba, J. Arthur, L. Hochstetler, V. Klomps, E. Korpi: *On the relationship of steady states of continuous and discrete models arising from biology.* Bulletin of Mathematical Biology, 2012, 74, 2779-2792

[Vilar, Leibler 2003] J.M.G. Vilar, S. Leibler: *DNA looping and physical constraints on transcription regulation.* Journal of Molecular Biology, 2003, 331, 981-989

[Wagner 2005] A. Wagner: *Robustness and Evolvability in Living Systems.* 2005, 1st ed., Princeton Universtiy Press

[Walhout et al. (ed.) 2013] M. Walhout, M. Vidal, J. Dekker (editors): *Handbook of systems biology: Concepts and insights*. 2013, 1st ed., Academic Press

[Walpole et al. 2013] J. Walpole, J.A. Papin, S.M. Pierce: *Multiscale Computational Models of Complex Biological Systems.* Annual Reviews of Biomedical Engineering, 2013, 15, 137-154

[Walter 2000] W. Walter: *Gewöhnliche Differentialgleichungen.* 2000, 7. Aufl., Springer

[Wang et al. 2012] R.-S. Wang, A. Saadatpour, R. Albert: *Boolean modeling in systems biology: an overview of methodology and applications.* Physical Biology, 2012, 9, 055001

[Wilkinson 2009] D.J. Wilkinson: *Stochastic modelling for quantitative description of heterogeneous biological systems.* Nature Reviews Genetics, 2009, 10, 122-133

[Willadsen, Wiles 2007] K. Willadsen, J. Wiles: *Robustness and state space structure of Boolean gene regulatory networks.* Journal of Theoretical Biology, 2007, 249, 749-765

[Wittmann et al. 2009a] D.M. Wittmann, J. Krumsiek, J. Saez-Rodriguez, D.A. Lauffenburger, S. Klamt, F.J. Theis: *Transforming Boolean models to continuous models: methodology and applications to T-cell receptor signaling.* BMC Systems Biology, 2009, 3:98

[Wittmann et al. 2009b] D.M. Wittmann, F. Blöchl, D. Trümbach, W. Wurst, N. Prakash, F.J. Theis: *Spatial analysis of expression patterns predicts genetic interactions at the mid-hindbrain boundary.* PLoS Computational Biology, 5 (11) e1000569

[Wittmann 2010] D.M. Wittmann: *Beyond Boolean Modeling in Systems Biology.* 2010, PhD thesis, München http://push-zb.helmholtz-muenchen.de/frontdoor.php?source_opus=25942&la=en

[Wittmann et al. 2010] D.M. Wittmann, C. Marr, F.J. Theis: *Biologically meaningful update rules increase the critical connectivity of generalized Kauffman networks.* Journal of Theoretical Biology, 2010, 266, 436-448

[Wittmann, Theis 2011]: D.M. Wittmann, F.J. Theis: *Dynamic regimes of random fuzzy logic networks.* New Journal of Physics, 2011, 13, 013041

[Wolpert et al. 2011] L. Wolpert, C. Tickle, P. Lawrence, E. Meyerowitz, E. Robertson, J. Smith, T. Jessell: *Principles of Development.* 2011, 4th ed., Oxford University Press

[Wolf et al. 2010] V. Wolf, R. Goel, M. Mateescu, T. Henzinger: *Solving the chemical master equation using sliding windows.* BMC Systems Biology, 4:42

[Yin, Zhu 2010] G.G. Yin, C. Zhu: *Hybrid Switching Diffusions: Properties and Applications.* 2010, 1st ed., Springer

[Zadeh 1965] L.A. Zadeh: *Fuzzy sets.* Information and Control, 1965, 8 (3) 338-353

[Zadeh 1995] L.A. Zadeh: *Fuzzy logic = Computing with Words.*
IEEE Transactions on Fuzzy Systems, 1995, 4 (2) 103-111

[Zaslaver et al. 2004] A. Zaslaver, A.E. Mayo, R. Rosenberg, P. Bashkin, H. Sberro, M. Tsalyuk, M.G. Surette, U. Alon: *Just-in-time transcription program in metabolic pathways.*
Nature Genetics, 2004, 36 (5) 486-491

[Zeng et al. 2010] L. Zeng, S.O. Skinner, C. Zong, J. Sippy, M. Feiss, I. Golding: *Decision making at a subcellular level determines the outcome of bacteriophage infection.*
Cell, 2010, 141 (4) 682-691

[Zevedei-Oancea, Schuster 2003] I. Zevedei-Oancea, S. Schuster: *Topological analysis of metabolic networks based in Petri net theory.* In Silicio Biology, 2003, 3, 0029

[Zhang et al. 2008] R. Zhang, M.V. Shah, J. Yang, S.B. Nyland, X. Liu, J.K. Yun, R. Albert, T.P. Loughran, Jr.: *Network model of survival signaling in large granular lymphocyte leukemia.*
PNAS, 2008, 105 (42) 16308-16313

[Zielinski et al. 2008] R. Zielinski, P.F. Przytycki, J. Zheng, D. Zhang, T.M. Przytycka, J. Capala: *The crosstalk between EGF, IGF, and insulin cell signaling pathways: Computations and and experimental analysis.* BMC Systems Biology, 2008, 3:88